Building and Improving Artificial Neural Network Classifier

Sanghita Saha, Ramanathan.L

Abstract*: Deep learning is a spectrum of machine learning which uses advanced neural networks to solve the various machine learning problems. Its working is very similar to the working of a human brain where the models take decision based on various input parameters. There are multiple open source libraries which implement neural networks, like Tensorflow, Theano, PyTorch, Keras etc. In this paper we have proposed a generic architecture that can be used for any type of classification problems with binary output or classification output using Deep Learning model: Artificial Neural Network (ANN). In the architectural model after Data preprocessing we first build the ANN classifier using Keras library with Tensorflow backends, second step we have apply Cross-validation method for better accuracy. Then we perform Dropout Regularization method for preventing from overfitting and at last we have applied grid search technique for parameter tuning that basically will test several combinations of Hyperparameter values and will eventually return the best selection choice with K-Fold cross validation. And the experimental results shows in higher accuracy with ours proposed architecture and in our proposed architecture results we remove the randomness from the model. In the proposed architecture we can again rebuild developing our model using Keras Callback function by using this feature in our model it does not create any major difference in terms of accuracy. But as we know the accuracy will vary with parameter tuning. The main advantage of using Keras Callback function method is it's saves a lot of time for building model and it is easy for debugging the model.

Index Terms*: Deep Learning, Keras, Tensorflow, ANN, Callback function.

I. INTRODUCTION

In recent times, with exponential development in technology with machine learning, it has become a part of our day to day life with high predicting capability. We are using machine learning algorithms in areas like health care, education system, financial markets, movie recommendations, criminal cases and many more . For evaluating our work we are using Deep learning models, it can be broadly classified into two category: Supervised Learning and Unsupervised Learning . Supervised Learning: In Supervised learning where we have a input data set (X) and output data (Y) then we have to use an algorithm for mapping from input to output function by identifying the correct pattern [1]. Examples: Artificial Neural Network, Convolutional Neural Network, Recurrent Neural Network etc. We can classify the supervised learning into two categories: Regression Problem, Classification Problem. And In unsupervised learning we have input data set (X) but we do not have any idea about output data set. Our algorithm itself have to find the structure and distribution. Again we can also classify the unsupervised learning into two categories: Clustering problem and Association problem. Examples: Self-Organizing Maps, Deep Boltzmann Machines and Autoencoders. For building and evaluating our work we are using ANN architecture with Keras Application Program Interface (API). We have chosen Keras for our architecture because it is more easy to use compare to other libraries and it supports both Tensorflow [2]and Theano backends [3][4]. And in our case we are using Tensorflow backend. ANN architecture working start from initializing some random small weights close to 0. And after every input from dataset it will do forward-propagation to get the predicted output value. After getting the predicted output value it will compare with the actual output value and it will measure the error [5][6][7]. And for the generated error it will do back-propagation. The whole steps will be repeated number of epochs times. The main problem have been observed in present architecture is Bias-Variance tradeoff . Our main aim in the proposed architecture to achieve the accurate result with Low-Bias and Low-Variance.

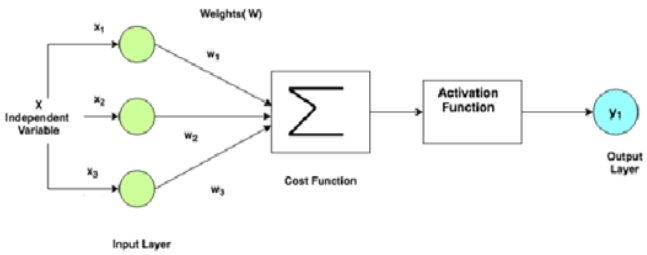


Fig.1. Neural Network work flow

A. Cost Function

Cost function is used for minimizing the difference between actual output and predicted output value, in other words we can say it's an error minimizing technique. It can be apply on entire set of training or mini batch wise.

Revised Manuscript Received on July 06, 2019.
Sanghita Saha, SCOPE, VIT, Vellore, India.
Ramanathan.L, SCOPE, VIT, Vellore, India.

