Internship ID:
ARL-090123-2

SPARTIFICIAL

An ML Model for LIGO Image Classification under the Zooniverse Gravity Spy Project

Pratik Sonavane
Anshuman Prasad
Apoorv Thapliyal
Bharat Sonkiya

Mr. Rohan Shah

Table of Contents

ABSTRACT

In 2015, the Advanced LIGO detectors achieved a remarkable feat by detecting gravitational waves, unlocking insights into cosmic occurrences like black hole collisions. Despite their subtle presence on Earth, these waves yield vital information about black holes. These detectors exhibit sensitivity not only to authentic gravitational signals but also to glitches—brief noise bursts with diverse origins. The project's objective is to pinpoint glitch sources through the analysis of gravitational-wave data, aiding LIGO scientists in mitigating their impact.

Our initiative focuses on constructing a robust machine learning model to classify various gravitational wave glitches identified by LIGO as part of the Zooniverse Gravity Spy project. Leveraging a computer vision CNN model, we trained it on an extensive collection of images sourced from the Gravity Spy project, subsequently validating it on numerous test images. Thanks to meticulous labeling, diverse gravitational wave patterns, and comprehensive evaluation metrics, the model boasts high performance and reliability. By scrutinizing intricate signatures across 22 different wave classes, the model achieves precise and accurate classifications. Furthermore, we've made this model accessible for public testing. The successful creation of this classification model holds profound potential in expediting the identification and categorization of gravitational wave signals, thereby enhancing astrophysical research and our understanding of the universe's dynamic gravitational phenomena.

1. INTRODUCTION

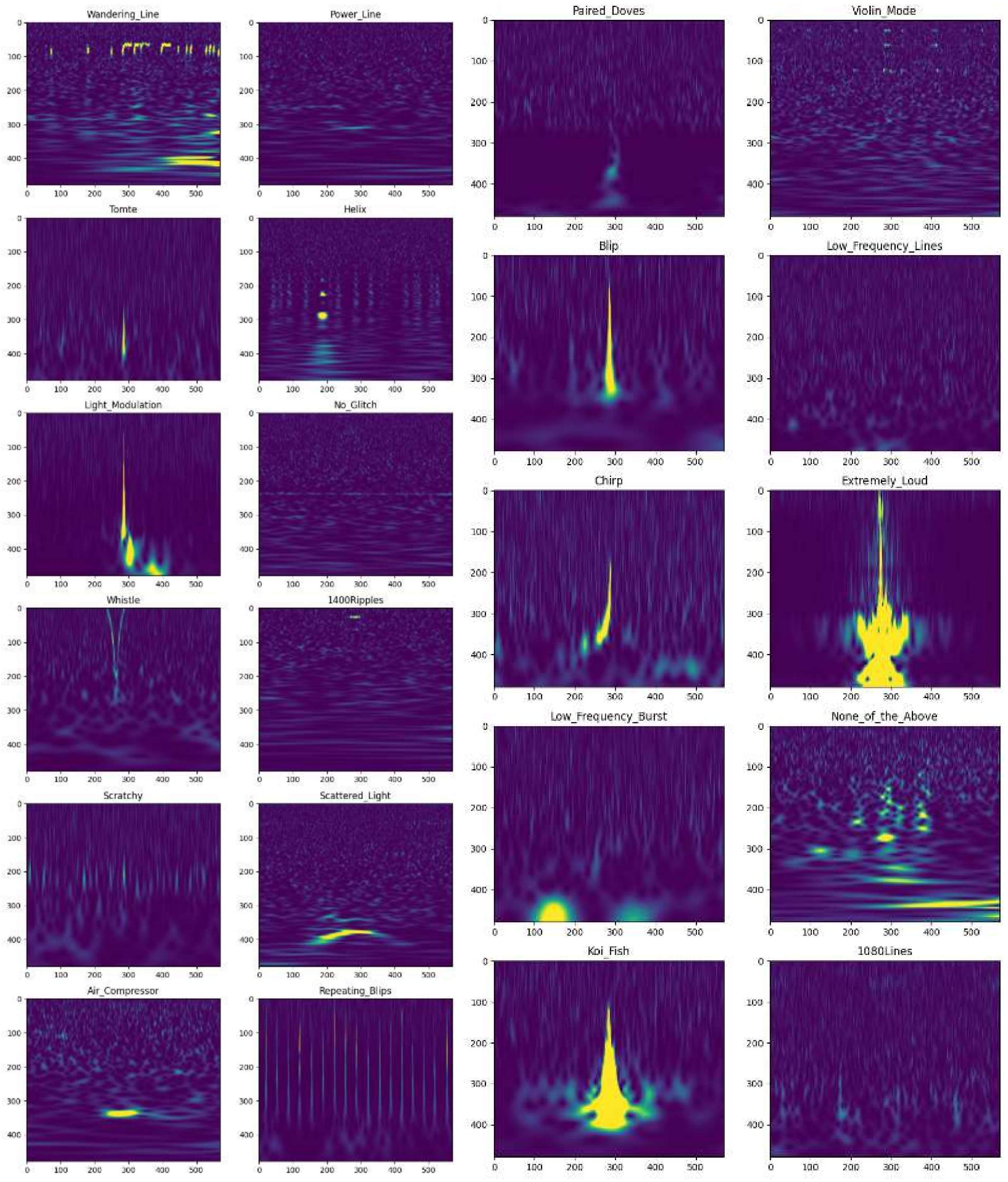
In 2015, a seminal breakthrough marked the observation of gravitational waves, realized through the operational proficiency of the Advanced Laser Interferometer Gravitational-Wave Observatory (LIGO) detectors**[2]**. These waves, born from extreme cosmic events, such as the convergence of black holes, bear profound implications for our comprehension of the universe. Nonetheless, their exceedingly subtle presence upon reaching our observational facilities mandates the employment of instruments of unparalleled sensitivity, as demonstrated by the Advanced LIGO apparatus, to facilitate their quantification. This epochal achievement marks a transformative avenue for advancing our insights into the cosmos, notably encompassing entities such as black holes and other heavy celestial bodies, ones which elude conventional observational modalities.

Central to this pursuit lies the intricate architecture of gravitational-wave detectors, intricate contrivances that serve not only as receptors of authentic gravitational signals but also as conduits for a diverse array of ambient noise sources. Among these, transient noise bursts, termed as glitches, stand as notable challenges. Emergent from an array of possible sources spanning instrumentation errors and environmental variables, glitches cast a formidable shadow over the discernment and interpretation of genuine gravitational-wave emissions. The present endeavor embarks on the deciphering of the origins underpinning glitches, with the objective of informing the LIGO scientific cohort with insights pivotal for effective mitigation strategies.

At the crux of this endeavour exists the progressive evolution of the Zooniverse Gravity Spy initiative**[3][4]**, which initially undertook a systematic classification endeavour aimed at distinguishing between various glitch archetypes. This approach to classification bestows an elemental comprehension of the diverse spectrum of glitch manifestations, a prerequisite to subsequent efforts in tracing their causal agents. Importantly, the collaborative engagements of volunteers culminated in the identification of numerous novel glitch variants. In its present phase, volunteers are brought forth to extend their collaborative undertakings by scrutinizing supplementary datasets derived from the LIGO detectors, thus embarking on a collaborative journey towards unraveling the intrinsic determinants of glitch occurrences.

The objective of our study was to leverage the Kaggle dataset**[1]**, which encompasses a diverse array of glitches identified within the Gravity Spy project, for the training of a model utilizing thousands of images. Our primary focus was on achieving both accuracy and precision in the classification task. Accurate classification of glitch categories not only holds the potential to expedite the advancement of contemporary gravitational wave research but also fosters the exploration of novel avenues in this domain. The classes provided in the dataset were as follows: *1080 Lines, 1400 Ripples, Air Compressor, Blip, Chirp, Extremely Loud, Helix, Koi Fish, Light modulation, Low Frequency Burst, Low Frequency Lines, No Glitch, Paired Doves, Power Line,* *Repeated Blips, Scattered Light, Scratchy, Tomte, Violin Mode, Wandering Line, Whistle and None of the Above.*

The following are sample images representing each class:



In this exposition, we commence by providing a concise overview of the current State-of-the-Art, contextualizing our study within the prevailing research landscape. We subsequently present the multifaceted solutions we devised to surmount the challenges encountered during the training of our model on the given dataset. The progression of our model development throughout the course of this internship is explicated in detail. Subsequently, we scrutinize the efficacy of our model, juxtaposing its performance against the diverse array of models conceived during this internship's duration. Our discourse culminates in an exploration of the insights garnered during the model's training process, underscoring potential future directions for the refinement of this model to address its inherent limitations.

2. REVIEW OF STATE-OF-THE-ART AND RELATED BACKGROUND

During our pursuit of a solution for our defined problem, we explored various approaches available on Kaggle. These approaches entailed training models on the same Kaggle dataset, with the aim of accurately classifying images into their respective categories. Among these, we encountered an implementation that employed a custom Artificial Neural Network (ANN) model and a distinct custom Deep Convolutional Generative Adversarial Network (DC GAN) model**[6]**. Another effective strategy was the application of a customized Convolutional Neural Network (CNN) model, which utilized categorical cross-entropy and accuracy metrics**[5].** Similarly, an alternate custom CNN model with a relatively modest number of layers, coupled with categorical cross-entropy, accuracy metrics, and techniques involving learning rate adjustments, presented notable outcomes**[7]**.

While these methodologies exhibited satisfactory outcomes, they were unable to address a pivotal challenge that arose when training our models on the dataset: the low precision resulting from an inherent imbalance in the dataset. Notably, the class with the highest image count had approximately 67 times more instances in comparison to the class with the lowest number of images. This pronounced imbalance posed persistent difficulties, with several models encountering overfitting issues that manifested as significant disparities between training and validation metrics, and test metrics. Consequently, maintaining a consistently high precision proved challenging. To mitigate this, we engaged in devising model oriented solutions as well as potential preprocessing steps targeted at mitigating this imbalance.

In the culmination of our efforts, our final model achieved an accuracy of approximately 96.31% while maintaining a high precision, surpassing the performance of all other models examined by a substantial margin. This report will delve into the progression leading to the development of this conclusive model, while also offering insights into potential avenues for refining its performance in the future.

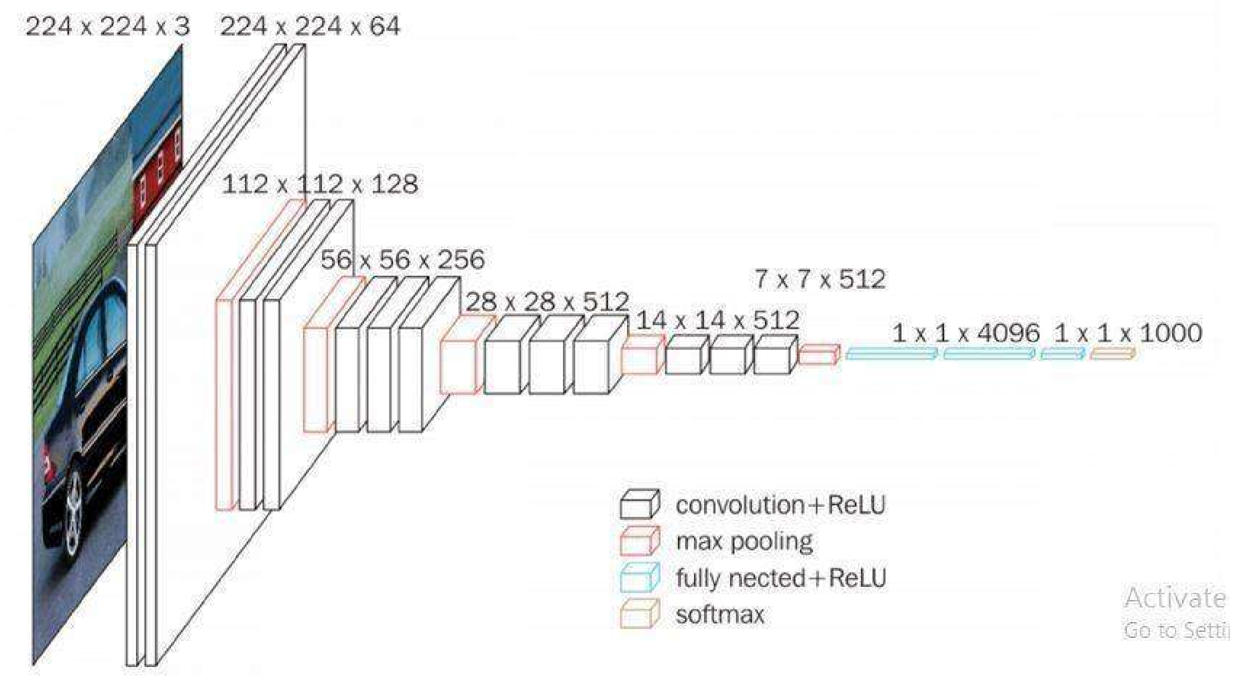
3. PROPOSED SOLUTIONS

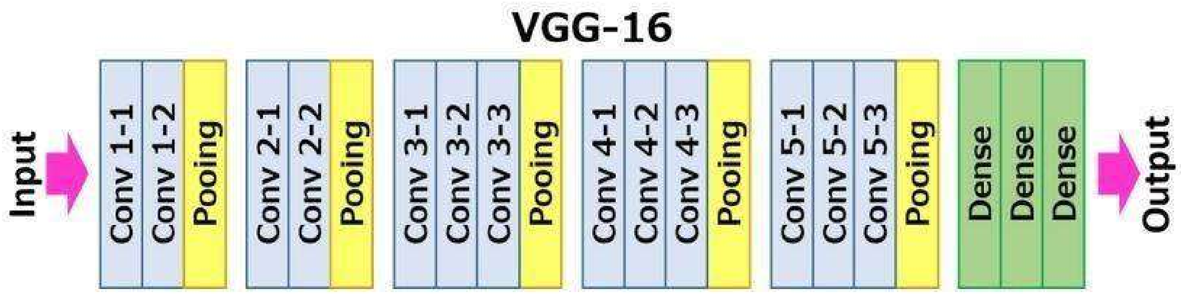
During the initial phases of model development, encompassing training and testing, our primary objective was to ensure not only high accuracy but also high precision, considering the challenges posed by the dataset's inherent class imbalance. To effectively address this issue, we explored various model architectures, focusing on devising strategies that could alleviate the impact of this imbalance. Our efforts led to the formulation of two distinct approaches aimed at rectifying this disparity. The first approach involved introducing a randomization element within the preprocessing stage during batch generation for training. This strategic addition augmented the available data for classes with fewer samples, in hopes of mitigating their data scarcity. The second approach centered on the implementation of class weights prior to the training process. These weights were intentionally biased towards selecting images from classes characterized by lower representation, thereby fostering a more equitable model training process.

