International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Identification of Fast Radio Bursts using Transfer Learning Approach with Data Augmentation

Vandana Jagtap[1,2] (0000-0002-9857-0727), Rakesh K Yadav[1]
[1.]Shri Venkateswara University, Gajraula, Amroha, Uttar Pradesh, India
[2.] Dr. Vishwanath Karad world peace University, Pune 411038, India
*Corresponding author. E-mail: vandana.jagtap@mitwpu.edu.in,
er.rakeshyadava@gmail.com

Abstract: The universe has many mysteries, such as pulsars, dying stars, supernovae, and fast radio bursts (FRBs), FRBs are millisecond long radio signals, detected as a spike in radio-telescope data. Identification of Fast Radio Bursts from available data involves manual inspection of exhaustive data/plots. Radio Frequency Interference in pose a major challenge in identification of Fast Radio Bursts due to their abundance in the observatory data. We present a machine-learning-aided system, which screens telescope-generated data and identifies potential Fast Radio Burst candidates in it. Proposed system employs Convolutional Neural Networks and Transfer Learning to classify potential Fast Radio Bursts from Radio Frequency Interference from data recorded by the uGMRT. We have used data simulation tools to synthesize additional samples in order to make up for the paucity of data. The VGG16-based model displayed the best receiver operating characteristics curve with the area under curve being 0.90 along with an accuracy of 90.67%.

Keywords: Fast Radio Burst, Deep Learning, Convolutional Neural Network, Machine Learning, Radio Frequency Interference, Transfer Learning, Computer Vision.

I. Introduction

FRBs are bright pulses of emission at radio frequencies (50 mJy to 100 mJy) with duration in the order of milliseconds or less. They are very energetic at their source but by the time they reach the earth's atmosphere, the strength of the signal reaching the Earth can be viewed as a thousand times less than from a mobile phone on the moon [1,2]. The origin have not been identified yet, with suggestions spanning from a rapidly rotating neutron star and a black hole to extra-terrestrial intelligence [1]. Fast radio bursts demonstrate the characteristics dispersion sweep of radio pulsars. Lorimer et al. identified the first FRB in 2007, but it was actually observed some six years earlier, from the archival data of a pulsar survey of the Magellanic Clouds. It was named as the "Lorimer Burst" [3]. Identification of fast radio bursts requires sophisticated methods to transform the raw data recorded by telescopes into graphs and plots (images). Since it is telescopic data monitoring the data becomes exhaustively large. In absence of automated systems, identification of FRBs requires manual inspection of these graphs and plots for the ever-increasing data. The process becomes more complicated due to the high incidence rate of radio frequency interference, which deceives the telescope into recording incorrect data, at the telescope premises.

A wide variety of computer systems have been developed for screening such data and identifying FRBs in them [4,5,6,7,8,9]. Most of the existing systems rely on machine learning at some stage in their processing. However, from the numerous available FRB-detection systems, not more work is available to cater specifically to the Giant Metrewave Radio Telescope (GMRT) and identified FRBs in the recorded data by the GMRT. The operating frequencies of the GMRT are comparatively lower than those at which other telescopes which have been able to detect FRBs operate. The lower frequencies mean that the telescope is more susceptible to noise[10,11,12]. Research on FRBs is largely dependent on the part of sky that a telescope in question observes. A robust system which is designed for the GMRT is therefore required to allow astrophysicists discover FRBs which will accelerate the research being carried out in the domain.

Proposed system employs systematic pre-processing to transform the telescope data into meaningful plots. The convolutional neural networks and transfer learning have been used to build image classifiers. The data recorded by the GMRT has been used to train and test the classifiers. Data simulation has been used to generate additional samples of FRB and RFI to compensate for the limited amount of data that we had. The classifiers were fine-tuned to extract the best performance.

In this research article, our objective is to compare transfer-learning models in the context of classifying fast radio bursts (FRB) and radio frequency interference (RFI). Additionally, we elucidate the process of generating simulated FRBs and conducting parameter estimation analysis. Subsequently, we assess the performance of four

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

models (VGG16, InceptionV3, Xception, and DenseNet121) by comparing their accuracy in estimating parameters.

This article is structured as follows: Section 1 provides an explanation of domain-specific terminology used throughout the paper. In Section 2, we introduce the methodologies and system design, focusing on the design of models for classification. Section 3 addresses implementation and data presentation. Section 4 presents and discusses the results obtained from the transfer learning models. Finally, Section 5 contains our conclusions.

Domain terminology

Several domain-specific terminology has been used throughout the paper. While explaining each terminology detail is beyond the scope of this paper, some fundamental terms have been explained below strictly within the scope of this paper.

Radio wave: A form of electromagnetic radiation having frequencies in the radio spectrum (3 KHz to 300 GHz).

Frequency channel: A radio telescope detects radio waves having frequencies in particular ranges. Such a range is called a frequency band. A frequency band is further divided into smaller portions called channels. A radio telescope measures energy of a signal observed in a particular band across all its channels.