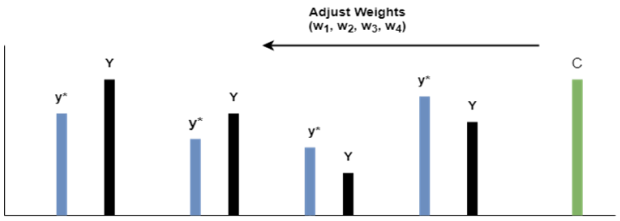
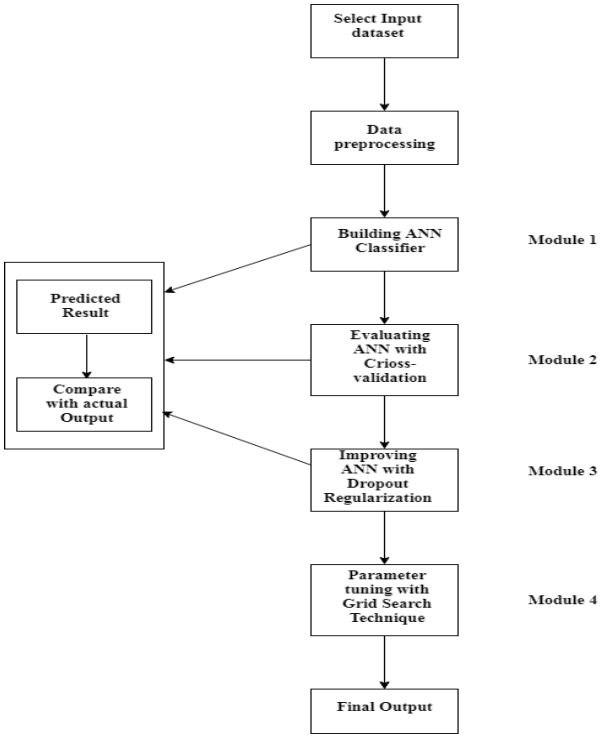


Fig.2. Cost function adjusting weights

B. Activation Function

It is an important feature for neural network it decides which neurons to activate or deactivate. It make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases [8][9]. Without activation function our model will become simple linear regression model where the model do not have the power or memory to decide in complex situation. There are various types of activation function present in neural network so based on the output requirement we can use it. And for neural network there exits several Optimization algorithms [10] for gradient Descent problem, for finding the minimum value of a function. Most commonly used in Neural Network Adam, RMSprop and Adadelta. The main objective of this paper is to improve the ANN architecture. In the first part of this paper, we have given a brief introduction about deep learning, ANN architecture, Keras, Tensorflow and other general information. In the second section, experimental work is described. The final section contains the results discussion and future work.

II. PROPOSED METHOD



I Fig.3. Building ANN Architecture with Parameter Tuning.

In this section we are going to explain about proposed architecture module working. The proposed architecture have been divided into four modules:

A. Data preprocessing:

This part is common and very important for all the machine learning model, it's a technique used for collecting good data based on our requirement which has a high impact on the training and test set. Based on the data set we have to do transformation and apply feature scaling for scaling the values in same range. This preprocessing step is the heart of every machine learning model.

B. Building ANN:

Here for building ANN we are using simple Holdout Method where we will divide the dataset into two different parts as training set data (80% in our experiment) and test set data (20%). For splitting dataset into two parts there is no hard and fast rule, if you have good dataset then we can take less than 80% also. The model take the training set as input and will give prediction on the test set which is completely unknown for the model. And this is the reason the test set data can suffer with high variance with different data set. The only advantage of this method is fast performance. This part is optional, we can use this part when time constraint is present.Another way of building ANN is using Callbacks Function in training set, Callbacks is a function sets and a very advance library feature in Deep learning models and can be implemented easily with Keras. This technique comes with the features like visualization of training and test set data, preventing overfitting in models [10]. Some important features that comes with the callbacks functions are: BaseLogger and History, these two feature is automatically applied to every Keras model; EarlyStopping, it can used for preventing overfitting in training set; LearningRateScheduler, it takes an each epoch as input and present learning rate and returns a new learning rate as output (float); ReduceLROnPlateau, it Reduce learning rate when accuracy has stopped improving by a factor of 2-10; ModelCheckpoint, it is used for save the model after each epoch; TensorBoard, it is used for visualize dynamic graph of our training and test set data. And at a time we can use more than one callbacks function for monitoring and improving the model. By using this feature in our model it does not create any major difference in terms of accuracy, but if we see in terms of time so it's saves a lot of time and it is easy for debugging purpose.

