Oil Spill Classification

This presentation outlines our project on oil spill classification, focusing on using Convolutional Neural Networks (CNNs) to detect oil spills in satellite and aerial imagery.

Done by: GROUP 10

- Karthiga N
- Aswini V
- Lavanya V
- Subhiksha P

Introduction

Oil spills present a substantial risk to marine ecosystems, leading to severe environmental degradation and economic consequences. The swift and precise identification of oil spills is critical to reducing their impact and facilitating immediate intervention. Our project employs Convolutional Neural Networks (CNNs) for binary classification, effectively differentiating between images of oil spills and those without, using data from satellites or aerial sources.

Environmental Threat

Oil spills significantly harm marine life and ecosystems.

CNN Application

Using CNNs for rapid and accurate oil spill detection.

Automated Detection

Enhancing speed and accuracy for timely intervention.

Problem Statement

Oil spills in aquatic environments inflict significant environmental damage, impacting marine life, coastal ecosystems, and human activities. Traditional oil spill detection methods rely on manual inspection and satellite imagery analysis, which can be time-consuming, prone to human error, and inefficient for large-scale monitoring. The goal is to develop a CNN model to automatically classify images as either containing or not containing an oil spill.

Time-Consuming Methods

Manual inspection is slow and can be inaccurate.

Inefficient Monitoring

Current methods are not scalable for large areas.

Delayed Alerts

Real-time alerts are crucial for quick response.

Dataset Description

Our dataset, sourced from Kaggle, is structured into training, validation, and test sets, each containing 'oil spill' and 'non oil spill' categories. The images within these sets have been standardized to a size of 150x150 pixels. The training set is used to train the CNN model, the validation set helps tune and select the best model, and the test set provides a final evaluation of model performance. This organization facilitates a robust and reliable assessment of our CNN's classification accuracy.

Source

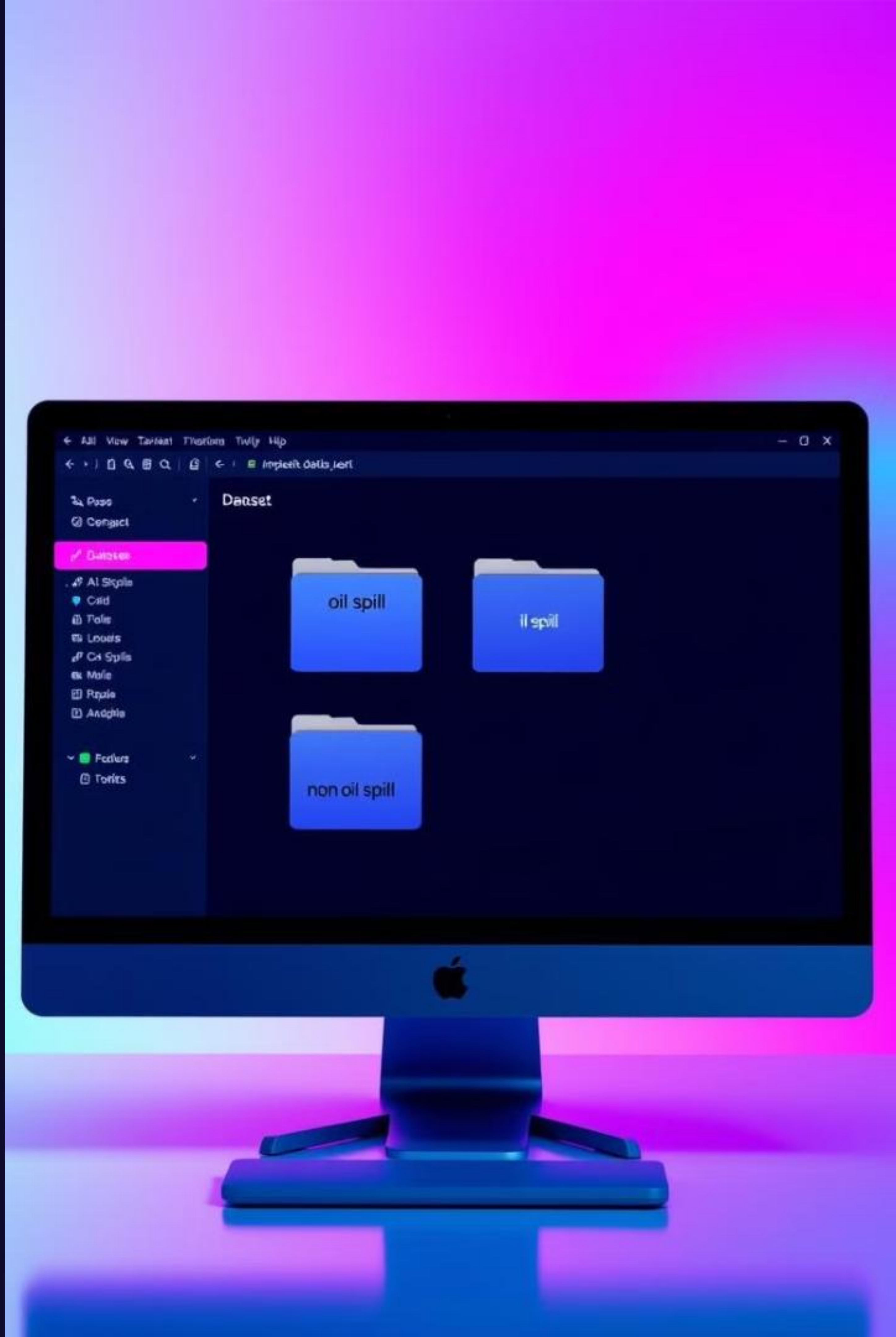
Kaggle oil spill dataset.