An additional consideration pertained to the decision between crafting custom models or fine-tuning pre-existing ones. In pursuit of a comprehensive solution, we adopted a dual-pronged approach. Initially, we constructed bespoke models leveraging the capabilities of the keras and tensorflow libraries in a Google Colab environment. Upon achieving satisfactory outcomes, our focus transitioned to the refinement of pre-built models through fine-tuning, also in a Google Colab environment. The pre-trained models subjected to this process included VGG16, Xception, and ResNet50.

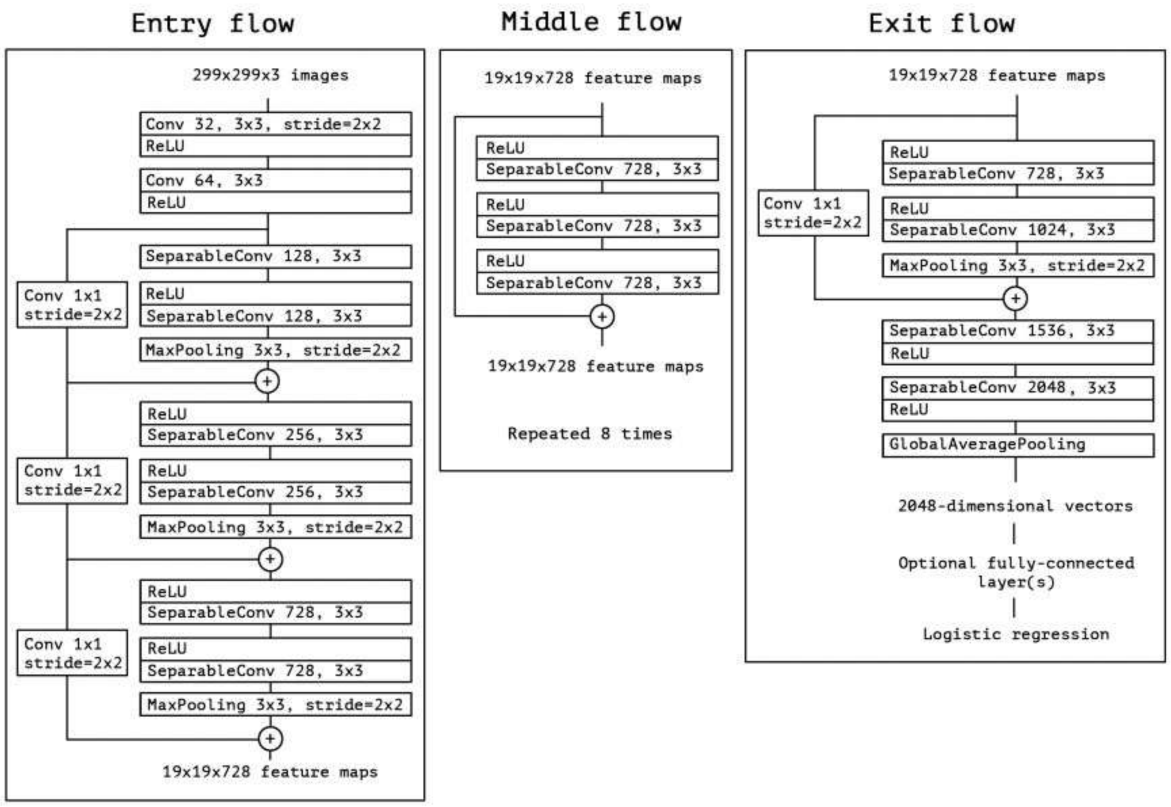
The following illustrate the architectures of all the pre-trained models we used in this project:

VGG16 model architecture[8]:

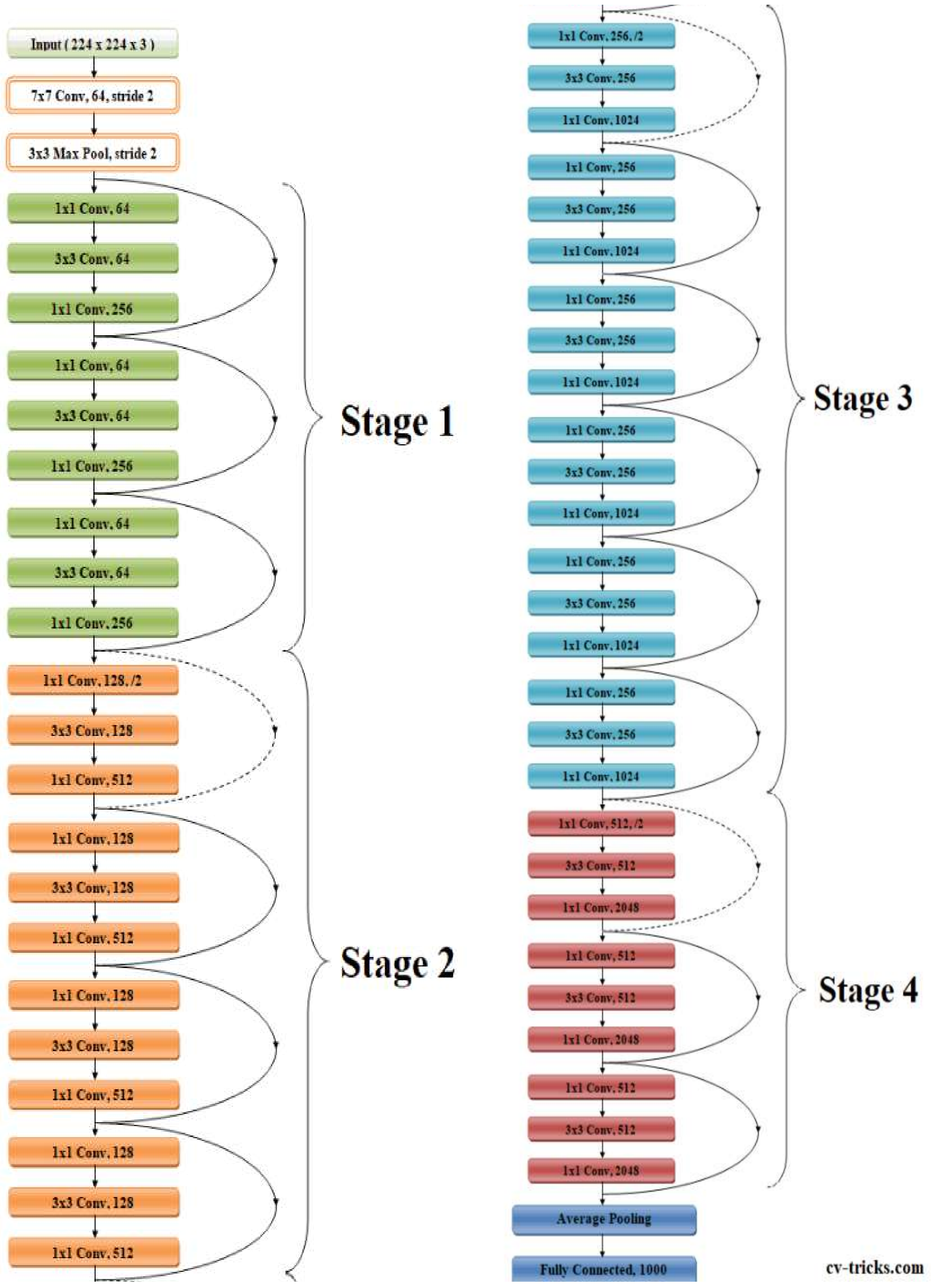




Xception model architecture[9]:



ResNet50 model architecture[10]:



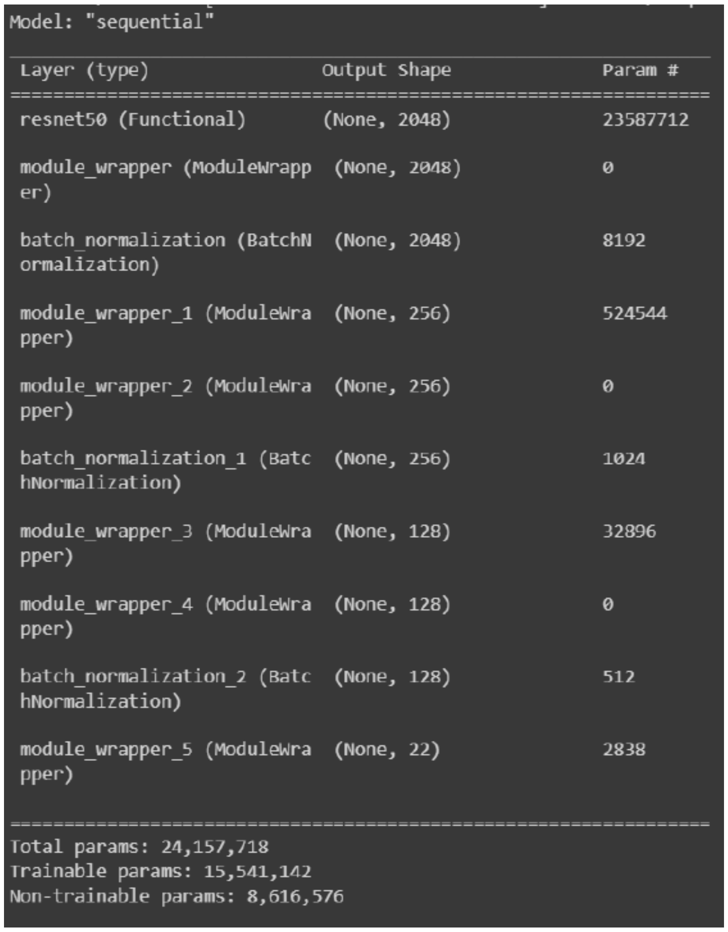
Within the preprocessing phase, our objectives covered expediting training while maintaining precision and addressing dataset imbalance. All our models were standardized to process images as square(224 x 224 or 300 x 300) RGB inputs, normalized with a scaling factor of 1./255. The preprocessing techniques under consideration encompassed samplewise centering, samplewise standard normalization, grayscale conversion, horizontal mirroring, and zooming. Furthermore, our model assessment encompassed a blend of metrics, including Accuracy, Precision, Precision-Recall Curves (PRC), Area Under the ROC Curve (AUC ROC), and Recall. To enhance model efficiency without compromising performance, judicious adjustments were made to our custom models, culminating in optimized performance within minimal training durations.

The variety of custom models we constructed featured arrangement in multiple stacks with each stack including some of the following layers:

- Two Dimensional Convolutional layer to execute the convolution operation on the preprocessed input image.
- ReLu Activation layer to expedite learning as well as introduce non-linearity in the learning process.
- Max Pooling layer to reduce the size of the output of the stack.
- Dropout layer to reduce overfitting.
- Batch Normalization to improve training stability and accelerate convergence.
- Final stack constructed using a Flatten layer followed by Dense layers and a Dropout layer with the final Dense layer having 22 nodes for the each class, along with a SoftMax activation function

Our ultimate model emerged as a meticulously fine-tuned ResNet50 architecture. The preparatory steps involved resizing the images to a standardized 224 by 224 pixel RGB format, followed by rescaling with a factor of 1./255. Further refinement involved the application of samplewise centering and samplewise standard normalization, systematically enhancing the data's suitability for training. In the pursuit of optimizing model performance, a strategic course was adopted. This involved the unfreezing of key components within the ResNet50 architecture, particularly the ultimate stack of dense networks. This stack was thoughtfully replaced with our own tailored implementation featuring custom stacks of Flatten, Batch Normalization, Dense and Dropout layers, effectively harmonizing the architecture with the problem at hand. Additionally, the penultimate stack of layers was also unfrozen, enabling the model to leverage the knowledge acquired from preceding layers. This strategic fine-tuning methodology resulted in noteworthy performance enhancements. The training process covered a comprehensive suite of evaluation metrics, including those mentioned earlier: Accuracy, Precision, Recall, Precision-Recall Curves (PRC), Area Under the ROC Curve (AUC ROC), and Recall. These metrics not only gauged the model's performance but also facilitated its iterative refinement. To optimize the learning process, the RMSprop optimizer was employed, augmented by early stopping and learning rate reduction mechanisms.

This ensemble of techniques enabled the model to converge to a final accuracy of approximately 96.31%. A pivotal facet of model validation was the investigation of the confusion matrix. This analysis revealed that the model's commendable accuracy did not come at the cost of precision, alleviating concerns related to overfitting. The fusion of fine-tuned architecture, meticulous preprocessing, and robust optimization techniques coalesced to yield a model poised to contribute effectively to the classification of gravitational wave glitches.