C. Evaluating ANN:

This part comes with the advantage of previous module, here in this section we are applying K-Fold cross validation method. It is an improved version on Holdout method, here the whole data set will be divided into k subsets and the holdout method will be repeated each time with new K subsets. And each time a new set will take as test data. The disadvantage of this method is running time is more.

D. Dropout Regularization:

It is technique used to prevent Overfitting in our model. It applied to the neurons so that some of neuron randomly become disabled at each iteration during training, we use this method if we have high variance in the result. The Dropout range should be in between (0.1 to 0.5) otherwise underfitting will occur in model. It can be apply on one layer or several layers of the network.

E. Parameter Tuning:

This is the final part of our architecture in this module we are going to choose the best parameter for our model among many. For parameter tuning there is no fixed rule. This part help in examine the final result from different combination of Hyperparameter like no. of epochs, batch size optimization algorithm .

III. EXPERIMENTS AND RESULTS

A. *Datasets*

For more accurate evaluation of the datasets, we have test the model on reputed datasets from Kaggle [16] and UCI Machine Learning repository [17]. For any model building it is advisable to use only reputed website datasets because without good data we cannot expect good prediction result from training set or test set. During our experiment time we have observed that the model is time dependent meaning if we feed good data first to the model it will show high accuracy but in your data there is no correlation among them it will give us low accuracy. For our experiment purpose we are using classification datasets only.

B. *Results*

Table 1: ANN Test Accuracy Results

Dataset Name	Epochs	Optimization Algorithm	Activation Function		Accuracy (%)
			Input Layer	Output layer	
Employee-Details	100	Adam	Relu	Sigmoid	89.45 %
Cleveland Heart Disease	100	RMSprop	Relu	Sigmoid	89.88 %
Wisconsin Cancer	150	RMSprop	Relu	Sigmoid	99.57 %
Banknote Authentication	100	Adam	Relu	Sigmoid	99.27 %
Wine class	150	Adam	Relu	Softmax	98.66 %
Iris Flower	150	RMSprop	Relu	Softmax	97.66 %

The above tables is the output of Method1: ANN Building Architecture using tool Spyder(Python 3.6) [11] . We have validate our model based on different parameter shown in table. In our experiment, classification task was carried out on the dataset with Keras API and Tensorflow backend [12][13]. Different activation functions were experimented to test the performance of the model. We obtain the best accuracy by Relu for input layers, Sigmoid for binary output and Softmax layer for categorical output [14][15]. And as an Optimization Algorithm we can use Adam and RMSprop in neural network and we have checked with other possible combination but we got the best result with Adam and RMSprop. For all the datasets we have taken batch size parameter as 100 and hidden layer 5 as constant value. And we are able to remove randomness from the model by using seed function which has been created by initializing random weights to the model.

C. *Graphs*

With the help of graph we can measure our model in terms of overfitting and underfitting. And we can see that in our models there is no overfitting.