Signal-to-noise ratio (SNR): It is a measure of how strong the desired signal is with respect to unwanted signal or noise.

Dispersion measure (DM): The electromagnetic signals detected by a radio telescope consists of different frequencies. The propagation of these signals is delayed due to the presence of charged particles in space. Lower frequencies are more susceptible to such delays than higher frequencies. The degree of effect due to this phenomenon on a signal is represented using a metric called dispersion measure [13, 14, 15].

II. Methodologies and system design

The data in very first step is stored as a ‘filterbank’ file. A filterbank files stores raw binary data pertaining to the energy levels of a signal detected by a radio telescope across several frequency channels. By consulting domain experts, a list of interesting candidates which were recorded in the filterbank file was obtained. An interesting candidate can either be a fast radio burst or an instance of radio frequency interference. The tool Candmaker, which is a part of the FRB-detection system FETCH [9, 24, 25, 26], was used to extract each interesting candidate from the filterbank file. Each candidate is extracted in its own HDF5 file. The HDF5 file format is used to store multidimensional data in the key-value format. One key is dedicated for one attribute of data and the corresponding value can be a single value or a collection of values. Candmaker generates DM-vs-time, frequency-vs-time and flux-vs-time plots for each candidate. Additionally, it determines numeric attributes such as dispersion measure, SNR, pulse width and other telescope-related parameters for each candidate.

Machine learning techniques are useful to identify hidden information in data and determine complex patterns. Using such techniques, the mapping function between a certain input and output can be determined. Numeric parameters such as DM, SNR, flux, etc. can be used to train a machine-learning model so that it can learn the rules for those parameters which can help in classification. Techniques such as decision trees, random forest classifiers, etc. can be used to build a classifier. The main challenge with these methods is that the above mentioned numeric parameters don’t assume a precise range for FRBs. Hence, classification based only on these parameters might not be enough to determine the most distinguishing patterns in data. The plots for candidates, however, exhibit a sufficiently credible pattern and can be used to classify candidates. Fig. 1 shows the DM-vs-time plots for a FRB and RFI generated using Candmaker. While there are exceptions, majority of FRB plots exhibit the ‘bow tie’ pattern at the Centre as seen in Fig. 1. RFI plots do not exhibit any particular pattern. Image classifiers which accept these plots as input therefore exhibit potential in identifying fast radio bursts from radio frequency interference.

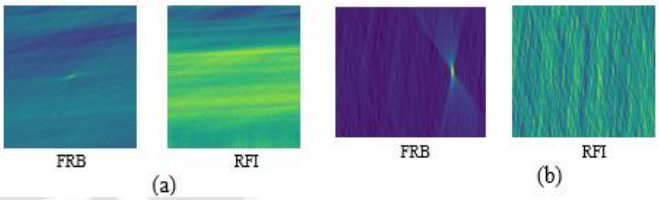

Figure 1. a. Sample images for FRB and RFI Real candidates (Dimensions: 256 x 256 x 3), b. Sample images for FRB and RFI Real candidates (Dimensions: 50 x 250 x 3)

Convolutional neural networks Neural networks are massively used to perform classification owing to their ability to train over thousands of ‘neurons’, arranged in layers, which learn different characteristics of data by figuring appropriate ’weights’ for the characteristics. Neural networks train over labelled data, i.e., they need to be told what the expected output is for every sample they train on. In case the neural network produces an incorrect output, it calculates the difference between the expected and the produced outputs to determine the error and updates its weights accordingly.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
Article Received: 05 July 2023 Revised: 25 August 2023 Accepted: 15 September 2023

The DM-time plots are treated as images and classification are performed using these images. If the size of an image is expressed as (width, height, channels), the total number of features present in every single image is: width height channels. As the size of input images rises, the aggregate number of features to be processed upsurges, which makes the classifier highly complex and computationally expensive.

Convolutional neural networks (CNNs) are extremely popular in image based cataloguing tasks. CNNs can handle high-dimensional image-data well, owing to the operation of 'convolution'. The weights learned by a CNN are arranged in 2D or 3D matrices called 'kernels'. A CNN has many convolutional layers wherein each layer can have multiple kernels. CNNs learn the features in an image by understanding the position-based relationships between pixels. They can also recognize features irrespective of their position in the image [16, 17,18]. These properties, coupled with the ability to handle feature rich data, make CNNs suitable to solve the problem of FRB classification.

Transfer learning Due to the low rate of FRB detection, the number of samples for the FRB class is very small. Neural networks rely on large data sets to train properly and to be able to classify unseen data with high confidence. Given the paucity of data, it was not feasible to build a neural network, which could exhibit high accuracy in classification [19, 20]. To combat this, the technique of transfer learning is used.