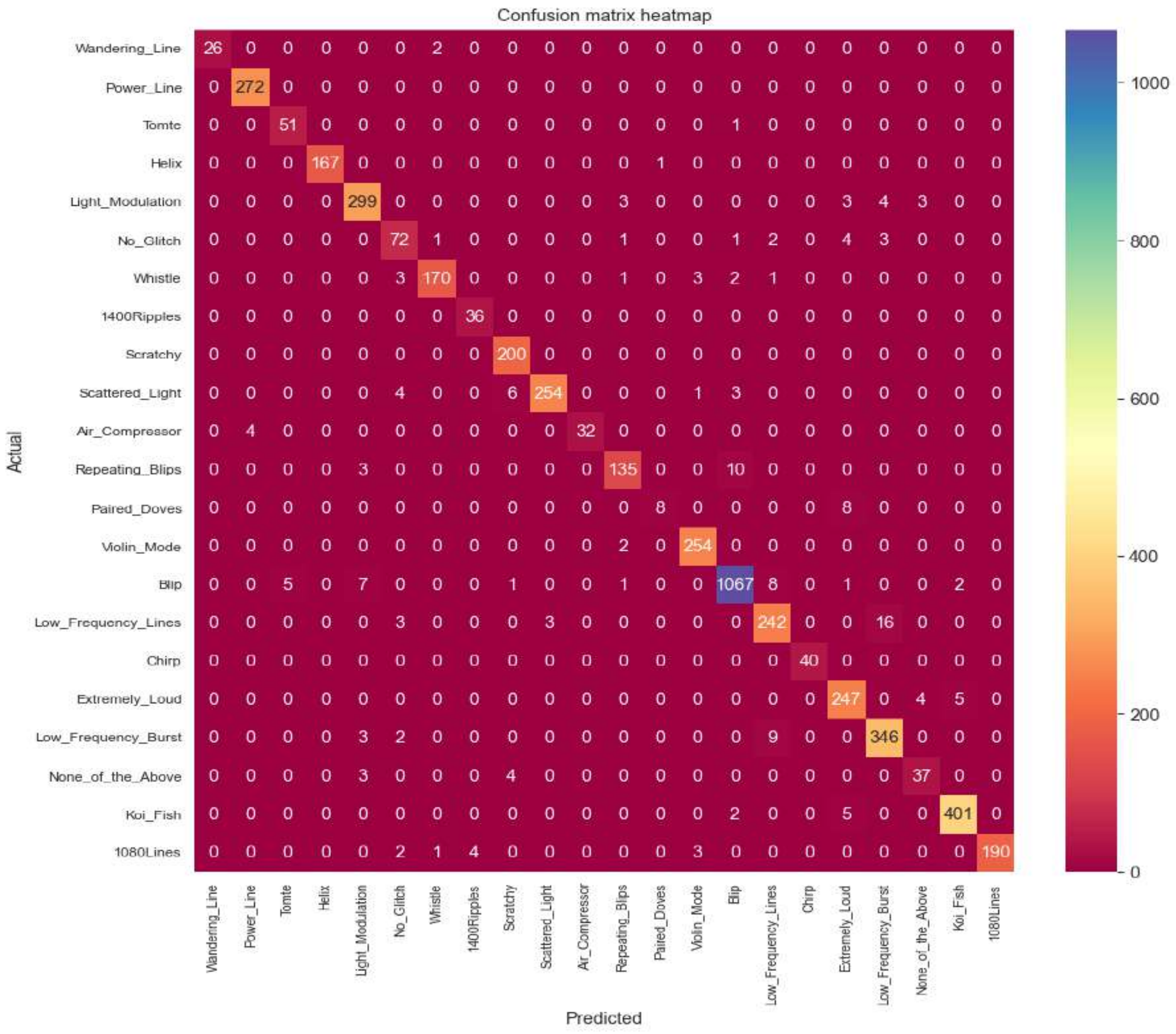


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 module_wrapper (ModuleWrapp  (None, 2048)             0
 er)

 batch_normalization (BatchN  (None, 2048)             8192
 ormalization)

 module_wrapper_1 (ModuleWra  (None, 256)              524544
 pper)

 module_wrapper_2 (ModuleWra  (None, 256)              0
 pper)

 batch_normalization_1 (Batc  (None, 256)              1024
 hNormalization)

 module_wrapper_3 (ModuleWra  (None, 128)              32896
 pper)

 module_wrapper_4 (ModuleWra  (None, 128)              0
 pper)

 batch_normalization_2 (Batc  (None, 128)              512
 hNormalization)

 module_wrapper_5 (ModuleWra  (None, 22)               2838
 pper)