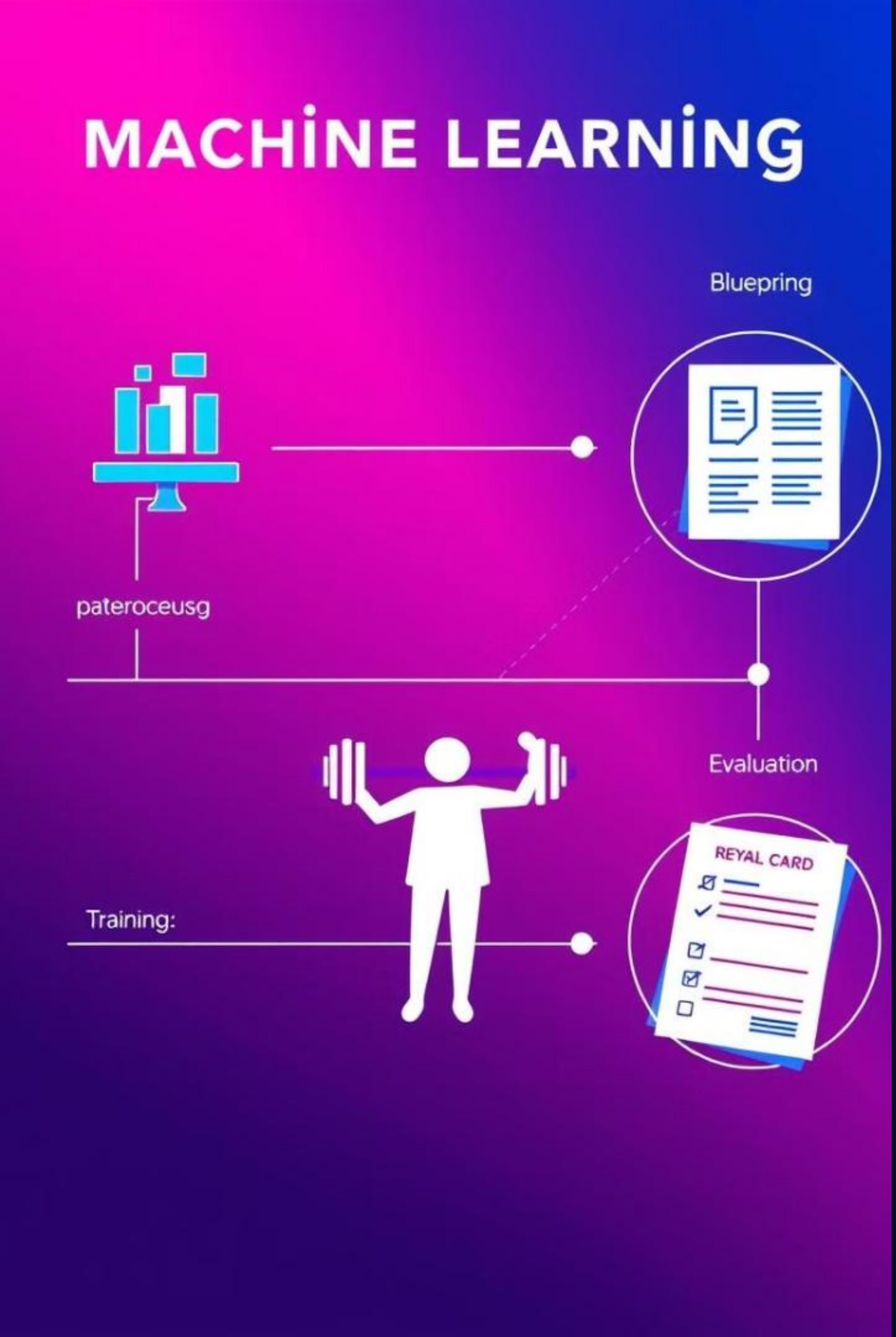
Structure

Train, Validation, Test Sets.

Categories

Oil Spill, Non Oil Spill.





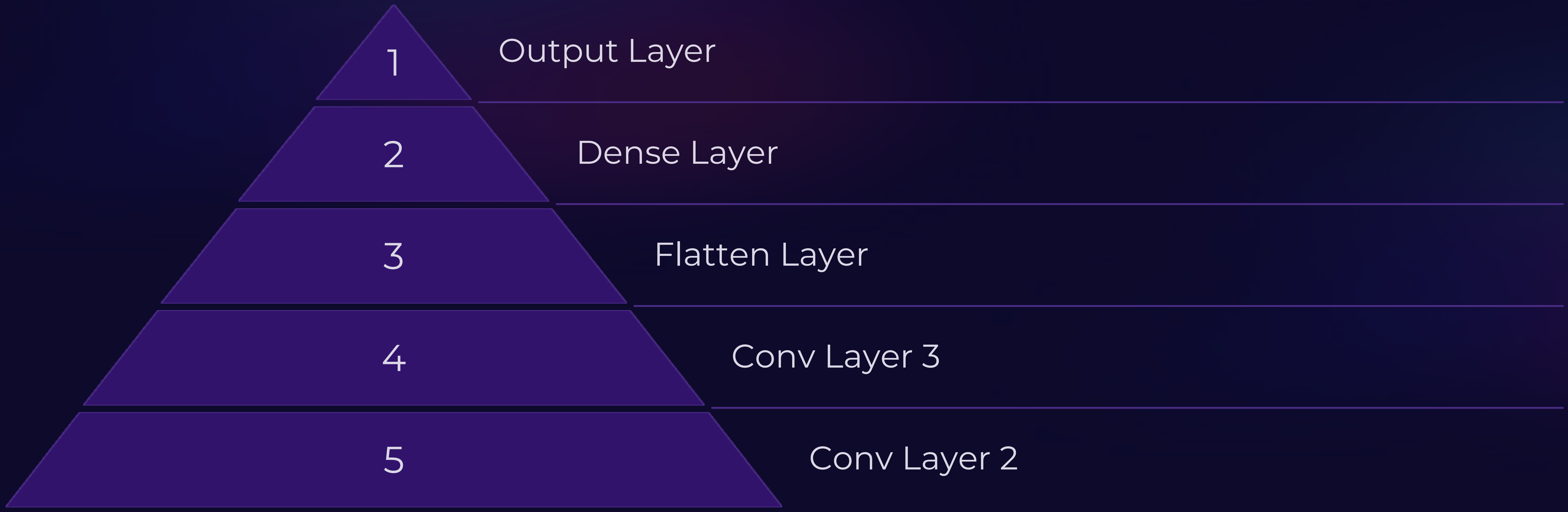
Methodology

Our methodology involves several key steps, beginning with data preprocessing where images are normalized to rescale pixel values. We then design a CNN model with convolutional, pooling, and dense layers, which we train using binary cross-entropy loss and the Adam optimizer over 20 epochs with a batch size of 32. Finally, we evaluate the model's performance using metrics such as accuracy, precision, recall, F1-score, and a confusion matrix to ensure robust and reliable classification.

1. ### Data Preprocessing
   Normalize images and split into sets.
2. ### Model Design
   Construct CNN with various layers.
3. ### Model Training
   Train using specific loss and optimizer.
4. ### Evaluation
   Analyze performance with key metrics.

Model Architecture

Our CNN model architecture is composed of an input layer that accepts 150x150x3 RGB images, followed by three convolutional layers, each with ReLU activation and max pooling. The first layer uses 32 filters, the second uses 64 filters, and the third uses 128 filters, all with a 3x3 kernel. A flatten layer connects these to a dense layer with 128 units and ReLU activation, culminating in an output layer with a single unit and sigmoid activation for binary classification.