In the field of automation, we have multiple domains under the umbrella of Artificial Intelligence. Transfer learning (TL) is one of the machine-learning approach. In transfer learning, the neural network trained to solve a specific problem is reused to solve another problem. The previously learned weights by the neural network to be reused are relearned. Transfer learning is based on the principle that the initial layers of a neural network learn primitive features seen in input samples which do not represent problem-specific information [21,22,23]. For instance, the initial layers of a CNN trained on a large image set are capable of identifying basic features such as lines, curves, shapes, etc. The concluding layer of the previously trained model, which is responsible for classification, is replaced by new layers so that the model can perform classification for the new problem.

To harness the power of transfer learning, we need to decide which layers of the previously trained model are to be frozen and which ones should be re-trained. This decision depends on the degree of similarity between the original problem and the new problem. If the two problems exhibit high resemblance, almost all the layers of the original model can be reused without training. Otherwise, only the initial layers would be frozen and the remaining layers are qualified along with the newly added layers. The potential of transfer learning in classification of fast radio bursts can be seen in the work done by Agarwal et al.2019 [9].

The pre-trained models VGG16, InceptionV3, Xception, DenseNet121 were trained on the ImageNet data set were used to implement transfer learning.

Since the ImageNet data set contains images belonging to hundreds of classes, only the initial layers of these models were frozen and the last layers were trained so as to reuse only those weights that are responsible for identification of basic features.

2.1. System design

Fig. 2 captures the functioning of the system. The GMRT stores radio-wave time-series data in filterbank files. The list of interesting candidates generated by the single-pulse-search program HEIMDALL[29] is used to extract data representing interesting events. The tool Candmaker is used to plot DM- time, frequency-time and flux-time plots per interesting candidate. The classification is done into radio frequency interference (class 0) and fast radio burst (class 1) using these plots. A major issue posing a challenge to this system is the imbalance between class samples. Since FRB detection is in nascent stages, the number of FRB candidates is considerably less. This will coax the classifier to get biased towards class 0. To overcome this short coming, RFI and FRB candidates are simulated using available tools. The candidates are simulated according to the specifications of the GMRT so that they bear resemblance with the real candidates. The extracted real candidates and simulated candidates together form the image-set to be used to build the classifier. The real and simulated candidates are transformed to make them have identical specification. Appropriate pre-processing techniques are implemented to enhance the performance of the classifier. The processed image-set is then split into testing and training sets. The classifier is trained on the training set. The model is fine-tuned by determining the optimal values for various hyper-parameters. Once the classifier delivers acceptable performance in the training phase, the testing images are fed to it to have them labelled. The performance of the classifier is evaluated by considering yardsticks such as accuracy, F1-score, area under the receiver operating characteristics curve, etc. The classifier outputs a comma-separated-value (CSV) file which contains the name of an image and the label assigned to it. A chart showing misclassified samples is also prepared to provide further scope to the ongoing research.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

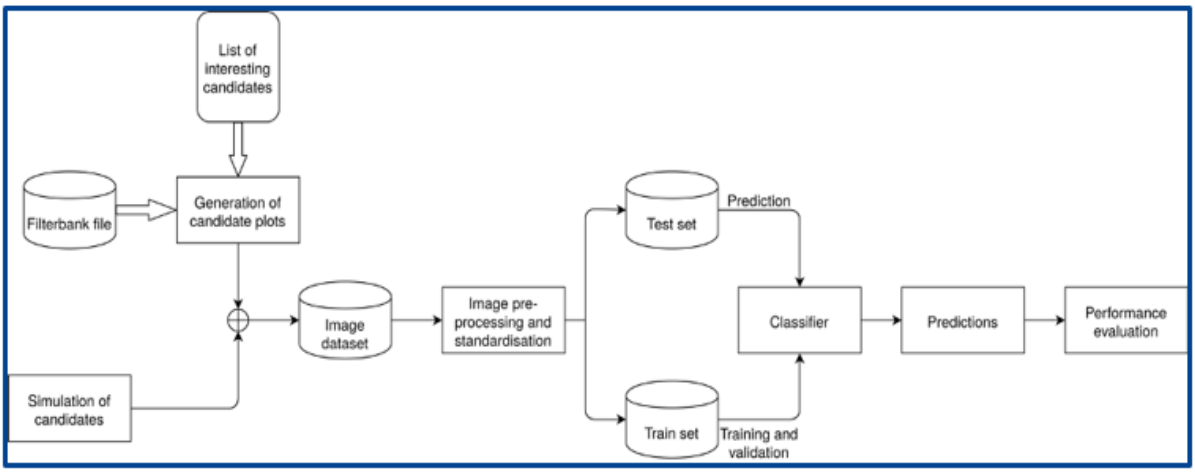

Figure 2. System architecture

III. Implementation

Data preparation

Extracting real candidates The DM vs. time and frequency vs. time plots for each interesting candidate were obtained from the filterbank file by using Candmaker. The plots were stored in an HDF5 (Hierarchical Data Format version 5) file per candidate.