=================================================================
Total params: 24,157,718
Trainable params: 15,541,142
Non-trainable params: 8,616,576
```

The Final Model

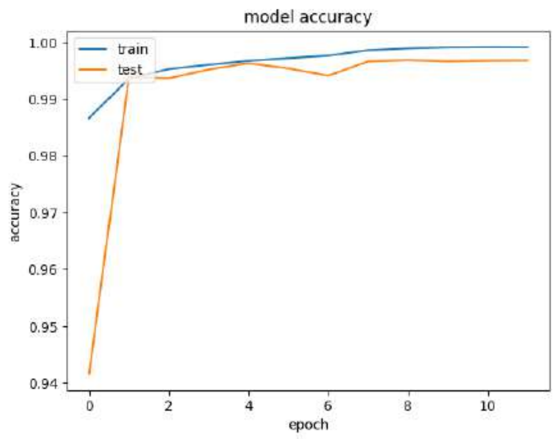
4. EXPERIMENTAL RESULTS AND DISCUSSION

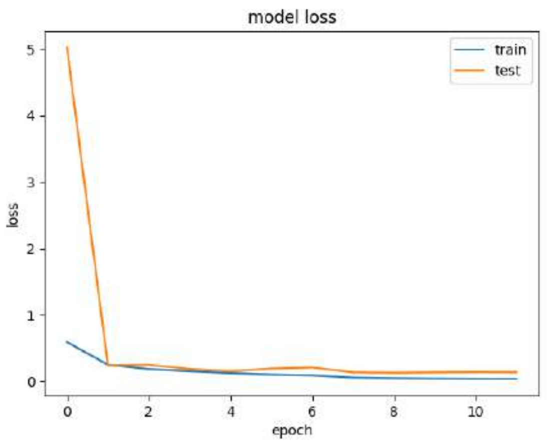
As outlined earlier, our developed model has demonstrated exceptional precision and accuracy, achieving an overall accuracy of 96.31% with a validation accuracy of 99.68%. The accompanying confusion matrix provides a visual representation of our model's performance:



-Accuracy achieved: 96.31%
-Precision(weighted) achieved: 96.35%
-Precision(micro) achieved: 96.31%
-Precision(macro) achieved: 94.96%
-Recall(weighted) achieved: 96.31%
-Recall(micro) achieved: 96.31%
-Recall(macro) achieved: 93.17%

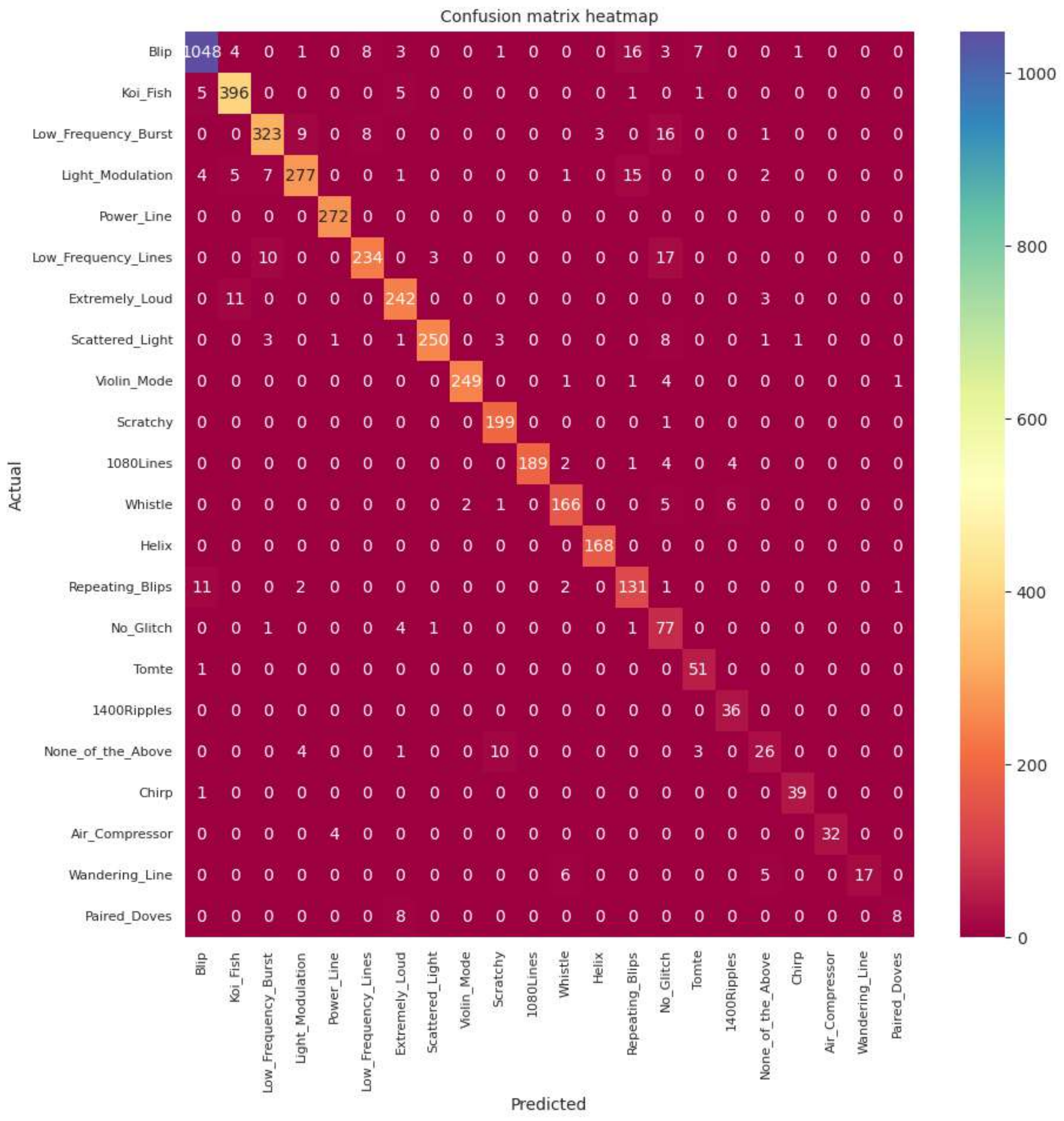
Inspecting the confusion matrix reveals a distinct absence of off-diagonal elements, underscoring the model's remarkable predictive capability. This phenomenon corroborates that our model effectively mitigates the challenges posed by overfitting due to the dataset's inherent imbalance. The ensuing visualizations depict the learning progression of our final model, which was attained through the fine-tuning process of the ResNet50 architecture:





Comparing our final model to an array of preceding iterations, it becomes evident that our final model has yielded heightened accuracy as well as precision. The following section will delve into an insightful examination of a few important metrics across select pivotal models developed over the course of this internship along with their respective confusion matrices:

Best custom model by Pratik:
-Accuracy achieved: 93.86%
-Accuracy by model: 99.50%
-Accuracy by validation: 99.28%
-Precision(weighted) achieved: 94.40%
-Precision(micro) achieved: 93.86%
-Precision(macro) achieved: 90.29%
-Precision by model: 95.62%
-Precision by validation: 93.57%
-Recall(weighted) achieved: 93.86%
-Recall(micro) achieved: 93.86%
-Recall(macro) achieved: 89.36%
-Recall by model: 93.27%
-Recall by validation: 90.29%



Actual \ Predicted	Blip	Koi_Fish	Low_Frequency_Burst	Light_Modulation	Power_Line	Low_Frequency_Lines	Extremely_Loud	Scattered_Light	Violin_Mode	Scratchy	1080Lines	Whistle	Helix	Repeating_Blips	No_Glitch	Tomte	1400Ripples	None_of_the_Above	Chirp	Air_Compressor	Wandering_Line	Paired_Doves
Blip	1048	4	0	1	0	8	3	0	0	1	0	0	0	16	3	7	0	0	1	0	0	0
Koi_Fish	5	396	0	0	0	0	5	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
Low_Frequency_Burst	0	0	323	9	0	8	0	0	0	0	0	0	3	0	16	0	0	1	0	0	0	0
Light_Modulation	4	5	7	277	0	0	1	0	0	0	0	1	0	15	0	0	0	2	0	0	0	0
Power_Line	0	0	0	0	272	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Low_Frequency_Lines	0	0	10	0	0	234	0	3	0	0	0	0	0	0	17	0	0	0	0	0	0	0
Extremely_Loud	0	11	0	0	0	0	242	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0
Scattered_Light	0	0	3	0	1	0	1	250	0	3	0	0	0	0	8	0	0	1	1	0	0	0
Violin_Mode	0	0	0	0	0	0	0	0	249	0	0	1	0	1	4	0	0	0	0	0	0	1
Scratchy	0	0	0	0	0	0	0	0	0	199	0	0	0	0	1	0	0	0	0	0	0	0
1080Lines	0	0	0	0	0	0	0	0	0	0	189	2	0	1	4	0	4	0	0	0	0	0
Whistle	0	0	0	0	0	0	0	0	2	1	0	166	0	0	5	0	6	0	0	0	0	0
Helix	0	0	0	0	0	0	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0
Repeating_Blips	11	0	0	2	0	0	0	0	0	0	0	2	0	131	1	0	0	0	0	0	0	1
No_Glitch	0	0	1	0	0	0	4	1	0	0	0	0	0	1	77	0	0	0	0	0	0	0
Tomte	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	51	0	0	0	0	0	0
1400Ripples	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	0	0	0	0	0
None_of_the_Above	0	0	0	4	0	0	1	0	0	10	0	0	0	0	0	3	0	26	0	0	0	0
Chirp	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	39	0	0	0