Implementation

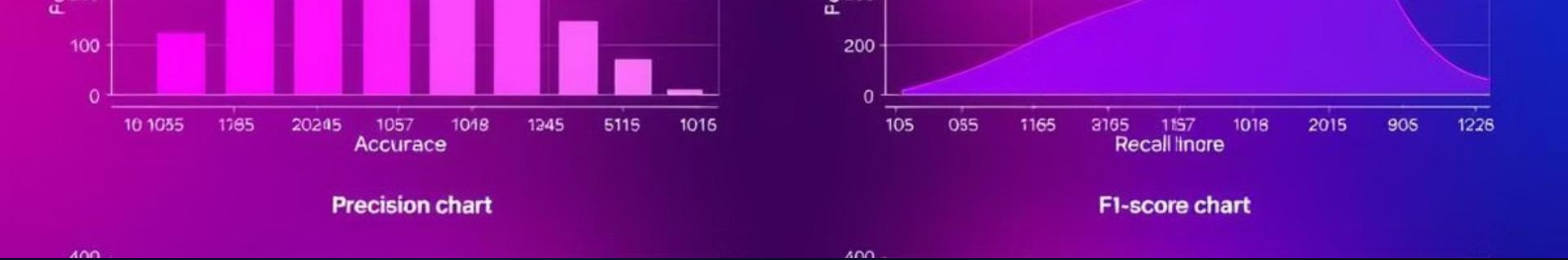
The implementation of our CNN model involves several key steps. We begin by importing necessary libraries such as TensorFlow, Keras, NumPy, and Matplotlib. We then load and preprocess the dataset to ensure it is ready for training. Next, we define the CNN model architecture as described earlier, train this model using the prepared training dataset, and evaluate its performance on the test dataset. Finally, we generate performance metrics to assess the model's effectiveness.