1. Employee Details Dataset:

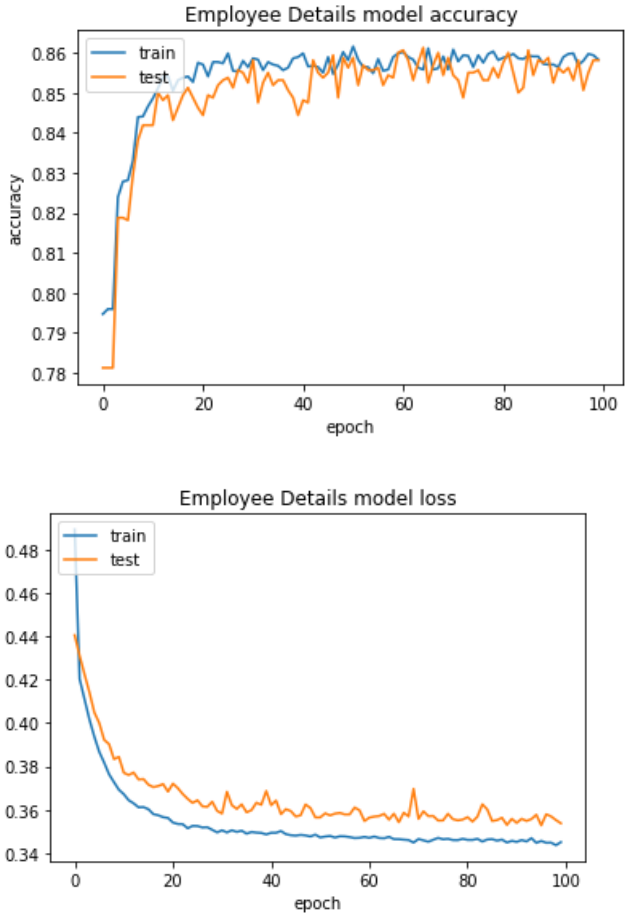


Fig.4. Employee accuracy and loss details

2. Cleveland Heart Disease Dataset [16]:

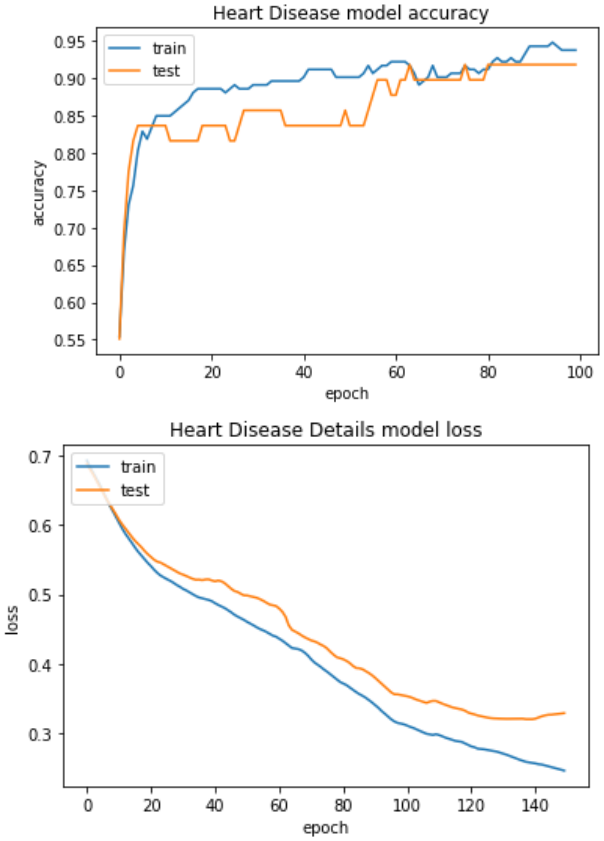


Fig.5. Cleveland Heart Disease accuracy and loss details