Air_Compressor	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0
Wandering_Line	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	5	0	0	17	0
Paired_Doves	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8

Best VGG16 fine tuned model(by Pratik):

-Accuracy achieved: 94.58%

-Accuracy by model: 99.85%

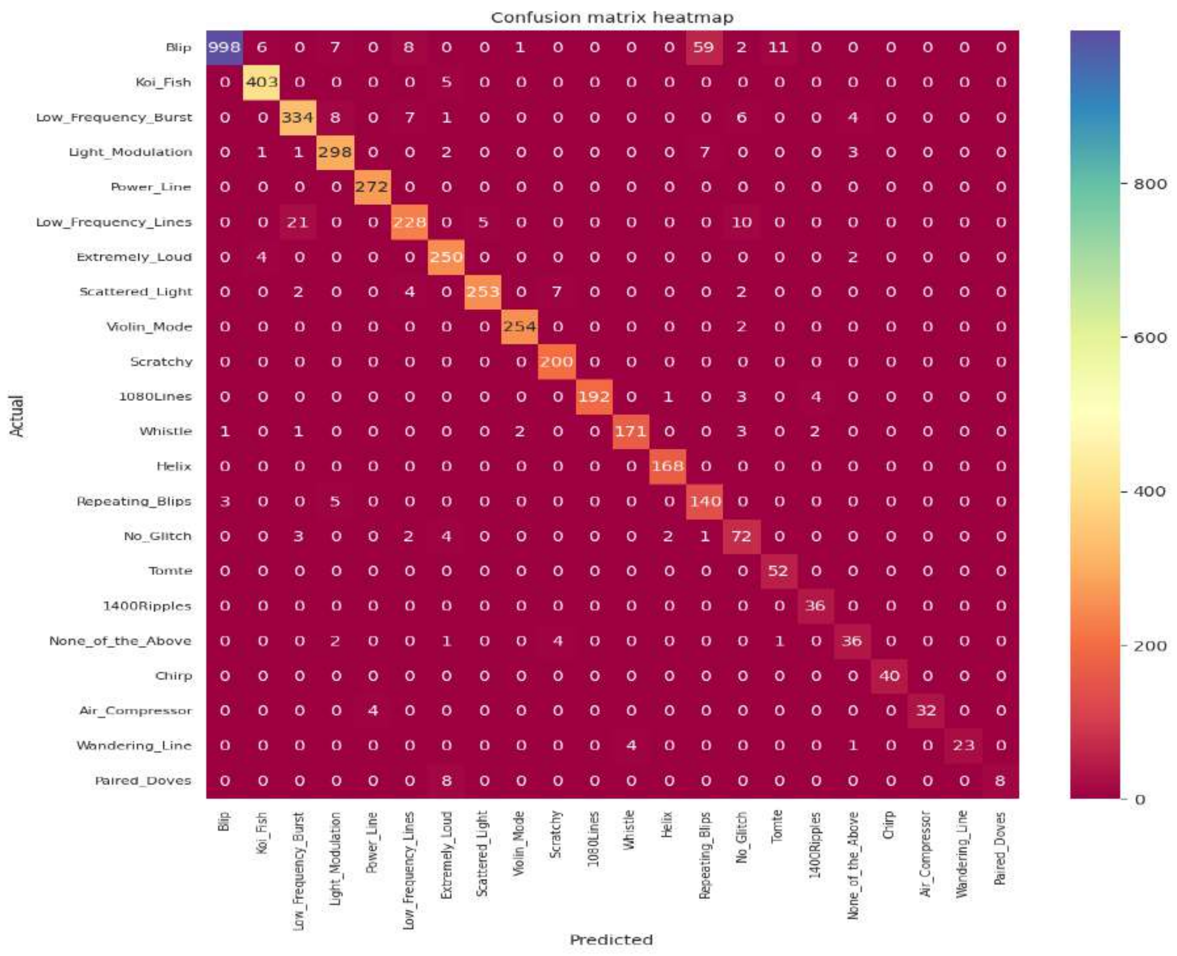
-Accuracy by validation: 99.60%

-Precision(weighted) achieved: 95.92%

-Precision(micro) achieved: 94.58%

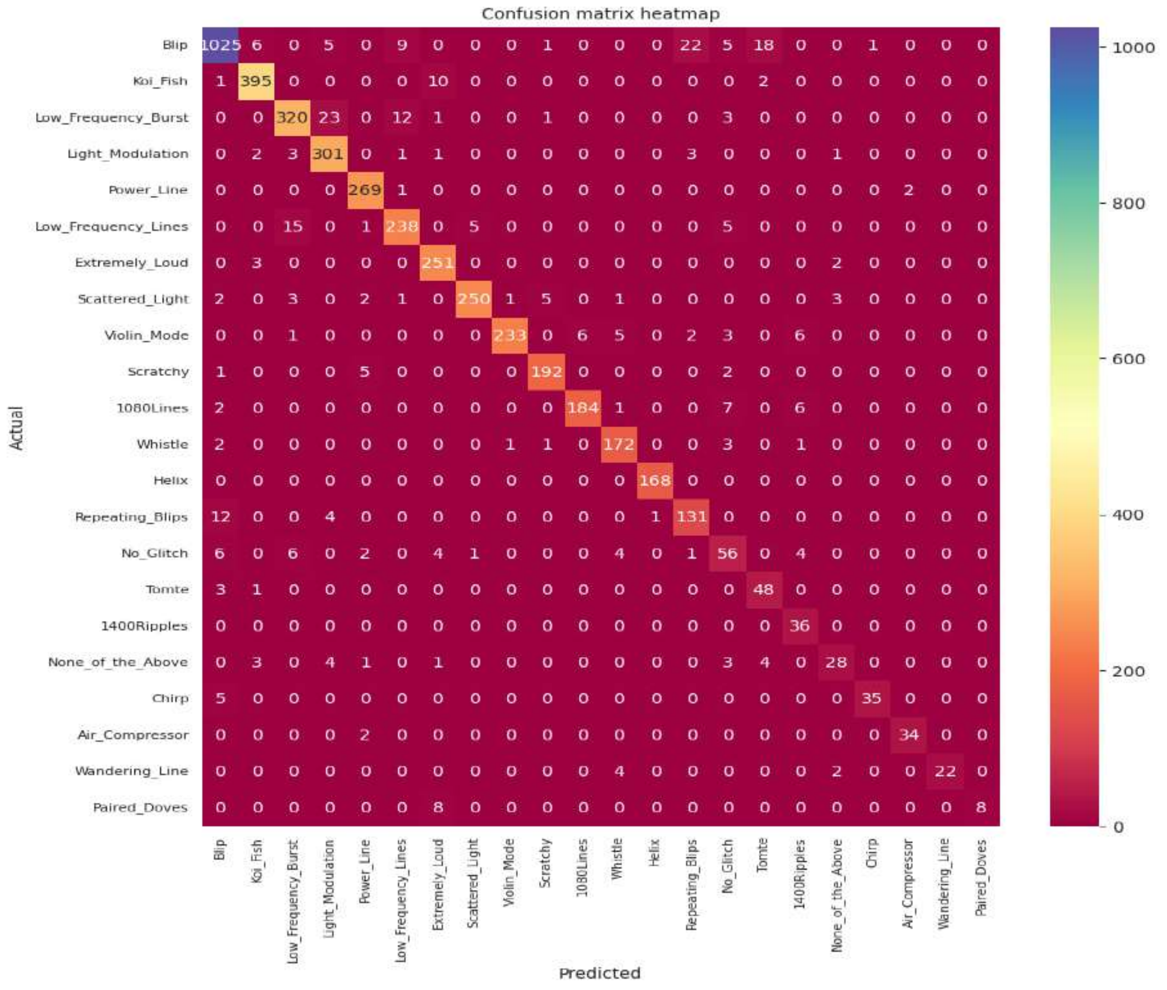
-Precision(macro) achieved: 90.71%

-Precision by model: 99.00%
-Precision by validation: 96.57%
-Recall(weighted) achieved: 94.58%
-Recall(micro) achieved: 94.58%
-Recall(macro) achieved: 93.36%
-Recall by model: 97.74%
-Recall by validation: 94.54%

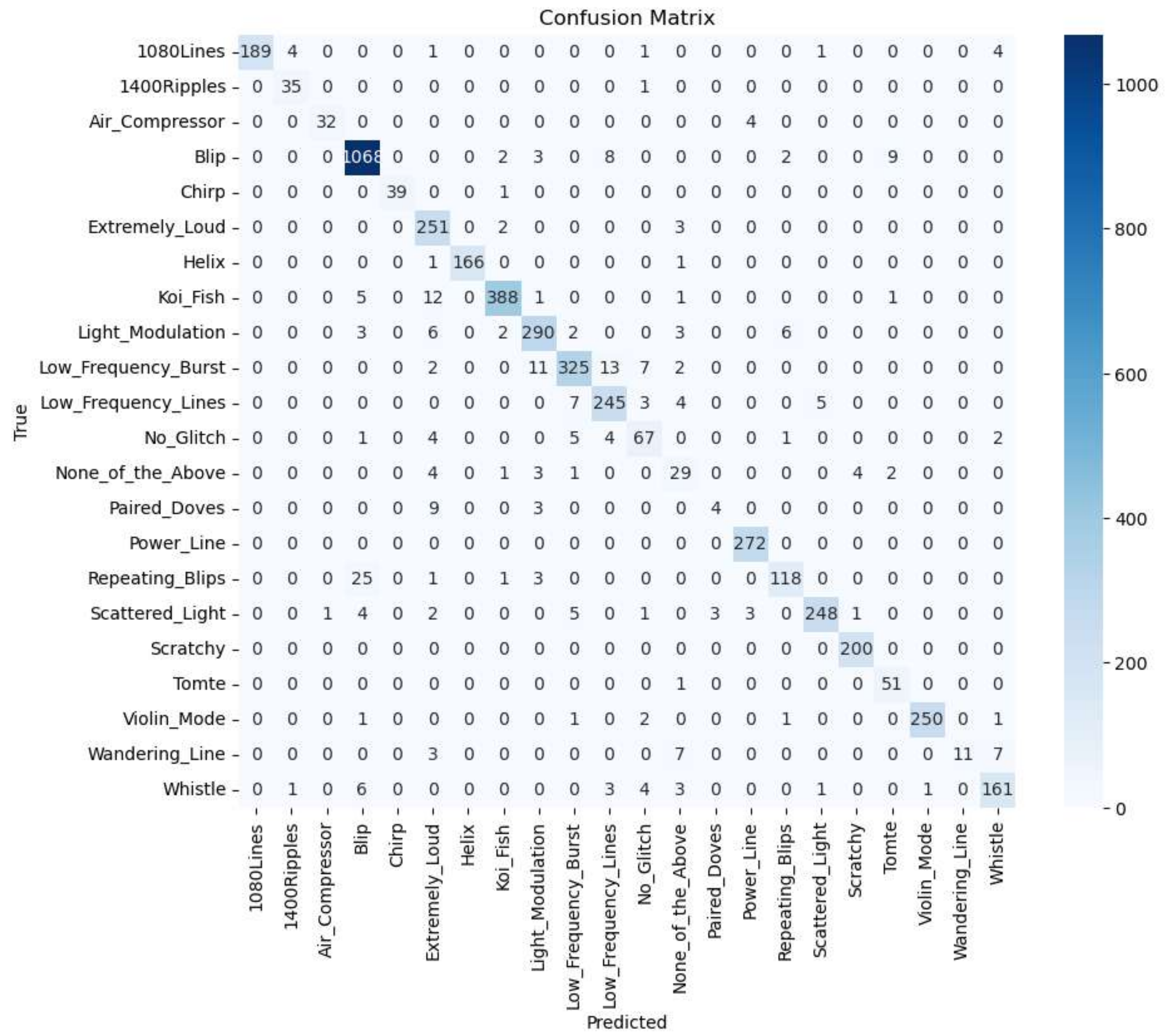


Best Xception fine tuned model(by Pratik):
-Accuracy achieved: 93.14%
-Accuracy by model: 99.76%
-Accuracy by validation: 99.25%
-Precision(weighted) achieved: 93.47%
-Precision(micro) achieved: 93.14%
-Precision(macro) achieved: 90.15%
-Precision by model: 98.01%
-Precision by validation: 92.27%
-Recall(weighted) achieved: 93.14%

-Recall(micro) achieved: 93.14%
-Recall(macro) achieved: 88.76%
-Recall by model: 96.76%
-Recall by validation: 91.21%



Best custom model by Apoorv:
Accuracy achieved: 94.05%
Accuracy by model: 98.54%
Validation accuracy: 93.52%



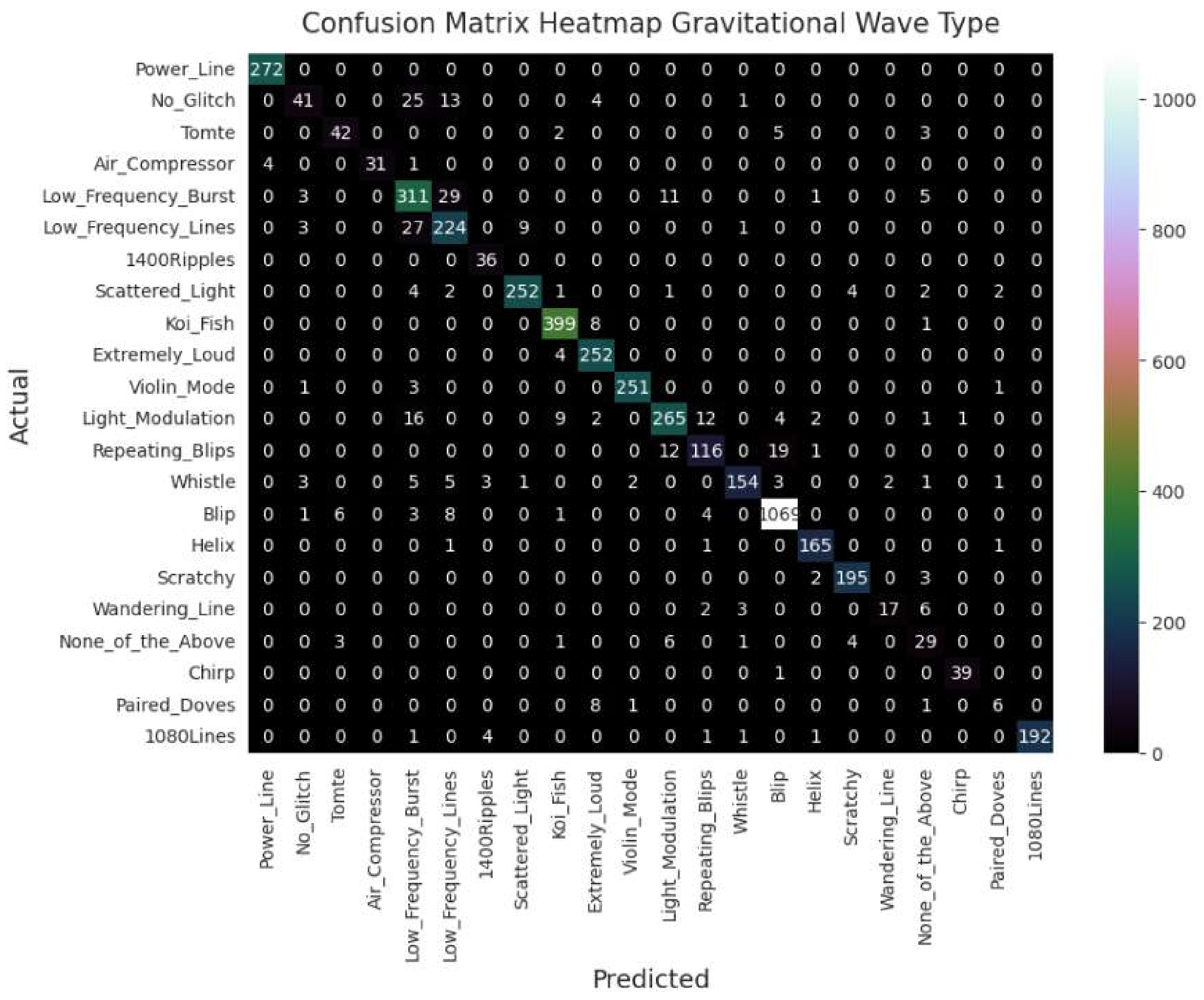
True \ Predicted	1080Lines	1400Ripples	Air_Compressor	Blip	Chirp	Extremely_Loud	Helix	Koi_Fish	Light_Modulation	Low_Frequency_Burst	Low_Frequency_Lines	No_Glitch	None_of_the_Above	Paired_Doves	Power_Line	Repeating_Blips	Scattered_Light	Scratchy	Tomte	Violin_Mode	Wandering_Line	Whistle
1080Lines	189	4	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	4
1400Ripples	0	35	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Air_Compressor	0	0	32	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0
Blip	0	0	0	1068	0	0	0	2	3	0	8	0	0	0	0	2	0	0	9	0	0	0
Chirp	0	0	0	0	39	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Extremely_Loud	0	0	0	0	0	251	0	2	0	0	0	0	3	0	0	0	0	0	0	0	0	0
Helix	0	0	0	0	0	1	166	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Koi_Fish	0	0	0	5	0	12	0	388	1	0	0	0	1	0	0	0	0	0	1	0	0	0
Light_Modulation	0	0	0	3	0	6	0	2	290	2	0	0	3	0	0	6	0	0	0	0	0	0
Low_Frequency_Burst	0	0	0	0	0	2	0	0	11	325	13	7	2	0	0	0	0	0	0	0	0	0
Low_Frequency_Lines	0	0	0	0	0	0	0	0	0	7	245	3	4	0	0	0	5	0	0	0	0	0
No_Glitch	0	0	0	1	0	4	0	0	0	5	4	67	0	0	0	1	0	0	0	0	0	2
None_of_the_Above	0	0	0	0	0	4	0	1	3	1	0	0	29	0	0	0	0	4	2	0	0	0
Paired_Doves	0	0	0	0	0	9	0	0	3	0	0	0	0	4	0	0	0	0	0	0	0	0
Power_Line	0	0	0	0	0	0	0	0	0	0	0	0	0	0	272	0	0	0	0	0	0	0
Repeating_Blips	0	0	0	25	0	1	0	1	3	0	0	0	0	0	0	118	0	0	0	0	0	0
Scattered_Light	0	0	1	4	0	2	0	0	0	5	0	1	0	3	3	0	248	1	0	0	0	0
Scratchy	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	200	0	0	0	0
Tomte	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	51	0	0	0
Violin_Mode	0	0	0	1	0	0	0	0	0	1	0	2	0	0	0	1	0	0	0	250	0	1
Wandering_Line	0	0	0	0	0	3	0	0	0	0	0	0	7	0	0	0	0	0	0	0	11	7
Whistle	0	1	0	6	0	0	0	0	0	0	3	4	3	0	0	0	1	0	0	1	0	161

Best custom model by Anshuman:

-Accuracy achieved: 92.33%

-Accuracy by model: 94.89%

-Accuracy by validation: 91.46%

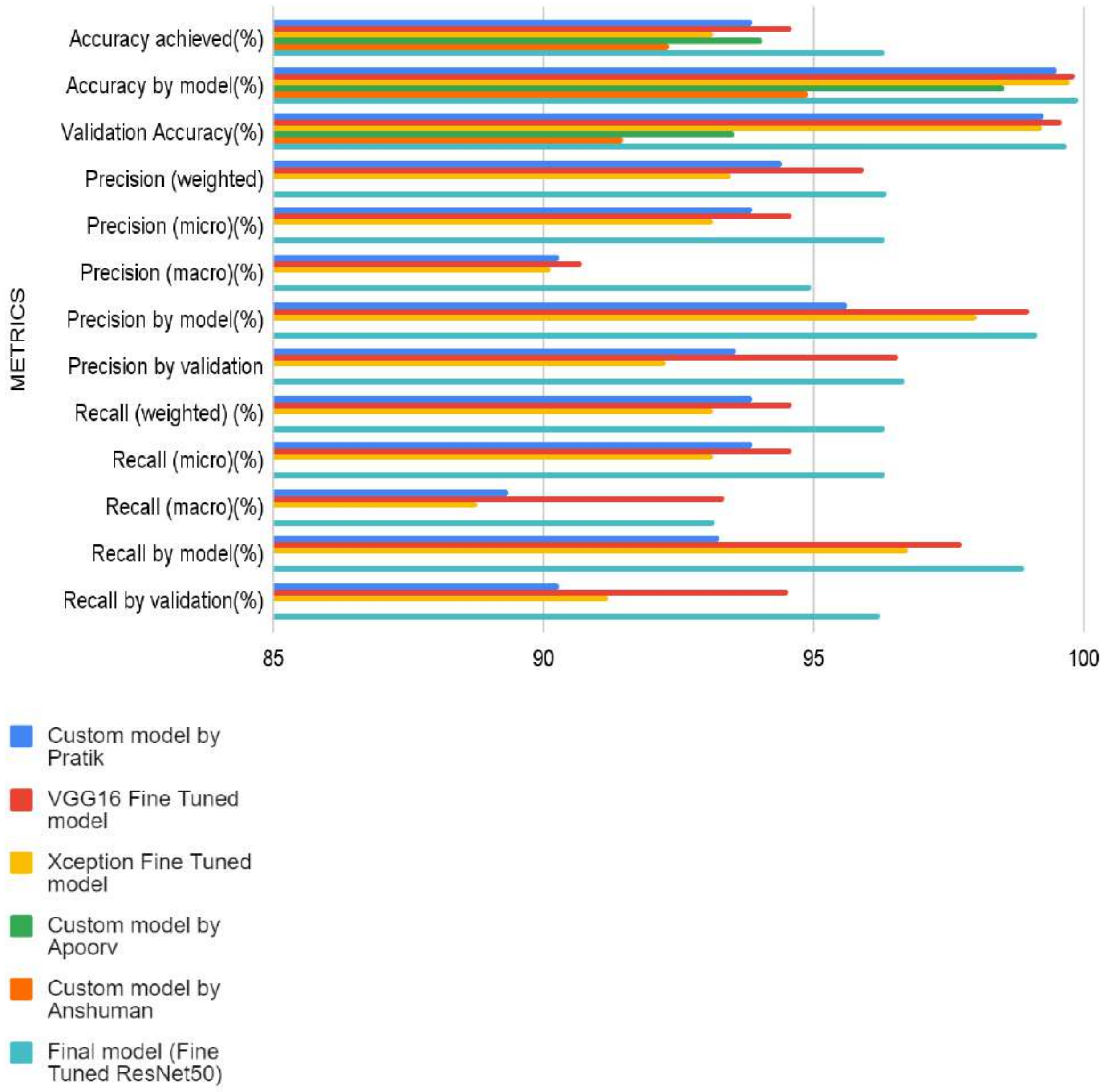


The following is a table which effectively summarises the performance metrics as stated above for each of these models (bold metric indicates the best result in the category):

Model	**Custom model by Pratik**	**VGG16 Fine Tuned model**	**Xception Fine Tuned model**	**Custom model by Apoorv**	**Custom model by Anshuman**	**Final model (Fine Tuned ResNet50)**
Accuracy achieved(%)	93.86	94.58	93.14	94.05	92.33	**96.31**