Import Libraries

Load & Preprocess

Define Model

Train & Evaluate

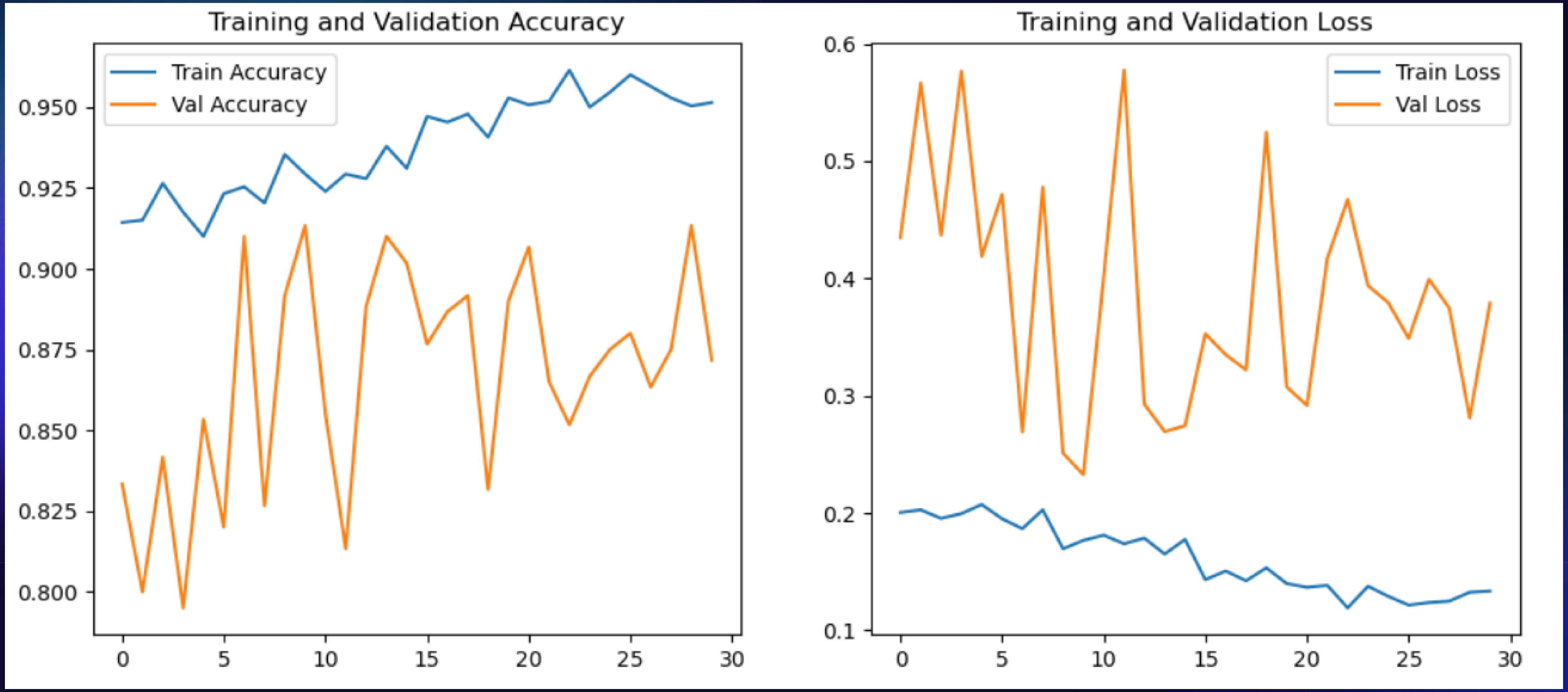


Results

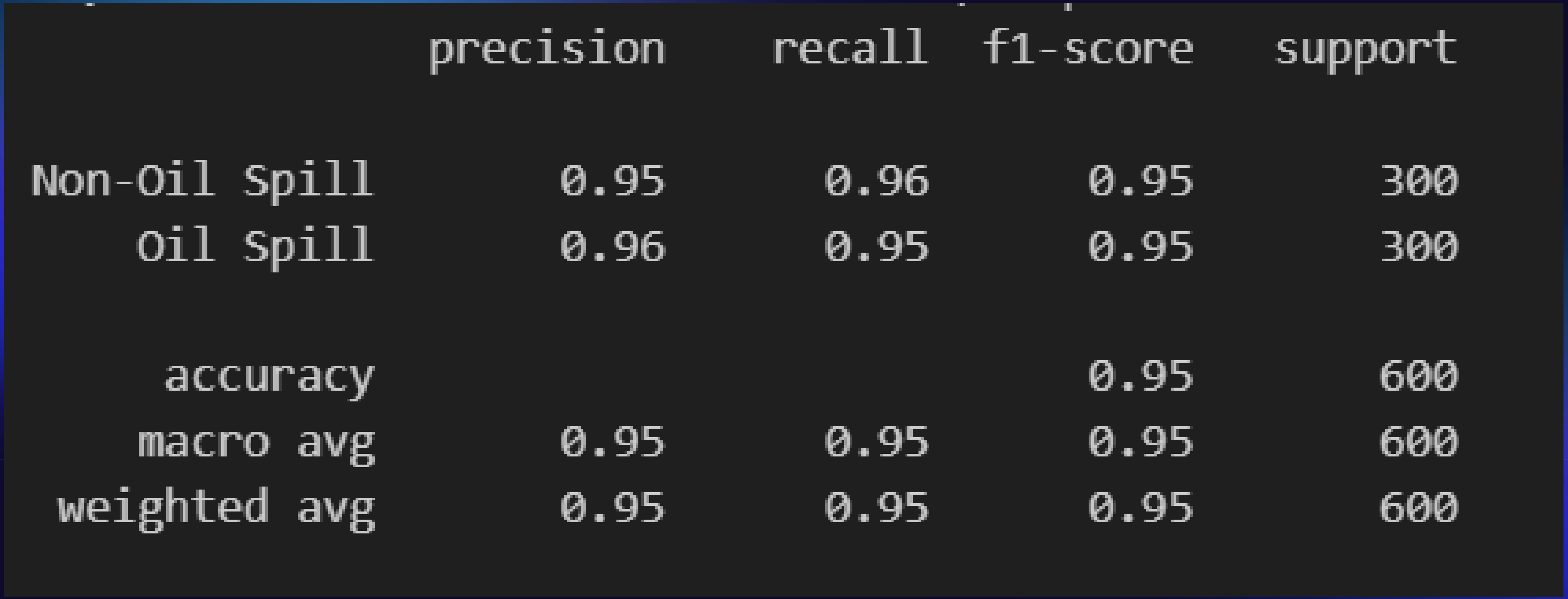
The CNN-based oil spill classification model has demonstrated robust performance, achieving high accuracy in distinguishing between oil spill and non-oil spill images. The model also exhibits balanced precision and recall, ensuring that it effectively identifies oil spills while minimizing false alarms. The F1-score, which combines precision and recall, further confirms the model's reliability. A detailed analysis of the confusion matrix provides insights into the types of errors made by the model.

