

Problem 3 Report

Pulsars are fascinating celestial objects—highly magnetized, rotating neutron stars that emit beams of electromagnetic radiation from their magnetic poles. When these beams are aligned with Earth, they appear as a series of pulses, making them detectable. However, identifying pulsar signals can be quite challenging since they are often buried in noise caused by man-made radio frequency interference (RFI).

The goal of this project is to create a classifier capable of distinguishing genuine pulsar signals from RFI, using both numerical features and image representations of pulsar signals.

Dataset Description We utilized two datasets:

- **Numerical Features Dataset:** Contains eight numerical features extracted from pulsar signals. The aim is to build a binary classifier that distinguishes between pulsar signals and RFI.
- **Image Dataset:** Consists of 32x32 RGB images derived from the raw signal data. This dataset is ideal for building a Convolutional Neural Network (CNN) model to classify pulsars.

Data Preprocessing **Data Cleaning** To prepare the data for modeling, we:

- Removed rows with missing values to maintain data integrity.
- Dropped irrelevant columns, such as "Unnamed: 0," which did not contribute to the classification task.

Feature Scaling (Numerical Dataset) Since models like SVM are sensitive to feature scales, the numerical data was standardized using StandardScaler, ensuring all features had zero mean and unit variance.

Image Preprocessing (Image Dataset) To prepare the image data for CNN training, the following steps were applied:

- **Resizing:** Standardized the image size to ensure uniformity.
- **Normalization:** Scaled pixel values to the [0,1] range.
- **Data Augmentation:** Applied techniques like horizontal flipping, random rotations, and color jittering to diversify the dataset and reduce overfitting.

Data Splitting

- **Training Set:** 70% of the data.
- **Test Set:** 30% reserved for evaluating model performance.


Model Development **Numerical Features Models** We experimented with the following models:

- **Support Vector Machine (SVM):** Effective for high-dimensional datasets.
- **Random Forest:** An ensemble method using multiple decision trees.
- **Voting Classifier:** Combined predictions from both SVM and Random Forest for enhanced performance.

Hyperparameter Tuning To maximize model performance, we used RandomizedSearchCV for tuning hyperparameters on both SVM and Random Forest models.

Image Features Model For the image dataset, a Convolutional Neural Network (CNN) was developed with the following components:

- **Conv2D Layers:** Extracted spatial patterns from the images.
- **MaxPooling:** Reduced the spatial dimensions for efficiency.
- **Fully Connected Layers:** Processed extracted features for classification.
- **Activation Functions:** ReLU for hidden layers and softmax for the output layer.



Some additional details regarding Image model:

Preprocessing:

The images were normalized to a $[-1, 1]$ range after being converted to tensors using `ToTensor()` and `Normalize()`. Data augmentation techniques, including random horizontal flipping, slight rotations, color jittering (brightness and contrast adjustments), and random cropping with padding, were applied to increase dataset variability and reduce overfitting. Weighted Random Sampling was utilized to address class imbalance, ensuring underrepresented classes were adequately sampled during training.

Architecture:

The CNN included four convolutional layers: The first three used increasing filter sizes (16, 32, and 64) to extract spatial features from the images. A fourth convolutional layer with 128 filters was added to enhance feature extraction. Max-pooling layers followed each convolutional block to reduce spatial dimensions while preserving critical information. Fully connected layers with dimensions of 512, 256, and 128 were used to process extracted features, and the output layer predicted logits for the two classes (pulsar and non-pulsar). ReLU activations were applied after each convolutional and fully connected layer to introduce non-linearity.

Optimization:

The Adam optimizer was used with an initial learning rate of 0.001 for efficient gradient-based optimization. A learning rate scheduler dynamically adjusted the learning rate when the validation loss plateaued, helping to fine-tune the model. A weighted cross-entropy loss function, with weights inversely proportional to class frequencies, was employed to mitigate the impact of class imbalance. Dropout was implemented in fully connected layers to improve generalization and reduce overfitting.

Model Evaluation **Performance Metrics** The models were evaluated using:

- **Accuracy:** Overall correctness of predictions.
- **Precision:** Proportion of positive predictions that were correct.
- **Recall:** Proportion of actual positives correctly identified.
- **F1-Score:** Balance between precision and recall.
- **Confusion Matrix:** To examine false positives and false negatives.

Results Numerical Dataset Models:

- **SVM:**
 - Accuracy: 98.16%
 - Precision (Pulsar): 0.92
 - Recall (Pulsar): 0.86
 - F1-Score (Pulsar): 0.89
- **Random Forest:**
 - Accuracy: 97.75%
 - Precision (Pulsar): 0.85
 - Recall (Pulsar): 0.89
 - F1-Score (Pulsar): 0.87
- **Neural Network:**
 - Accuracy: 97.81%
 - Precision (Pulsar): 0.88
 - Recall (Pulsar): 0.85
 - F1-Score (Pulsar): 0.87
- **Voting Classifier:**
 - Accuracy: 98.16%
 - Precision (Pulsar): 0.92
 - Recall (Pulsar): 0.86
 - F1-Score (Pulsar): 0.89

Image Dataset Model (CNN):

- **CNN:**
 - Accuracy: 98.65%
 - Precision (Pulsar): 0.98
 - Recall (Pulsar): 0.98
 - F1-Score (Pulsar): 0.98

Comparison and Discussion Model Comparison:

- **Numerical Dataset:** The Voting Classifier achieved the highest accuracy (98.16%) and maintained a balance between precision and recall.
- **Image Dataset (CNN):** The CNN outperformed the numerical models, achieving an impressive 98.65% accuracy with strong precision and recall balance.

False Positives vs. False Negatives: Minimizing false positives (misclassifying RFI as pulsars) while maximizing true positives (correctly identifying pulsars) is critical for this task. The Voting Classifier balanced precision and recall effectively, while the CNN showed superior overall performance.

Conclusion **Summary of Findings:**

- The **Voting Classifier** was the top performer for numerical data, achieving high accuracy with a balance between precision and recall.
- The **CNN model** achieved the highest accuracy overall (98.65%), effectively capturing spatial patterns from the pulsar image dataset.