Accuracy by model(%)	99.50	99.85	99.76	98.54	94.89	**99.91**
Validation	99.28	99.60	99.25	93.52	91.46	**99.68**

Accuracy(%)						
Precision (weighted) (%)	94.40	95.92	93.47	N.A.	N.A.	**96.35**
Precision (micro)(%)	93.86	94.58	93.14	N.A.	N.A.	**96.31**
Precision (macro)(%)	90.29	90.71	90.15	N.A.	N.A.	**94.96**
Precision by model(%)	95.62	99.00	98.01	N.A.	N.A.	**99.14**
Precision by validation(%)	93.57	96.57	92.27	N.A.	N.A.	**96.69**
Recall (weighted) (%)	93.86	94.58	93.14	N.A.	N.A.	**96.31**
Recall (micro)(%)	93.86	94.58	93.14	N.A.	N.A.	**96.31**
Recall (macro)(%)	89.36	**93.36**	88.76	N.A.	N.A.	93.17
Recall by model(%)	93.27	97.74	96.76	N.A.	N.A.	**98.89**
Recall by validation(%)	90.29	94.54	91.21	N.A.	N.A.	**96.23**

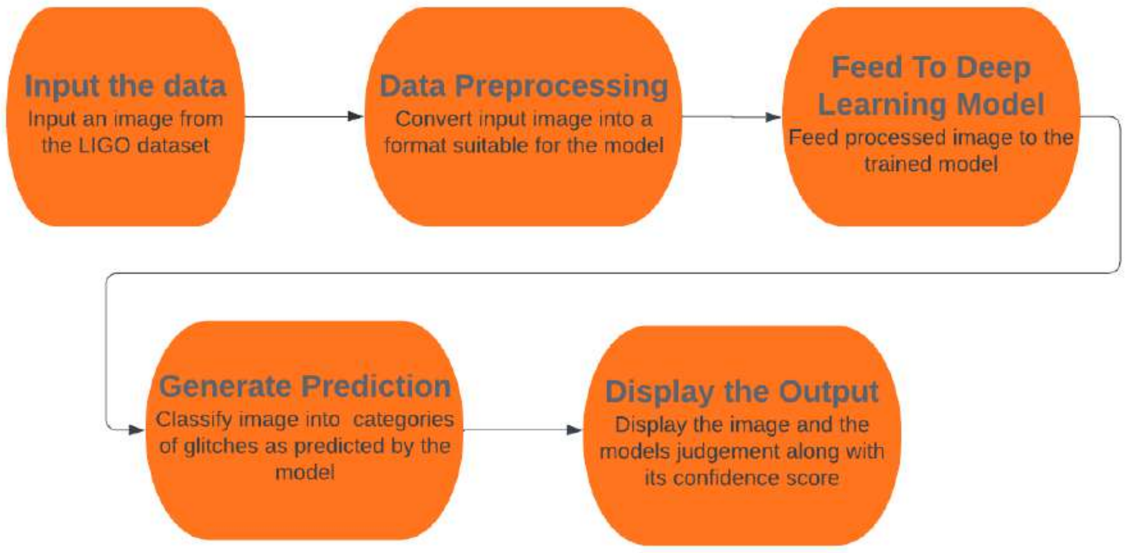
The following chart illustrates this data further:



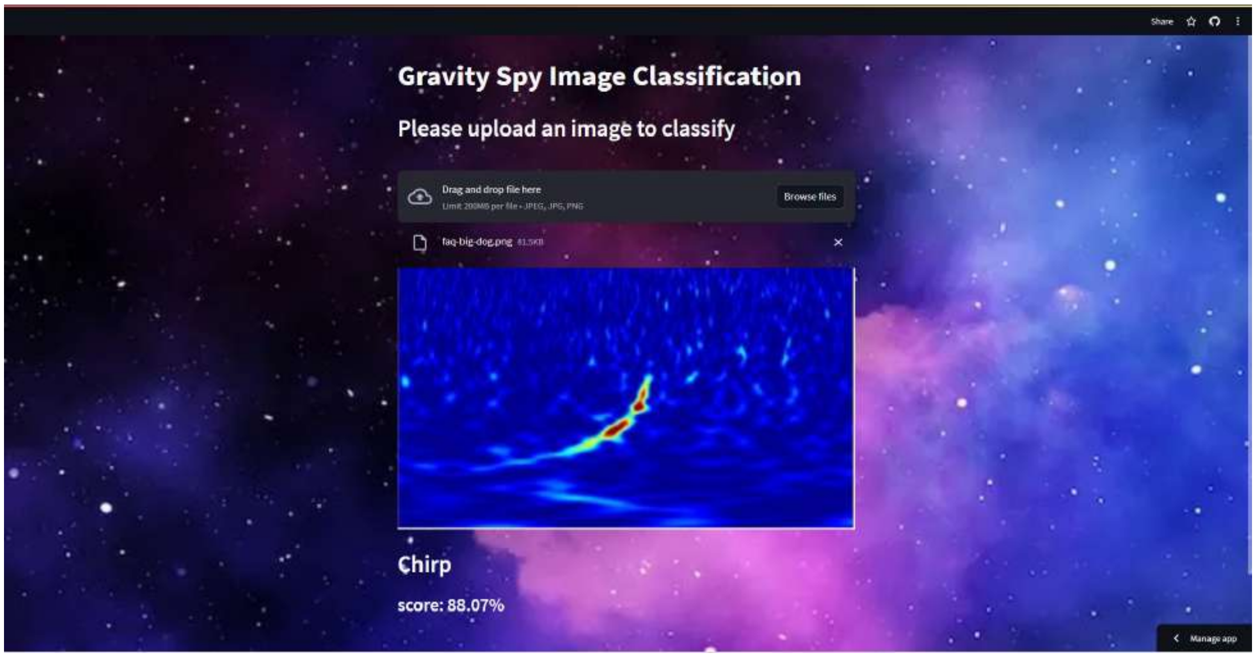
As is evident from the data above, in 6 out of the 7 test metrics as well as 12 out of the 13 metrics listed above, the final model had the best performance out of all the models we developed. Furthermore, the disparity between the training and validation metrics, and the test metrics for the models above is indicative of the difficulties faced because of overfitting of data. This highlights the significant advantage the final model had over all the models developed previously.

We have successfully concluded the deployment phase of our model, rendering it available to the public for comprehensive testing and exploration. To facilitate this accessibility, we leveraged the capabilities offered by Streamlit services and have also made our model repository openly accessible on the GitHub platform. We implore users to test our model further through our publicly available, free and open Gravity Spy Image Classification WebApp powered by Streamlit services.

The following is a flowchart that explains the basic working of the WebApp:


Input the data
Input an image from the LIGO dataset
Data Preprocessing
Convert input image into a format suitable for the model
Feed To Deep Learning Model
Feed processed image to the trained model
Generate Prediction
Classify image into categories of glitches as predicted by the model
Display the Output
Display the image and the models judgement along with its confidence score

The following are examples of the Gravity Spy Image Classification WebApp showcasing the predictive capabilities of our model on *test* images:



Gravity Spy Image Classification

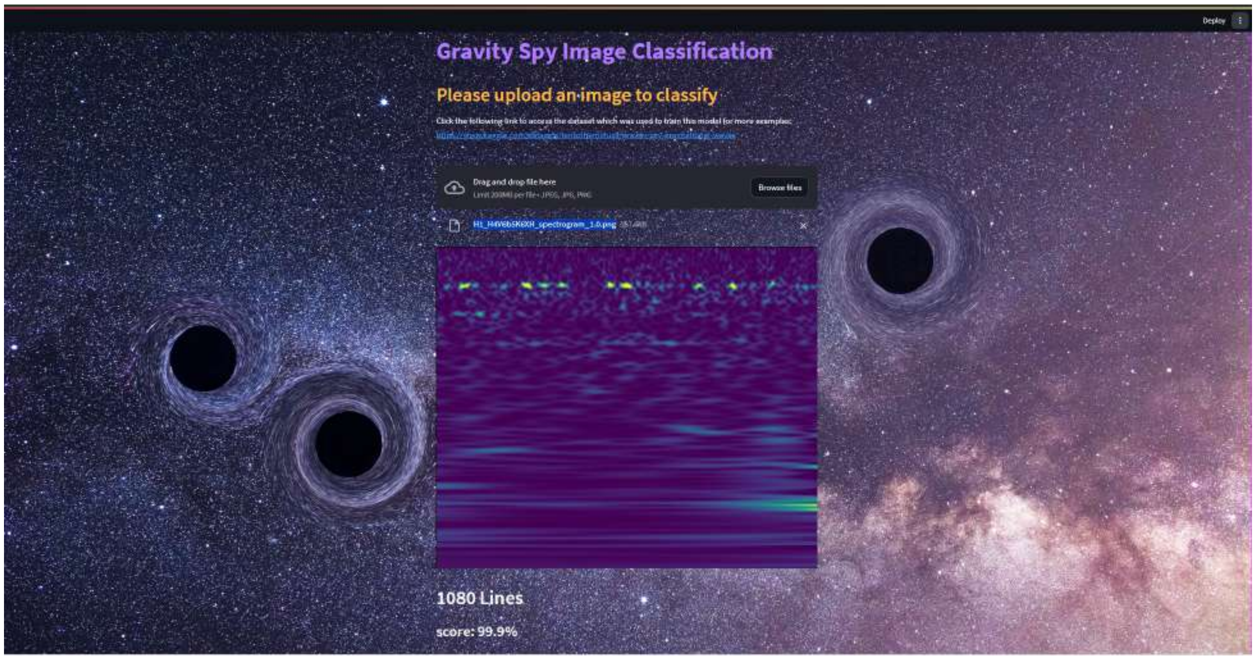
Please upload an image to classify

faq-big-dog.png 81.5KB

Chirp

score: 88.07%

The first direct observation of gravitational waves was made on 14 September 2015 and this wave is being classified correctly by our model through our WebApp



Gravity Spy Image Classification

Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:

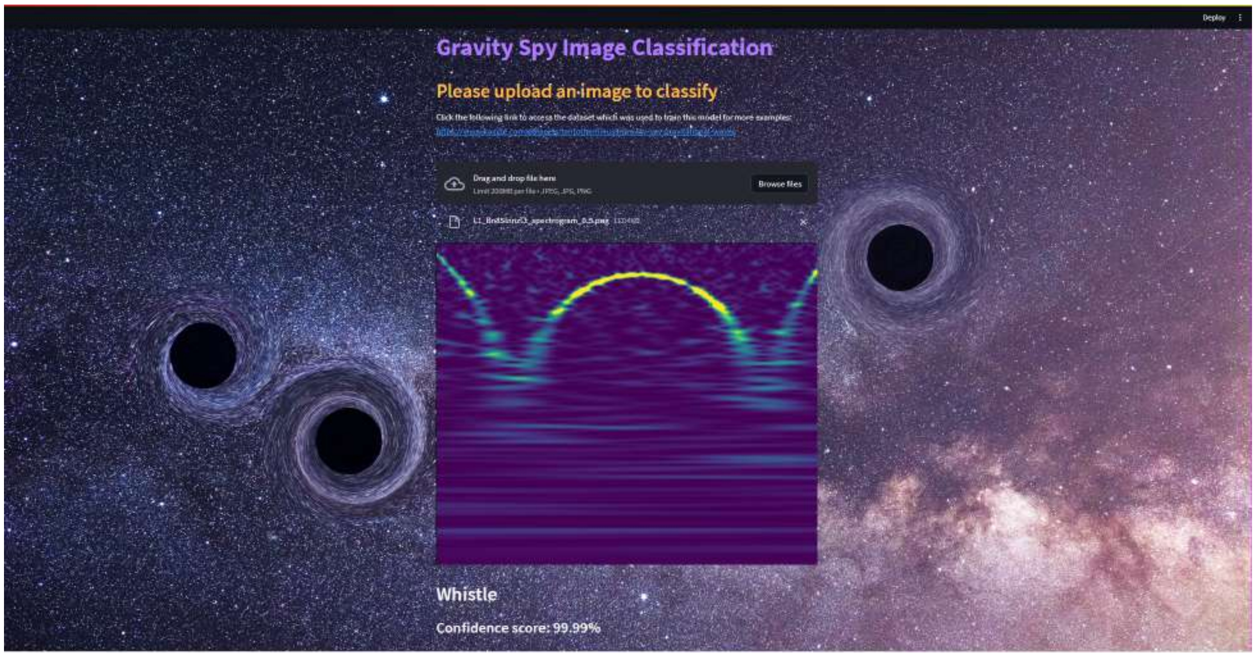
[illegible]

H1_H4V6b5K6XH_spectrogram_1.0.png

1080 Lines

score: 99.9%

H1_H4V6b5K6XH_spectrogram_1.0.png - *1080 Lines*


Deploy
Gravity Spy Image Classification
Please upload an image to classify
Drag and drop file here
Browse files

Click the following link to access the dataset which was used to train this model for more examples:
[illegible]

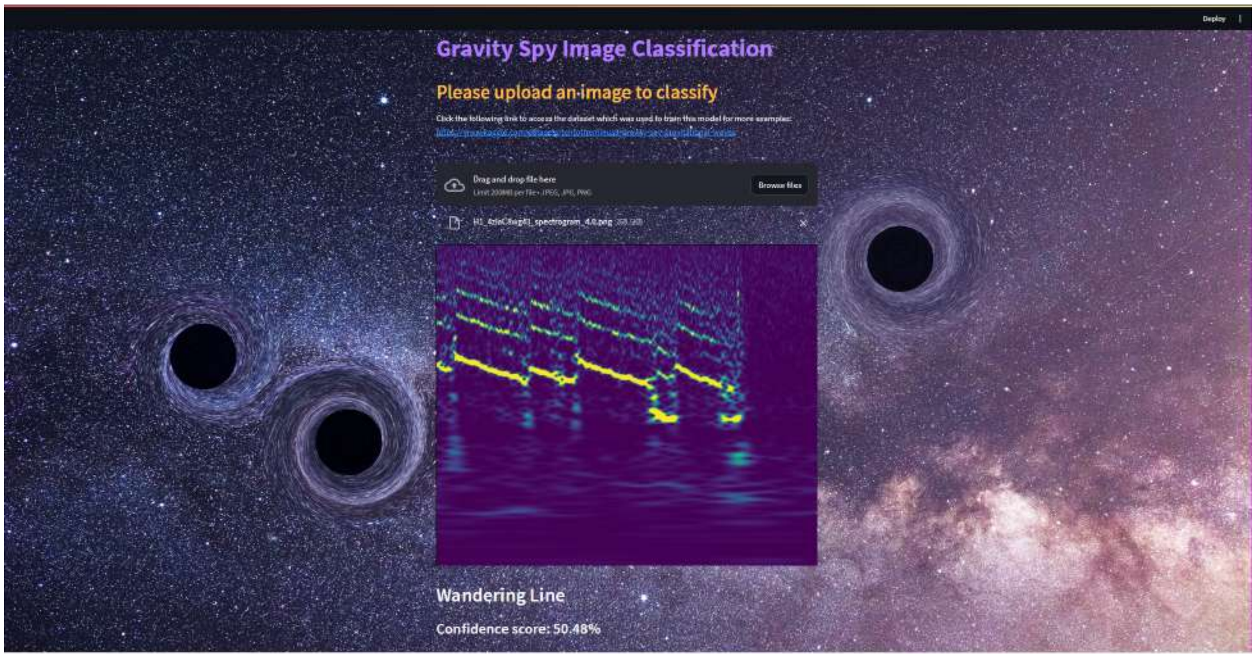
Limit 200MB per file • JPEG, JPG, PNG

L1_Bn85lnnzCt_spectrogram_0.5.png [illegible]

Whistle

Confidence score: 99.99%

L1_Bn85lnnzCt_spectrogram_0.5.png - *Whistle*


Deploy
Gravity Spy Image Classification
Please upload an image to classify
Drag and drop file here
Browse files

Click the following link to access the dataset which was used to train this model for more examples:
[illegible]

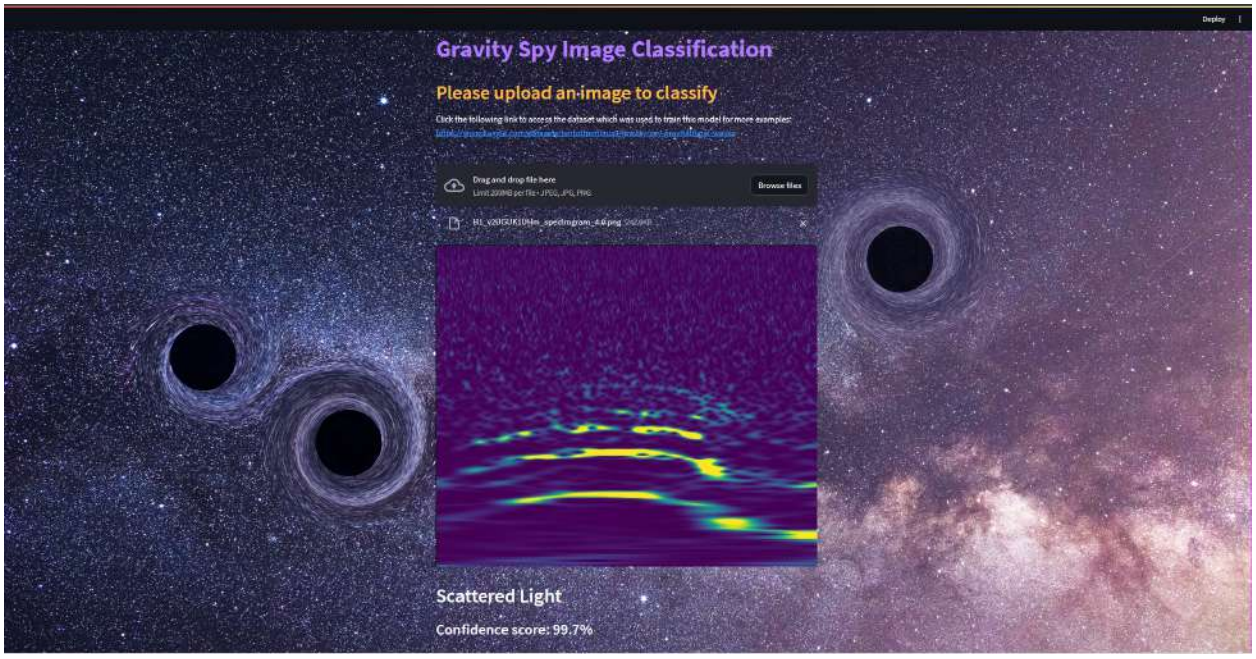