Simulation of candidates to train a neural network properly, a sufficiently large data set must be used. However, the data set used had a mere 1274 samples. Additional frequency-time and DM-time images of fast radio bursts and RFI were generated through simulation. The package used to perform simulation was single_pulse_ml by Dr. Liam Connor [8] This package has the facility to simulate images based on the specifications of the concerned telescope. The biggest contributing factor to these images is the number of frequency and time channels as, they affect the resolution of the generated images. These numbers must be set so that a proper balance between simulation performance and simulation accuracy is obtained. The simulation script also provides the facility to adjust other parameters such as DM or the number of generated images. The number of frequency channels as well as the time resolution in the script were changed to match those of the GMRT.

After using the simulation script, the required data were generated as an HDF5 file. The actual images were stored as 0-1 normalized NumPy arrays. Thus, the DM-vs-time and frequency-vs-time plots for all simulated candidates were obtained as a single HDF5 file.

Transformation of candidate plots into images

Only the DM-time plots were used to perform classification since they exhibited features which were sufficiently unique for both the classes. The time-series in-formation was read from the HDF5 files using h5py and the DM-time information was transformed into plots using matplotlib. The plots had data pertaining to a single image-channel. Since all the pre-trained models which we used required 3-channel images as input, we converted all the plots to 3-channel PNG (Portable Network Graphics) images. During initial testing, it was found that the simulated FRB-like candidates were misleading the classifier and affected the class-1 recognition rate. Hence, only simulated RFI candidates were used for building models.

Data pre-processing

Blurring simulated candidates, the simulated- RFI images were noisier than the real-RFI images. To counter this, the Gaussian Blur filter was applied to each simulated RFI-image so that both the real and simulated-RFI images had similar texture.

Resizing images

The dimensions of the real and simulated candidates were as follows:

Real candidates: (256 ×256 ×3), Simulated candidates: (50 ×250 ×3), All images were resized to (10×0 100×3) since all samples given to a CNN must have the same dimension. This also resulted in each pixel-value being scaled to [0, 1].

Normalization of pixels The pixel-values of each image were transformed so that they had a mean of zero with unit standard deviation. This led to the pixel- values being centered. The images used to train ImageNet were centered as well. This operation ensured that the distribution of pixel values was compatible with the distribution seen in images on which the original models were trained.

Flipping The RFI images were greater in number than the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
Article Received: 05 July 2023 *Revised:* 25 August 2023 *Accepted:* 15 September 2023

real-FRB images. The classifier would therefore learn the features of RFI better and get biased toward class 0. To handle this, the class-1 images were flipped about the X- and Y- axes and the new image was added to the training data set. Flipping the images introduced variety to the training set and increased the support of class-1 samples as well.

Building the model

We used the Keras framework to construct our neural network models. The following pre-trained models were re-purposed: VGG16, InceptionV3, Xception and DenseNet121. A high-level view of the architecture of the classifier is shown in Fig. 3. The weights of these models for the ImageNet data set were downloaded and the models were set up. The topmost layer, which is responsible for classification, was removed from each of these models. At the end of the last layer, the feature map obtained was flattened and passed on as input to a fully connected layer having 48 neurons with ReLu activation. Finally, all activation values were fed to a 2-neuron fully connected layer to perform classification.

Batch normalization

When the input to a neural network propagated through all its layers, the variance amongst the expected and the actual outcome is calculated as the error. This deference, we call it error is then corrected by updating the weights in all the previous layers. When a layer in a neural network gets trained, it learns weights according to the distribution observed in its input which it receives from the previous layer. Due to error propagation, the weights of layers get updated, which may result in a change in the distribution in input for some layers (Ioffe and Szegedy, 2015). Thus, the neural net-work may never stabilize. To counter this effect, the 'Batch Normalization' layer provided by Keras was used. This layer normalizes per-batch activation scores of the previous layers so that their mean is approximately 0 and their standard deviation is approximately one. This ensures that the weights learned by the model stabilize faster and hence the number of epochs required is reduced.

Dropout layer

Since the sample data set was not huge in size, there were a high possibility of the neural network trying to over fit the training data. To avoid this, a dropout layer was introduced after the 48-neuron dense layer. Some input activation scores was set by dropout layer randomly sets to a layer to zero. The weights corresponding to such activation scores are not updated during error propagation. This results in the neural network treating the same layer differently in each epoch since random links are ignored in each epoch. The dropout rate was set to 20% to randomly ignore 20% of the activation scores of the 48-neuron layer.