3. Wisconsin Cancer [17]:

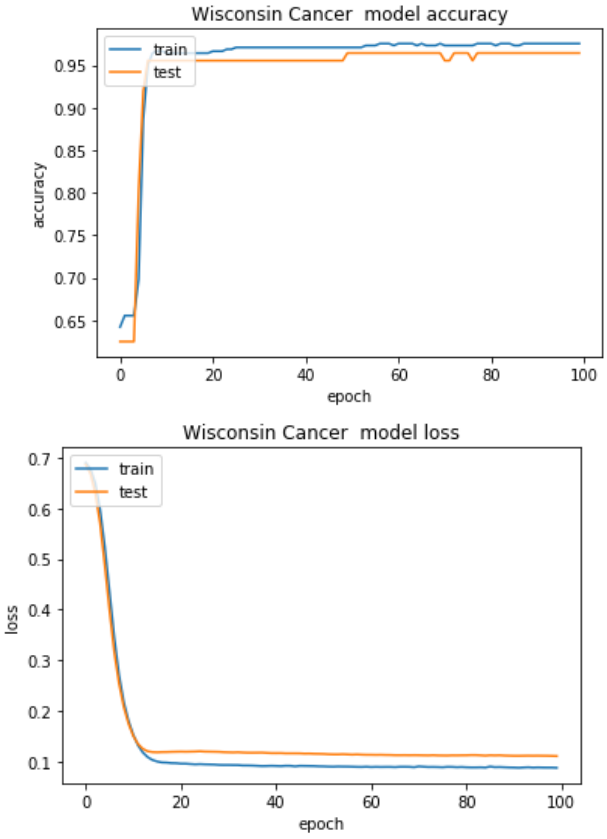


Fig.6. Wisconsin Cancer accuracy and loss details

4. Banknote Authentication [19]:

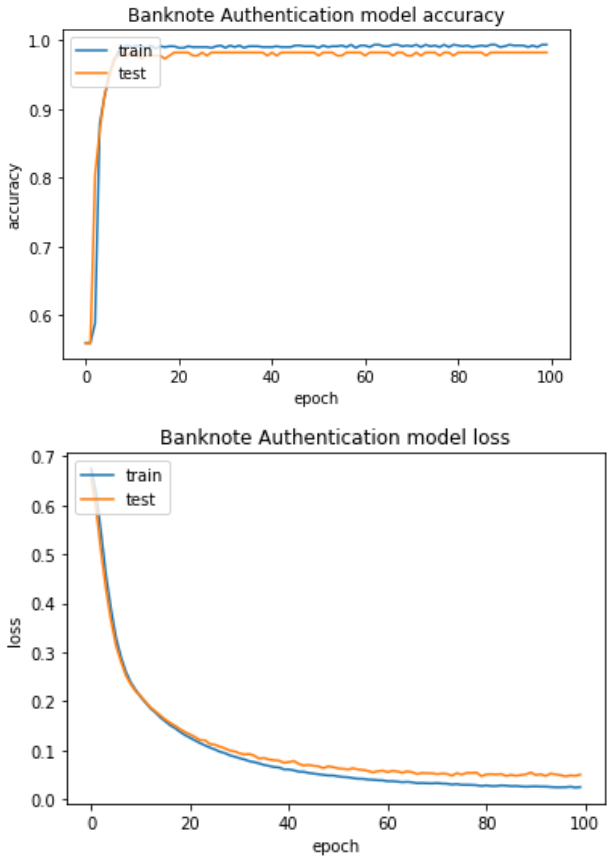
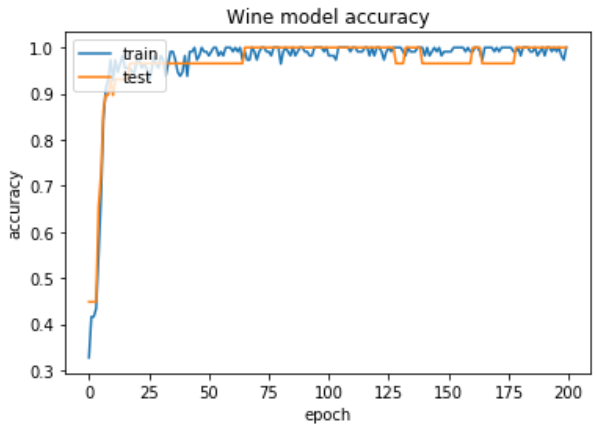