Limit 200MB per file • JPEG, JPG, PNG

H1_4zIaC8wg43_spectrogram_4.0.png [illegible]

Wandering Line

Confidence score: 50.48%

H1_4zIaC8wg43_spectrogram_4.0.png - *Wandering Line*


Deploy
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_v2UGUK1UHm_spectrogram_4.0.png

Gravity Spy Image Classification

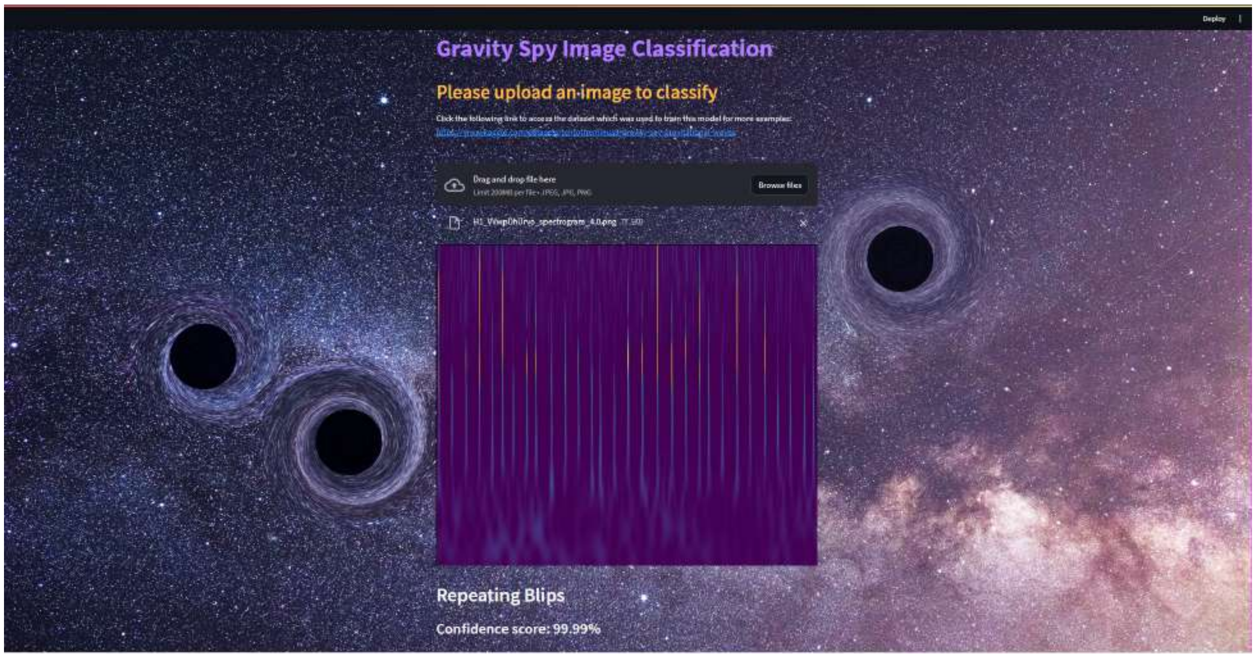
Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:
[illegible]

Scattered Light

Confidence score: 99.7%

H1_v2UGUK1UHm_spectrogram_4.0.png - *Scattered Light*


Deploy
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_VVwpDhUrvo_spectrogram_4.0.png

Gravity Spy Image Classification

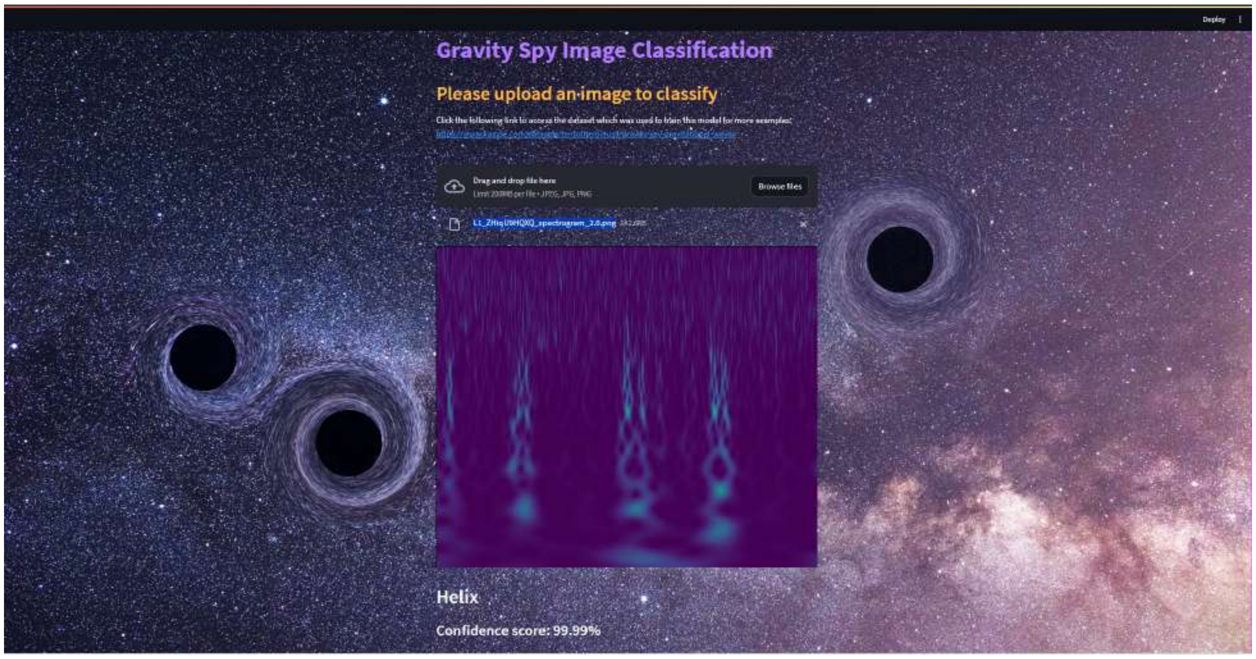
Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:
[illegible]

Repeating Blips

Confidence score: 99.99%

H1_VVwpDhUrvo_spectrogram_4.0.png - *Repeated Blips*


Deploy
Gravity Spy Image Classification
Please upload an image to classify
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
L1_ZHiqU9HQXQ_spectrogram_2.0.png

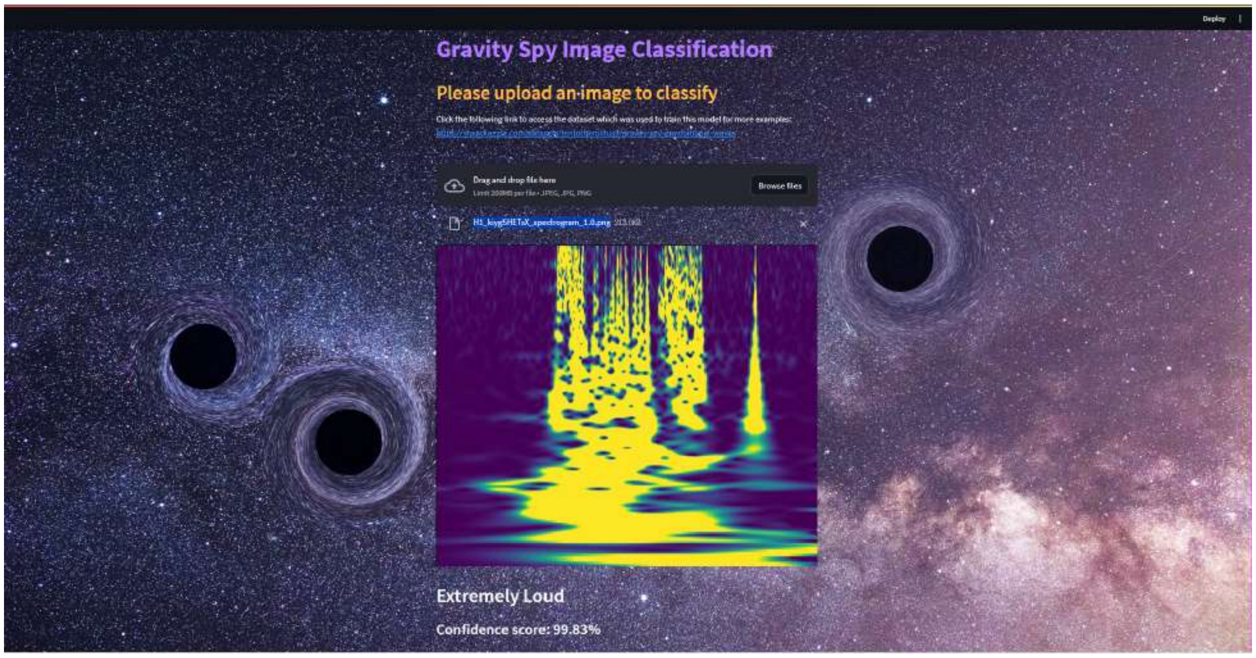
Click the following link to access the dataset which was used to train this model for more examples:

[illegible]

Helix

Confidence score: 99.99%

L1_ZHiqU9HQXQ_spectrogram_2.0.png - *Helix*


Deploy
Gravity Spy Image Classification
Please upload an image to classify
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_kiyg5HETsX_spectrogram_1.0.png

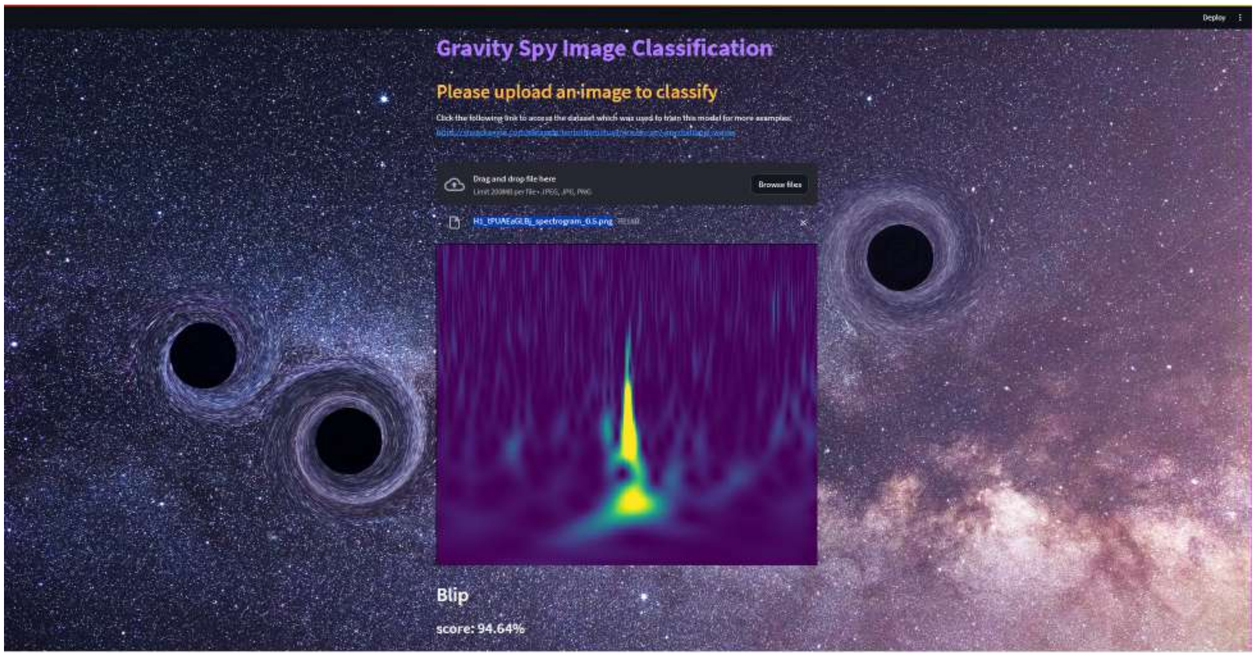
Click the following link to access the dataset which was used to train this model for more examples:

[illegible]

Extremely Loud

Confidence score: 99.83%

H1_kiyg5HETsX_spectrogram_1.0.png - *Extremely Loud*


Deploy
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_tPUAEaGLBj_spectrogram_0.5.png

Gravity Spy Image Classification

Please upload an image to classify

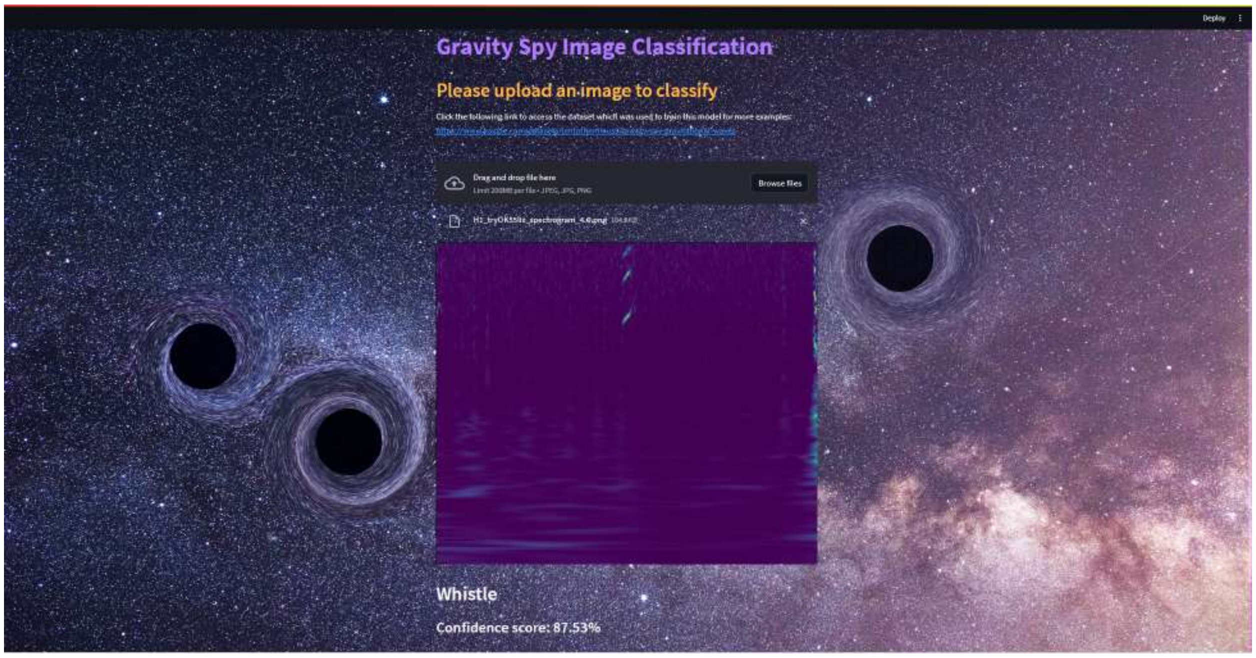
Click the following link to access the dataset which was used to train this model for more examples:

Blip

score: 94.64%

H1_tPUAEaGLBj_spectrogram_0.5.png - *Blip*

The following are some inaccurate classifications:


Deploy
Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_tryOK55Iix_spectrogram_4.0.png

Gravity Spy Image Classification

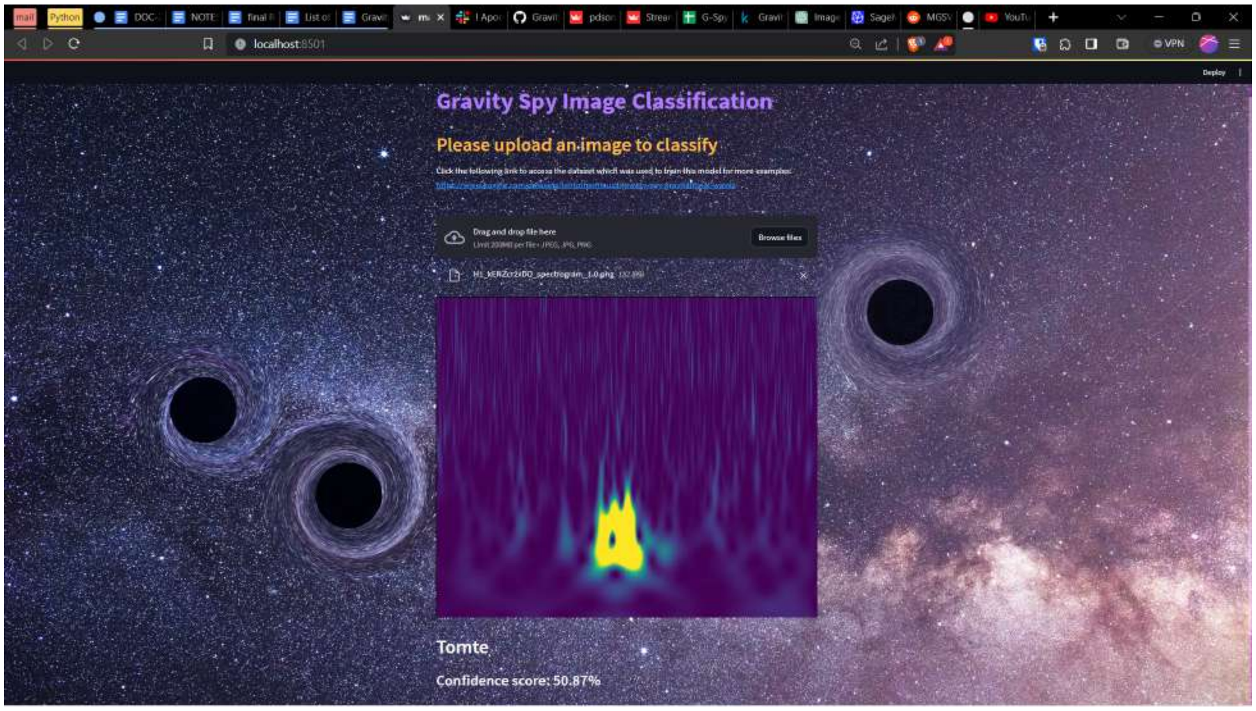
Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:

Whistle

Confidence score: 87.53%

H1_tryOK55Iix_spectrogram_4.0.png - *Wandering Line* being misclassified as *Whistle*


Gravity Spy Image Classification

Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:

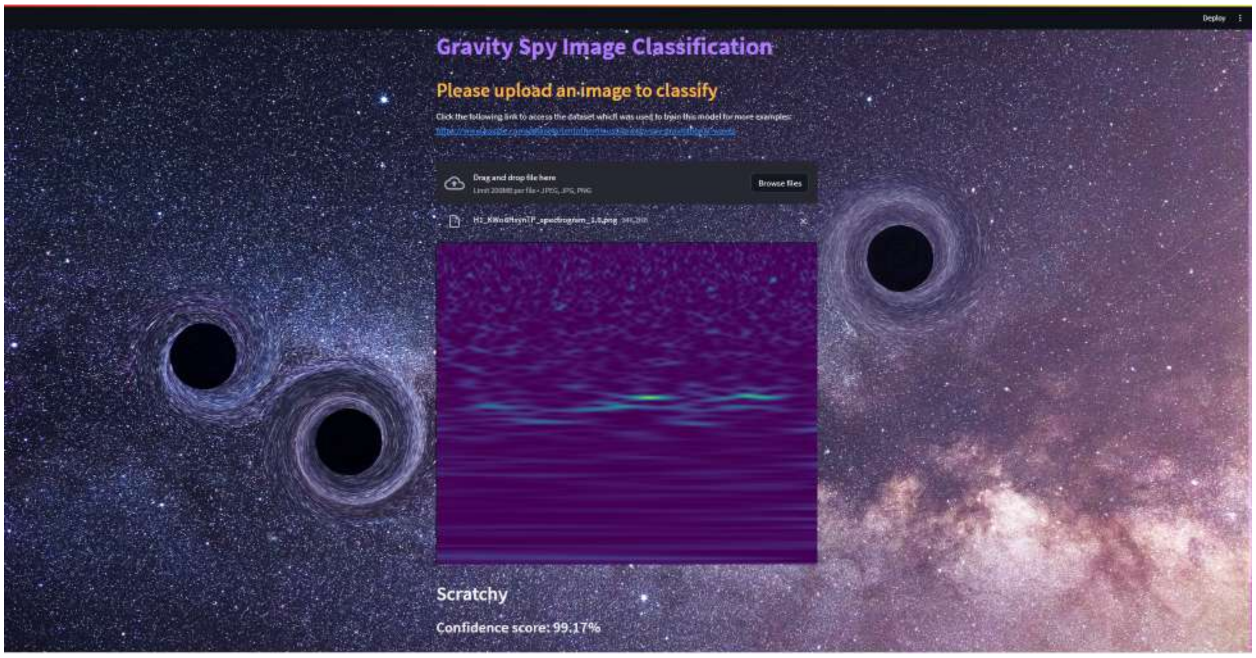
[illegible]

H1_kENZcr2xDO_spectrogram_1.0.png

Tomte

Confidence score: 50.87%

H1_kENZcr2xDO_spectrogram_1.0.png - *None of the Above* type image being misclassified as *Tomte*


Gravity Spy Image Classification

Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:

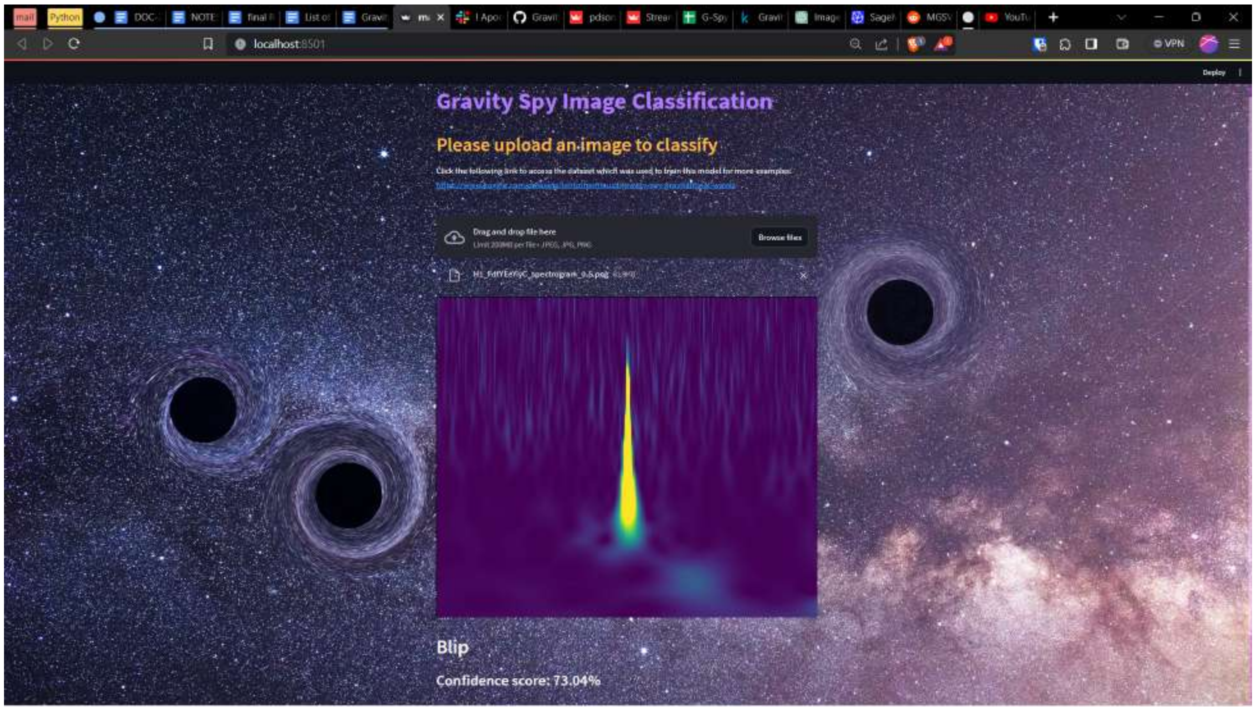
[illegible]

H1_KWo6HxynTP_spectrogram_1.0.png

Scratchy

Confidence score: 99.17%

H1_KWo6HxynTP_spectrogram_1.0.png - *None of the Above* type image being misclassified as *Scratchy*


Gravity Spy Image Classification

Please upload an image to classify

Click the following link to access the dataset which was used to train this model for more examples:

[illegible]

Drag and drop file here
Limit 200MB per file • JPEG, JPG, PNG
Browse files
H1_FdtYEeYiyC_spectrogram_0.5.png

Blip

Confidence score: 73.04%

H1_FdtYEeYiyC_spectrogram_0.5.png - *Repeated Blips* being misclassified as *Blip*