Obtaining class labels

The last fully connected layer produced output as sequence of logits (activation) values, one per class. These values do not follow a standard scale and hence must be standardized. The Softmax function transforms these logits values into corresponding probabilities such that the sum of the probabilities is 1. The available sample belongs to the class with the highest probability. The Softmax function is given as

$$\textbf{Softmax } \vec{x}_i = \frac{e^{x_i}}{\Sigma_{j=1}^{k}\ e^{x_j}}$$

$\vec{x}_i$= Input_Vector

e^{xi}= standard_Exponential_Function for Input_Vector

K = Number_of_Classes in the Multi-Class_Classifier

e^{xj} = Standard_Exponential function for Output_Vector

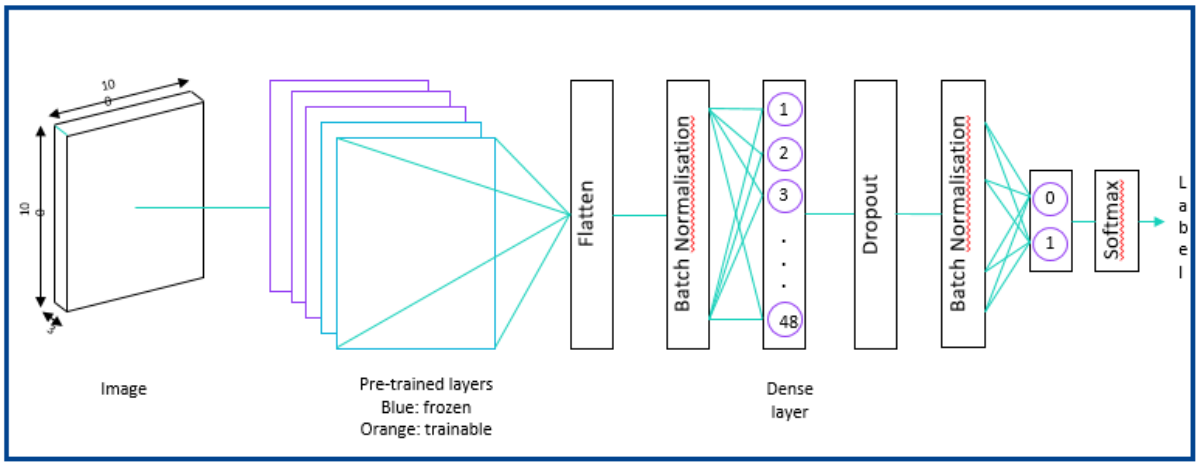

Figure 3. Structure of the classifier

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Error propagation and optimization

Minimization of an objective function is achieved by gradient descent, and based on its parameters. The parameters of neural network is updated by using gradient descent. By updating the parameters of the model in the direction opposite to that of its gradients is the method employed. In this way, we move downwards towards Minima. Batch gradient descent, Stochastic gradient descent, Mini-batch gradient descent are based on exactly when/at what intervals the parameter updates are performed these are various variants of gradient descent as.

For both gradient descent, an update is performed after calculating gradients according to the entire data set. which leads to slower convergence. Opposite to that, in stochastic gradient descent, updates are performed based on gradients from a single sample. While definitely faster, stochastic gradient descent causes significantly higher variance, and may result in convergence to a local, not a global minimum. Thus, we used mini-batch gradient descent, which takes the best of both worlds. Due to the used of mini-batch gradient descent, a batch size is determined and an update takes place per batch. This reduces the excessive variance of stochastic gradient descent, while retaining the advantage of high throughput and faster convergence. The weight update for mini-batch gradient descent

$$w_{ij} = w_{ij} - \alpha * \frac{\partial\left(\frac{1}{b}\right) - L(wij)}{\partial wij}$$

The speed of convergence, however, depends on not only the type of gradient descent, but also the learning rate. Learning rate decides how much a neural network gets influenced by error in output while updating its weights to account for it. Too small a learning rate makes for a slow convergence, but if it is made too large, it can lead to fluctuations or even divergence. To set a suitable learning rate, optimizers are used. We use the AdaDelta optimizer [29,30].

IV. Results

The models based on VGG16, InceptionV3, Xception and DenseNet121 were fine-tuned by varying the number of frozen layers and other hyper-parameters and training the model. For analysing the performance of the models for different combinations of hyper-parameters, the details of each run were stored in a CSV file. The CSV records the information like Pre-trained model used, b- batch size, Wi j - the weights of the network, α- the learning rate, L(Wij)- the loss function.

- Number of trained layers from the pre-trained model
- Size of the dense layer which is not a part of the pre-trained model
- Dropout rate
- Learning rate
- Number of epochs
- Training accuracy
- Training loss
- Testing accuracy
- F1-score for test data

Through analysis of the CSV, the optimal number of layers which must be trained for each model was determined. For getting comparable results, the following parameter were kept constant for all models: Size of dense layer: 48, Number of epochs: 10, Learning rate: automatic (figured out by the Adadelta optimizer), Dropout rate: 20%.

Since the number of class-0 samples out number class-1 samples, we evaluated models based on accuracy and F1-score. In our case, the accuracy itself is not completely indicative of the model's performance. The F1-score gives us an idea about how well the classifier performs for both the classes.