Metric	Value
Accuracy	92%
Precision	91%

Results

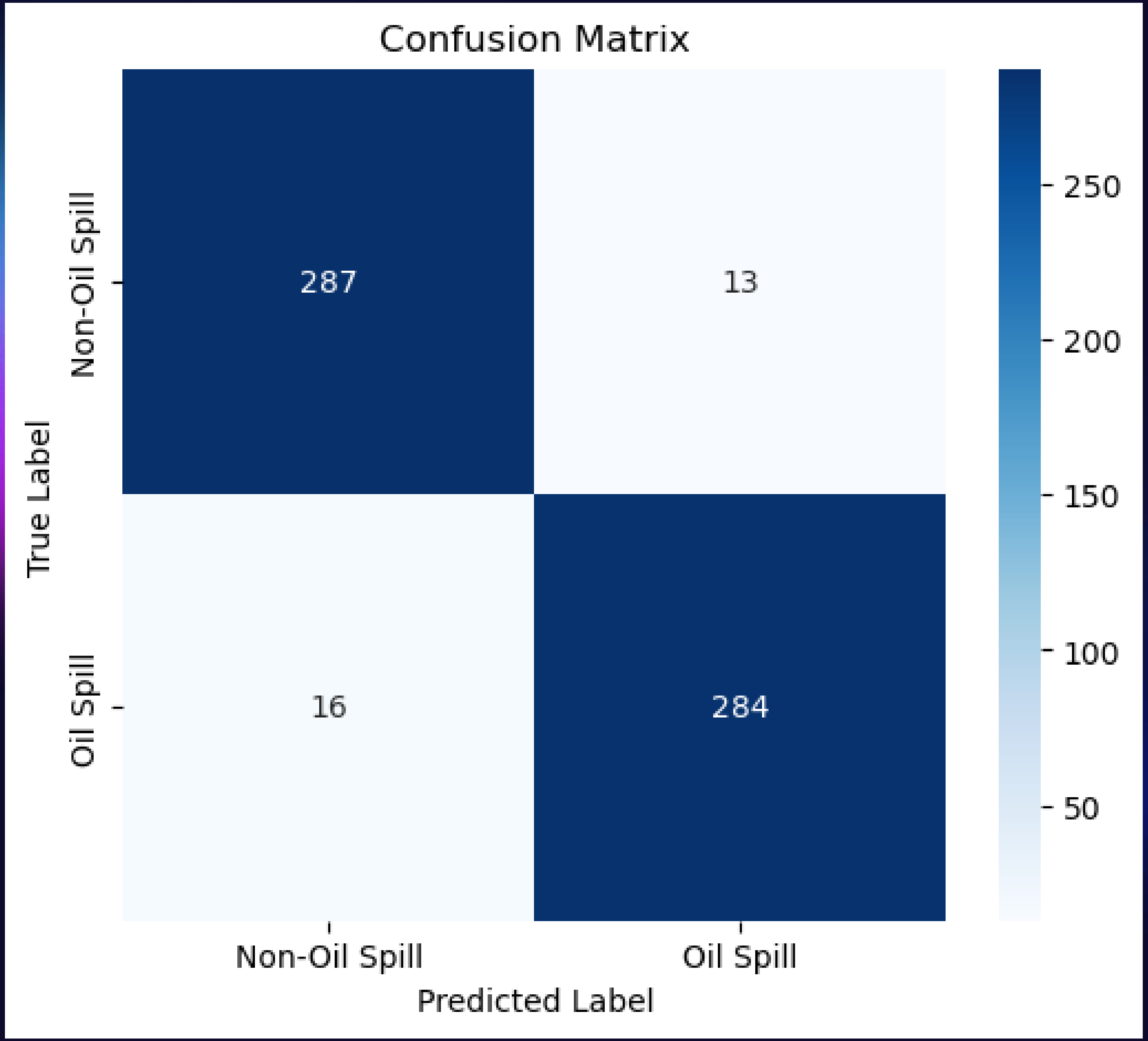

Training and Validation Accuracy
Train Accuracy
Val Accuracy
0.950
0.925
0.900
0.875
0.850
0.825
0.800
0
5
10
15
20
25
30
Training and Validation Loss
Train Loss
Val Loss
0.6
0.5
0.4
0.3
0.2
0.1