Our intention in adopting this open approach is to cultivate an environment conducive to collaborative engagement and methodical refinement. By providing unfettered access to our model's codebase on GitHub, we invite interested parties to actively participate in its evolution. The repository will serve as a hub for monitoring updates, conducting empirical evaluations, sharing constructive insights, and engaging in an iterative dialogue aimed at augmenting the model's performance. Furthermore, we have harnessed the Streamlit WebApp services to construct an interface that enables users to directly interact with and appraise our model's functionalities. This dual accessibility, through both GitHub and Streamlit, underlines our commitment to fostering a culture of shared inquiry and collective advancement within the scientific community.

With these measures in place, we anticipate that our model will serve as a springboard for ongoing discussions, refinements, and collaborative contributions. This commitment to open access underscores our aspiration to harness the collective intellect to further enhance the model's capabilities, aligning with the progressive nature of scientific exploration, collaboration and innovation.

5. CONCLUSION AND FUTURE SCOPE

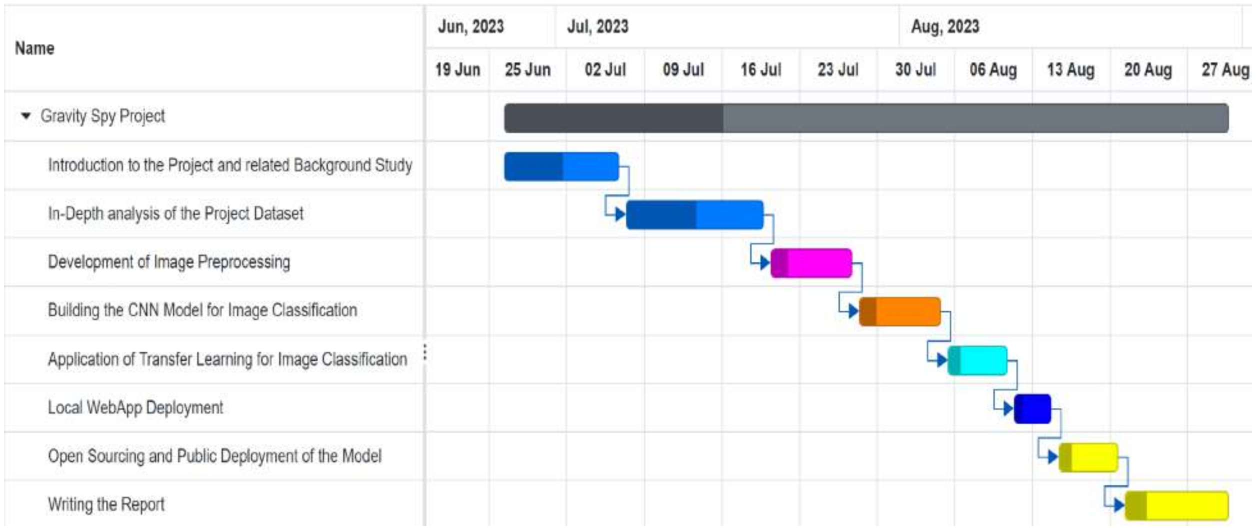
Our project was grounded in a clear objective: to create a model that could deliver accuracy, precision, reliability as well as accessibility. The result of this endeavour is a well-crafted and finely tuned model, adept at categorizing an array of gravitational wave images and glitches into 22 distinct classes. This accomplishment has practical implications for advancing gravitational wave research and lays the foundation for further refinements. Upon reflection, the impact of our model becomes evident in the realm of cosmological exploration. Our thoughtful approach to addressing challenges associated with imbalanced datasets and overfitting underscores our methodical strategy.

However, the exploration of employing more expansive and intricate models, accompanied by extended training cycles, emerges as a promising avenue for potential performance enhancements. The notion of data augmentation emerges as an intriguing prospect, albeit one that demands careful consideration given the unique intricacies of our dataset. The adjustment of class weights and the exploration of alternative pre-trained models offer avenues for enhancing precision and bolstering the model's utility. As we conclude this phase, it's clear that our model is more than just a culminating achievement for us; it represents a stepping stone that symbolizes the ongoing quest to decode the universe's mysteries. Our model serves as a testament to the iterative and collaborative nature of scientific inquiry, inviting future collaborations to further unravel the challenges posed by the cosmos that lie ahead in our enigmatic spacetime.

6. REFERENCES

1. Gravity Spy (Gravitational waves) dataset
2. LIGO, Caltech
3. Gravity Spy Zooniverse project
4. Gravity Spy: Integrating Advanced LIGO DetectorCharacterization, Machine Learning, and CitizenScience
5. Rob Harrand's Model code(Kaggle)
6. Baris Dincer's Model code(Kaggle)
7. Al Mahmud Al Mamun's Model code(Kaggle)
8. VGG16 Model Architecture
9. XCeption Model Architecture
10. ResNet50 Model Architecture

7. PROJECT TIMELINE (GANTT CHART)


Name
Jun, 2023
Jul, 2023
Aug, 2023
19 Jun
25 Jun
02 Jul
09 Jul
16 Jul
23 Jul
30 Jul
06 Aug
13 Aug
20 Aug
27 Aug
Gravity Spy Project
Introduction to the Project and related Background Study
In-Depth analysis of the Project Dataset
Development of Image Preprocessing
Building the CNN Model for Image Classification
Application of Transfer Learning for Image Classification
Local WebApp Deployment
Open Sourcing and Public Deployment of the Model
Writing the Report