Fig.7 Banknote Authentication accuracy and loss details

6. Wine Class [20]:



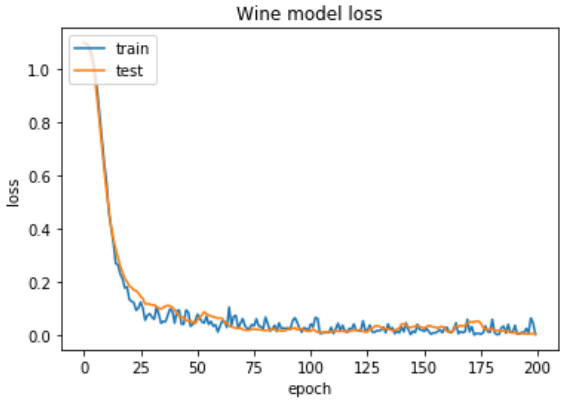


Fig.8. Wine class classification's accuracy and loss details

7. Iris Flower [21]:

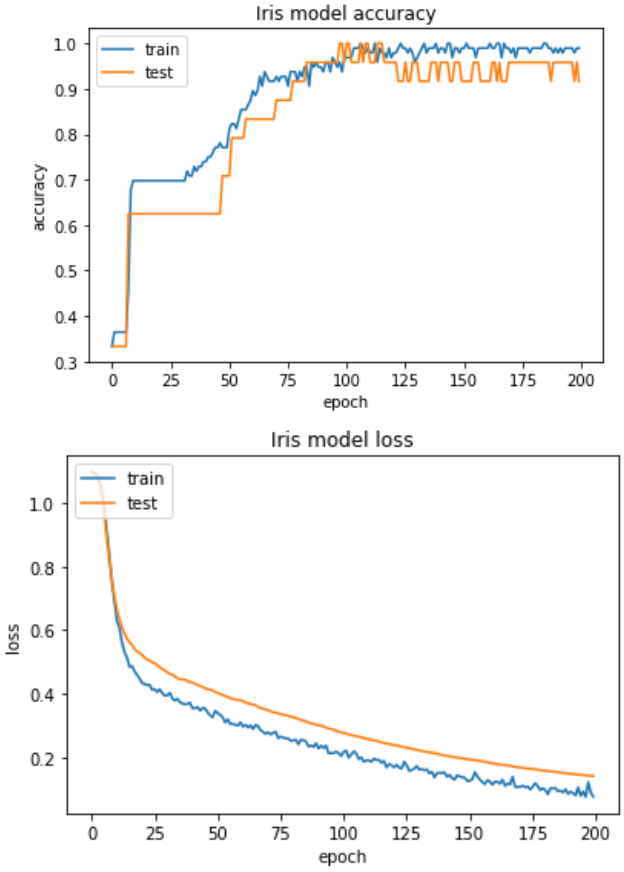


Fig.9. Iris Flower classification's accuracy and loss details

IV. CONCLUSION AND FUTURE WORK

Deep learning approaches are increasing effectively every day. Deep learning contribute fast and efficient solutions in data analysis. In this paper we are proposing a generic architecture using ANN model which is not specific to any problem means any type of classification problem we can feed to our model and it is implemented by using Keras API and Tensorflow backend which is giving higher accuracy compare to other models [16]. The models have been experimented with several good reputed dataset for measuring the performance. The result using our proposed architecture has been compared with other architecture with CPU setting and it is found that our proposed architecture accuracy is high and memory requirement is less. And from our experiments we suggest that activation function for input layer Relu should be used and we can use Sigmoid or Softmax activation function in output layer for binary and categorical output .

For future improvement of this project can be done with GPU setting for measuring the performance compare to CPU and can further be applied on Regression, Clustering, Causal-Discovery types of problems .