Table 1 shows how each of the models performed against various metrics of evaluation. The 'Trained layers' column gives the number of layers starting from the last layer that were trained from the respective base model. The metrics accuracy, precision, recall and F1- score will now be discussed. In the context of these metrics, we define the following symbols:

True positives (TP): It is the total number of samples that are classified in the positive class (FRB, in our case) and which actually belong to it.

True negatives (TN): It is the total number of samples that are classified in the negative class (RFI, in our case) and which actually belong to it.

False positives (FP): The number of negative samples (RFI) which were classified as positive samples (FRB).

False negatives (FN): The number of positive samples (FRB) which were classified as negative samples (RFI).

Table 1. Comparative analysis of performance

Base Model	Trained Layers	Accuracy(%)	Precision	Recall	F1 Score
VGG16	6	90.97	0.79	0.89	0.84
InceptionV3	26	80.56	0.58	0.92	0.71
Xception	19	86.81	0.69	0.89	0.78
DenseNet121	21	93.06	0.97	0.76	0.85

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

Accuracy, or the recognition rate, is the proportion of samples that were correctly classified among all samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + F}$$

Precision is the proportion of the correctly labelled positive samples out of all positive-labelled samples. It is a measure of how relevant the result is.

$$Precision = \frac{TP}{TP + FP}$$

Recall is the proportion of positive samples that were correctly labelled.

$$Recall = \frac{TP}{TP + FN}$$

Accuracy must be as high as possible. However, accuracy doesn't guarantee good performance in recognition across all classes. For problems which suffer from class imbalance, such as ours, in which the difference in the number of samples for each class is large, accuracy by itself is not sufficient to evaluate performance. Precision and recall give a better idea of performance in such cases. Precision tells us how exact the results are while recall represents the proportion of valid results. Ideally, both should be as high as possible, but there is always a tradeoff between the two. Another metric called the F1-score combines both precision and recall by taking their harmonic mean

$$F1\text{–}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

F1-score collectively accounts for both precision and recall and should therefore be as high as possible. The performance of different models was therefore com- pared by consulting their F1-scores.

Fig. 4 shows the receiver operating characteristics curves for all models along with the area under each curve. The ROC curve represents the true-positive rate (the proportion of correctly labelled positive samples) and the false-positive rate (the complement of true negative rate) for a classifier at various classification thresholds. ROC indicates the ability of a model to distinguish between various classes. It therefore gives an idea of the performance of a model across all classes. The area under the ROC curve represents how good a model is. The closer the 'knee' in the ROC curve to the top-left corner, more is the area under the curve. The VGG16-based model had the highest area-under-curve and exhibited the best balance between recognition of both the classes.

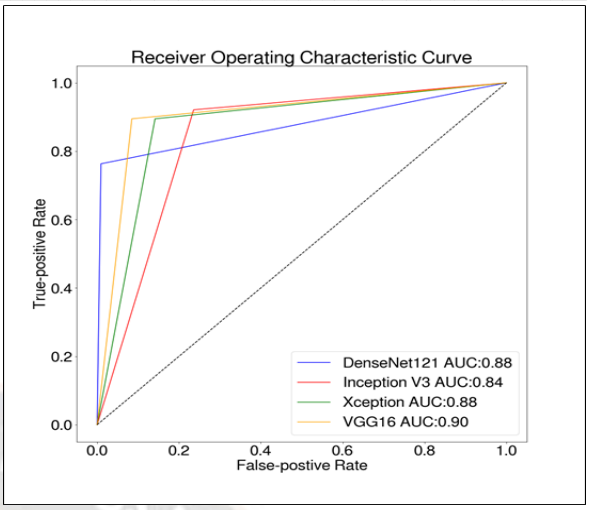

Figure 4: The Receiver Operating Characteristic (ROC) curve for all models.

V. Conclusion and future work

The VGG16-based model was the most optimal model to solve the problem statement, since it had the best ROC curve. Also, the model exhibited good performance in recognizing samples of both classes. To some extent, false-positives were tolerable for our project due to the nascent stages in which FRB data exists, but the goal was to have as high sensitivity as possible. As we wish to propel research into the domain, it goes without saying, that the miss rate (false-negatives) should be low. Based on these factors, the VGG16 model is preferred over all the other models that we have built. Thus, the model implemented successfully with an accuracy of 90.97% to classify fast radio bursts and radio frequency interference (RFI). Transfer learning with the aforementioned architecture and transformations is the optimal path to pursue, considering the complexities of the feature space, sparsity and lack of availability of data sets and anomalies in detection and identification.

Simulation of data is computationally expensive. We had to compromise on the quality of simulated data and the number of simulated samples. A graphics processing unit (GPU) can perform complex and number-crunching operations much faster than a center processing unit (CPU) due to its enhanced hardware capabilities and sophisticated support for parallel computation. The tradeoff between simulation time and accuracy of simulated data can be overcome to an extent by using powerful GPU-aided hardware.