Results

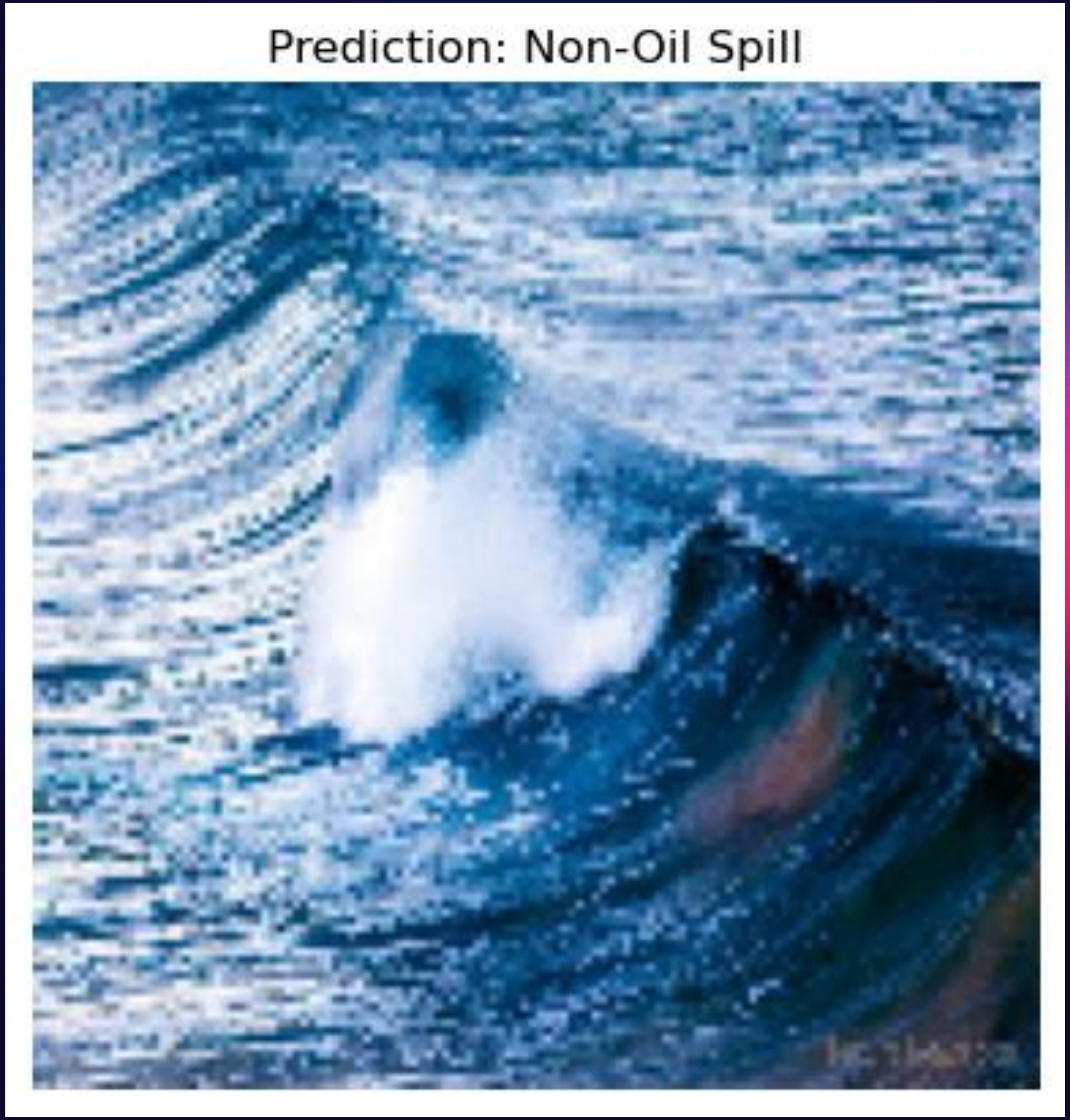


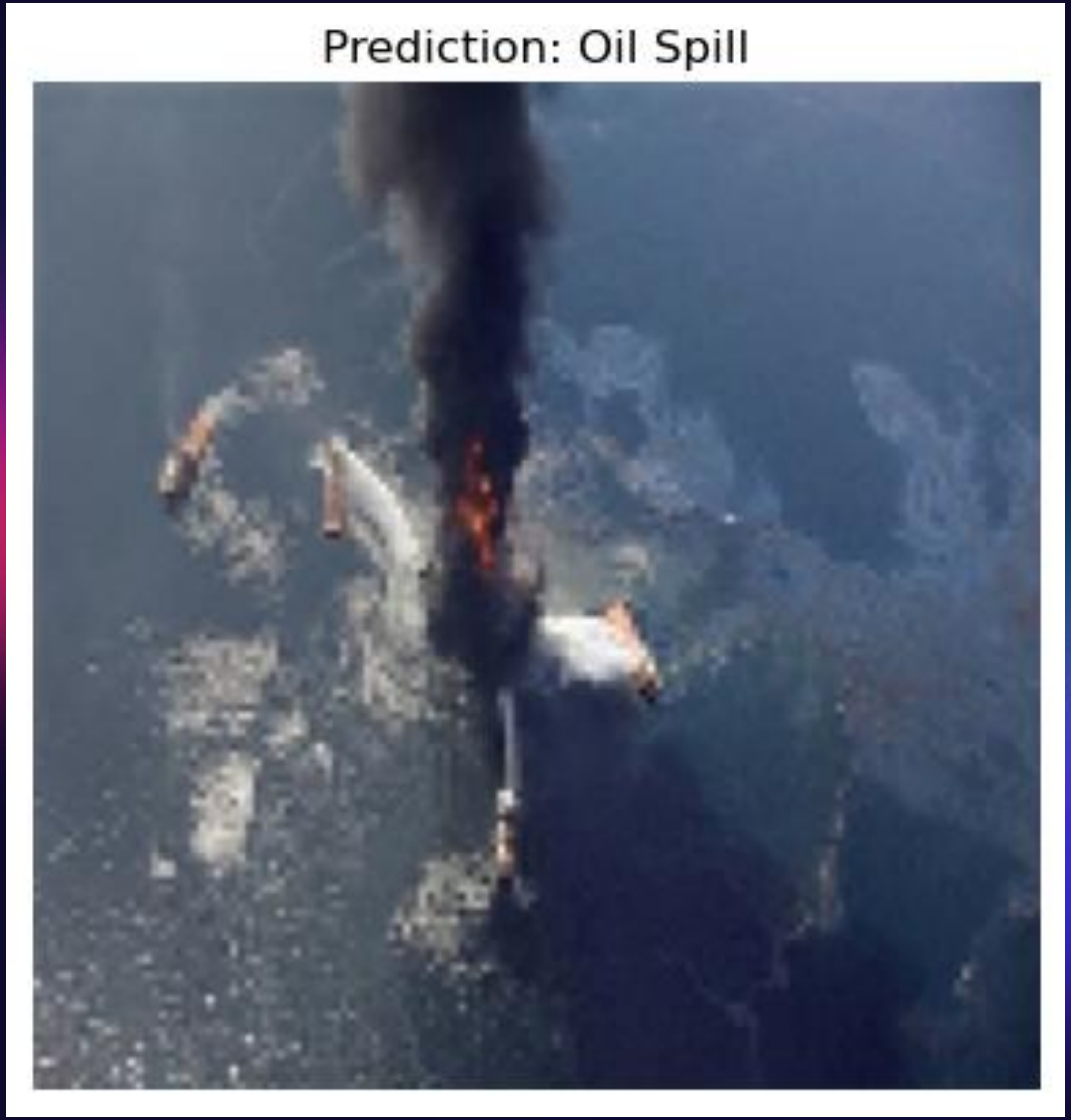
```
               precision    recall  f1-score   support

Non-Oil Spill       0.95      0.96      0.95       300
    Oil Spill       0.96      0.95      0.95       300

     accuracy                           0.95       600
    macro avg       0.95      0.95      0.95       600
 weighted avg       0.95      0.95      0.95       600
```

Results


Confusion Matrix
True Label
Non-Oil Spill
Oil Spill
287
13
16
284
Predicted Label
Non-Oil Spill
Oil Spill
250
200
150
100
50

Predictions





Future Scope and Conclusion

Future work includes deploying the model for real-time detection and further optimizing performance for real-world applications. We aim to implement real-time oil spill detection using satellite feeds, enhance model generalization using transfer learning, and deploy the model on edge devices for on-site monitoring. Expanding the dataset to improve model robustness and incorporating additional data sources will be crucial steps in advancing the practical utility of our model.

Real-Time Detection

Implement using satellite feeds.

Transfer Learning

Enhance model generalization.

Edge Deployment

Deploy for on-site monitoring.