V. REFERENCES

1. M. A. Nielsen, "Neural Networks and Deep Learning," Mach. Learn., pp. 875–936, 2015.
2. L. Rampasek and A. Goldenberg, "TensorFlow: Biology's Gateway to Deep Learning?," Cell Syst., vol.2, no. 1, pp. 12–14, 2016.
3. J. Schmidhuber, "Deep learning in neural networks : An overview," Neural Networks, vol. 61, pp. 85–117, 2015.
4. P. P. Groumpos, "Deep Learning vs. Wise Learning: A Critical and Challenging Overview," IFAC-PapersOnLine, vol. 49, no. 29, pp. 180–189, 2016.
5. A. Gulli, Title Page Deep Learning with Keras Implement neural networks with Keras on Theano and TensorFlow.
6. I. K. Sandhu, M. Nair, H. Shukla, and S. S. Sandhu, "R eview A rticle ARTIFICIAL NEURAL NETWORK :," vol. 5, no. 3, 2015.
7. W. Wang, Y. Gao, and A. Z. Jin, "Interpretive Reservoir: A Preliminary Study on the Association between Artificial Neural Network and Biological Neural Network," Proc. Int. Jt. Conf. Neural Networks, vol. 2018–July, pp. 1–8, 2018.
8. E. Gelenbe and Y. Yin, "Deep learning with dense random neural networks," Adv. Intell. Syst. Comput., vol. 659, pp. 3–18, 2018.
9. B. Karlik, "Performance analysis of various activation functions in generalized MLP architectures of neural networks," Int. J. Artif. Intell. Expert Syst., vol. 1, no. 4, pp. 111–122, 2015.
10. P. Li, "Optimization Algorithms for Deep Learning."
11. P. Sherkhane, "Survey of deep learning software tools - IEEE Conference Publication," pp. 236–238, 2017.
12. F. Ertam, "Data classification with deep learning using tensorflow," 2nd Int. Conf. Comput. Sci. Eng. UBMK 2017, pp. 755–758, 2017.
13. J. L. Berral-Garcia, "When and How to Apply Statistics, Machine Learning and Deep Learning Techniques," 2018 20th Int. Conf. Transparent Opt. Networks, pp. 1–4, 2018.
14. J. Wang and D. Pei, "Kernel-based deep learning for intelligent data analysis," 1st Int. Conf. Electron. Instrum. Inf. Syst. EIIS 2017, vol. 2018–Janua, pp. 1–5, 2018.
15. K. G. Kim, "Book Review: Deep Learning," Healthc. Inform. Res., vol. 22, no. 4, p. 351, 2016.
16. https://archive.ics.uci.edu/ml/datasets/heart+Disease
17. https://www.kaggle.com/thebrownviking20/intro-to-keras-with-breast-cancer-data-ann
18. K. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karamouzis, and D. I. Fotiadis, "Machine learning applications in cancer prognosis and prediction," Comput. Struct. Biotechnol. J., vol. 13, pp. 8–17, 2015.
19. https://archive.ics.uci.edu/ml/datasets/banknote+authentication
20. https://archive.ics.uci.edu/ml/datasets/wine
21. https://archive.ics.uci.edu/ml/datasets/

AUTHORS PROFILE

Sanghita Saha she completed her B.E in computer science and engineering from Jorhat engineering college which is in Jorhat, Assam, India and she has completed her Mtech in Computer Science specialization in Information security from Vellore institute of technology, Vellore, India. She is currently doing job in Aptiv (Automotive Industry) Bangalore and having experience of one year. Her research interests are Information security, machine intelligence, Embedded. She currently working in technologies like Vehicle to Everything(V2X), Over the Air Services(OTA) and Autonomous Driving. Her ongoing research are on Malware Analysis using machine learning algorithms,

GPS point calculation using Recurrent Neural Network(LSTM) .

Dr. Ramanathan L has received his B.E. in Computer Science & Engineering from Bharathidasan University, Tiruchirappalli, India, and M.E in Computer Science from Satyabhama, Chennai, India, the Ph.D. degree in Computer Science and Engineering from VIT University, Vellore, India. He is currently an Assistant Professor (Selection Grade) in VIT, Vellore, India. His area of interest is Data Mining, Internet of Things, Image processing, Database systems, Software Engineering, Cloud Computing, and Virtualization. He is having 14+ years of teaching experiences. He has published more papers in International Journals and Conferences. He is an Editorial board member/reviewer of International/ National Journals and Conferences. His ongoing research is on Prediction, Classification, and Clustering for Educational systems. He is a member of IACSIT, CSI, ACM, IACSIT, IEEE (WIE), and ACEEE