The model can be further enhanced by simulating data, which bears more resemblance with real data.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 9
DOI: https://doi.org/10.17762/ijritcc.v11i9.8111
***Article Received:** 05 July 2023 **Revised:** 25 August 2023 **Accepted:** 15 September 2023*

VI. Acknowledgements

We thank the National Center for Radio Astrophysics (NCRA) for providing us with the data recorded by the GMRT and allowing us to use their high performance computing servers.

References

[1] Petroff E. et al., Fast Radio Bursts, 2019, Astron Astrophys rev. 27-4
[2] D. R. Lorimer et al. A Bright Millisecond Radio Burst of Extragalactic Origin.Science318,777-780(2007)
[3] Lynch R. S., http:// pulsarsearchcollaboratory.com/wp-content/uploads/2016/01/PSC_search_ guide.pdf
[4] Wagstaff K.L. et al., A Machine Learning Classifier for Fast Radio Burst Detection at the VLBA 2016, The Astronomical Society of the Pacific, vol. 128, no. 966,
[5] The CHIME/FRB Collaboration, The CHIME Fast Radio Burst Project: System Overview 2018, The Astrophysical Journal, vol. 863, no. 1
[6] Zhang Y. G. et al., Fast Radio Burst 121102 Pulse Detection and Periodicity: A Machine Learning Approach ., 2018, The Astronomical Journal, vol. 866, no. 2.
[7] Pang D.et al. A novel single-pulse search approach to detection of dispersed radio pulses using clustering and supervised machine learning, 2018, Monthly Notices of the Royal Astronomical Society, 3302–3323, vol. 480, issue 3.
[8] Liam Connor et al, Applying Deep Learning to Fast Radio Burst Classification, The Astronomical Journal 2018 AJ 156 256
[9] Agarwal, Devansh et al., FETCH: A deep-learning based classifier for fast transient classification, Monthly Notices of the Royal Astronomical Society 497 (2020): 1661-1674.B.
[10] Bhattacharyya et al., The GMRT high resolution southern sky survey for pulsars and transients: survey description and initial discoveries, The astronomical journal, vol. 2016 ApJ 817 130
[11] J. W. T. Hessels et al. FRB 121102 Bursts Show Complex Time–Frequency Structure, 2019 ApJL 876 L23
[12] Pilia, Maura, The Low Frequency Perspective on Fast Radio Bursts, 2022, Universe 8, no. 1: 9.
[13] Niu, CH., Aggarwal, K., Li, D. et al. A repeating fast radio burst associated with a persistent radio source. Nature 606, 873–877 (2022)
[14] K M Rajwade, et al., Possible periodic activity in the repeating FRB 121102, Monthly Notices of the Royal Astronomical Society, Volume 495, Issue 4, July 2020, Pages 3551–3558.
[15] Bussons Gordo, J., Fernández Ruiz, M., Prieto Mateo, M. et al. Automatic Burst Detection in Solar Radio Spectrograms Using Deep Learning: deARCE Method. Sol Phys 298, 82 (2023).
[16] Aya Nabil Sayed, et al., Deep and transfer learning for building occupancy detection: A review and comparative analysis, Engineering Applications of Artificial Intelligence, Volume 115,2022
[17] F. Zhuang et al., A Comprehensive Survey on Transfer Learning, in Proceedings of the IEEE, vol. 109, no. 1, pp. 43-76, Jan. 2021.
[18] Hussain, Mahbub et al. ,A Study on CNN Transfer Learning for Image Classification, UK Workshop on Computational Intelligence, 2018.
[19] Lu, Ying., Transfer Learning for Image Classification, 2017.
[20] Sara Hosseinzadeh Kassani, at el., Deep transfer learning based model for colorectal cancer histopathology segmentation: A comparative study of deep pre-trained models, International Journal of Medical Informatics,Volume 159,2022.
[21] B. Bamne, et al., Transfer learning-based Object Detection by using Convolutional Neural Networks, 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2020, pp. 328-332 (conf)
[22] Ioffe, Sergey, and Christian Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, International conference on machine learning. pmlr, 2015.
[23] Sandeep Kadam, & T. Srinivasarao. (2023). ElitGA : Elitism Based Genetic Algorithm for Evaluation of Mutation Testing on Heterogeneous Dataset. International Journal of Intelligent Systems and Applications in Engineering, 11(4s), 509–516. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2720.
[24] D. L. Jones et al., "Big data challenges for large radio arrays," 2012 IEEE Aerospace Conference, Big Sky, MT, USA, 2012, pp. 1-6, doi: 10.1109/AERO.2012.6187090.
[25] C. Patel et al PALFA Single-pulse Pipeline: New Pulsars, Rotating Radio Transients, and a Candidate Fast Radio Burst, 2018 ApJ 869 181
[26] FETCH, https://github.com/devanshkv/fetch
[27] single pulse ml, https://github.com/ liamconnor/single_pulse_ml
[28] HEIMDALL,https://sourceforge.net/p/ heimdall-astro/wiki/Home/
[29] Billings Lee, 2013, Scientific American, https://www.scientificamerican.com/article/a-brilliant-flash-then-nothing-new-fast- radio-bursts-mystify-astronomers/
[30] https://towardsdatascience.com/a- comprehensive-hands-on-guide-to- transfer-learning-with-real-world- applications-in-deep-learning- 212bf3b2f27a.
[31] https://machinelearningmastery.com/how- to-use-transfer-learning-when- developing-convolutional-neural- network-models/
[32] Prof. C. Ranjeeth Kumar. (2020). Malware Detection Using Remedimorbus Application. International Journal of New Practices in Management and Engineering, 9(01), 08 - 15. https://doi.org/10.17762/ijnpme.v9i01.82.
[33] Sable, N.P., Rathod, V.U. (2023). Rethinking Blockchain and Machine Learning for Resource-Constrained WSN. In: Neustein, A., Mahalle, P.N., Joshi, P., Shinde, G.R. (eds) AI, IoT, Big Data and Cloud Computing for Industry 4.0. Signals and Communication Technology. Springer, Cham. https://doi.org/10.1007/978-3-031-29713-7_17.
[34] N. P. Sable, V. U. Rathod, R. Sable and G. R. Shinde, "The Secure E-Wallet Powered by Blockchain and Distributed Ledger Technology," 2022 IEEE Pune Section International Conference (PuneCon), Pune, India, 2022, pp. 1-5, doi: 10.1109/PuneCon55413.2022.10014893.
[35] V. U. Rathod and S. V. Gumaste, "Role of Routing Protocol in Mobile Ad-Hoc Network for Performance of Mobility Models,"

2023 IEEE 8th International Conference for Convergence in Technology (I2CT), Lonavla, India, 2023, pp. 1-6, doi: 10.1109/I2CT57861.2023.10126390.

[36] N. P. Sable, V. U. Rathod, P. N. Mahalle and D. R. Birari, "A Multiple Stage Deep Learning Model for NID in MANETs," 2022 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2022, pp. 1-6, doi: 10.1109/ESCI53509.2022.9758191.

[37] N. P. Sable, M. D. Salunke, V. U. Rathod and P. Dhotre, "Network for Cross-Disease Attention to the Severity of Diabetic Macular Edema and Joint Retinopathy," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-7, doi: 10.1109/SMARTGENCON56628.2022.10083936.

[38] Dwarkanath Pande, S. ., & Hasane Ahammad, D. S. . (2022). Cognitive Computing-Based Network Access Control System in Secure Physical Layer. Research Journal of Computer Systems and Engineering, 3(1), 14–20. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/36.

[39] V. U. Rathod, N. P. Sable, N. N. Thorat and S. N. Ajani, "Deep Learning Techniques Using Lightweight Cryptography for IoT Based E-Healthcare System," 2023 3rd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2023, pp. 1-5, doi: 10.1109/CONIT59222.2023.10205808.

[40] V. U. Rathod and S. V. Gumaste, “Role of Deep Learning in Mobile Ad-hoc Networks”, IJRITCC, vol. 10, no. 2s, pp. 237–246, Dec. 2022.

[41] N. P. Sable, V. U. Rathod, P. N. Mahalle, and P. N. Railkar, “An Efficient and Reliable Data Transmission Service using Network Coding Algorithms in Peer-to-Peer Network”, IJRITCC, vol. 10, no. 1s, pp. 144–154, Dec. 2022.

[42] N. P. Sable, R. Sonkamble, V. U. Rathod, S. Shirke, J. Y. Deshmukh, and G. T. Chavan, “Web3 Chain Authentication and Authorization Security Standard (CAA)”, IJRITCC, vol. 11, no. 5, pp. 70–76, May 2023.

[43] Vijay U. Rathod* & Shyamrao V. Gumaste, “Effect Of Deep Channel To Improve Performance On Mobile Ad-Hoc Networks”, J. Optoelectron. Laser, vol. 41, no. 7, pp. 754–756, Jul. 2022.

[44] Rathod, V.U. and Gumaste, S.V., 2022. Role of Neural Network in Mobile Ad Hoc Networks for Mobility Prediction. International Journal of Communication Networks and Information Security, 14(1s), pp.153-166.

[45] Y. Mali, “A Comparative Analysis of Machine Learning Models for Soil Health Prediction and Crop Selection”, International Journal of Intelligent Systems and Applications in Engineering (IJISAE), vol. 11, no. 10s, pp. 811–828, Aug. 2023.

[46] N. P. Sable, V. U. Rathod, M. D. . Salunke, H. B. Jadhav, R. S. . Tambe, and S. R. . Kothavle, “Enhancing Routing Performance in Software-Defined Wireless Sensor Networks through Reinforcement Learning”, International Journal of Intelligent Systems and Applications in Engineering (IJISAE), vol. 11, no. 10s, pp. 73–83, Aug. 2023.