**Artificial Intelligence Assignment**

**BlueTide**

**SUBMITTED BY:**

Chithmi Dilaksha

Table of Contents

[CHAPTER 01 Introduction 4](#_Toc200448759)

[1.1 Introduction to BlueTide 4](#_Toc200448760)

[1.2 Problem Statement 4](#_Toc200448761)

[1.3 Machine Learning Solves the Problems 5](#_Toc200448762)

[1.4 Brief Plan and Justification 5](#_Toc200448763)

[1.5 Objectives 6](#_Toc200448764)

[CHAPTER 02: Methodology 7](#_Toc200448765)

[2.1 Dataset Sourcing & Evaluation 7](#_Toc200448766)

[2.2 Models 8](#_Toc200448767)

[2.3 Data Preprocessing 9](#_Toc200448768)

[2.3.1 Dataset Loading 9](#_Toc200448769)

[2.3.2 Data Cleaning 9](#_Toc200448770)

[2.3.3 Image Preprocessing 12](#_Toc200448771)

[2.4 Model Building and Training 13](#_Toc200448772)

[2.4.1 Training Strategy and Hyperparameters 14](#_Toc200448773)

[CHAPTER 03: Result 16](#_Toc200448774)

[3.1 Result Analysis 16](#_Toc200448775)

[3.2 Evaluation Metrics Description 18](#_Toc200448776)

[3.3 Evaluation Procedure using ResNet50 & EfficientNetB7 20](#_Toc200448777)

[3.4 Testing 21](#_Toc200448778)

[3.5 Deployment 23](#_Toc200448779)

[3.5.1 Web Interface 23](#_Toc200448780)

[3.5.2 Backend code 26](#_Toc200448781)

[3.5.3 Tools & Frameworks Used 27](#_Toc200448782)

[CHAPTER 04: Discussion 28](#_Toc200448783)

[CHAPTER 05 Conclusion 29](#_Toc200448784)

[References 30](#_Toc200448785)

**Table of Figures**

[Figure 1 - Sea Animal Classification Dataset 7](#_Toc200448811)

[Figure 2 - Freediving Dataset 7](#_Toc200448812)

[Figure 3 - Shipwreck Dataset 8](#_Toc200448813)

[Figure 4 - Dataset Loading in K\_Means\_Clustering 9](#_Toc200448814)

[Figure 5 - Dataset Loading in Random\_Forest\_Classifier 9](#_Toc200448815)

[Figure 6 - Dataset Loading ResNet50 9](#_Toc200448816)

[Figure 7 - Dataset Loading EfficientNetB7 9](#_Toc200448817)

[Figure 8 - Data Cleaning 9](#_Toc200448818)

[Figure 9 - Drop Missing Values 10](#_Toc200448819)

[Figure 10 - Remove Outliers 10](#_Toc200448820)

[Figure 11 - Drop Unnecessary Columns 10](#_Toc200448821)

[Figure 12 – K-Means Clusteringlibraries 10](#_Toc200448822)

[Figure 13 - ResNet50 libraries 10](#_Toc200448823)

[Figure 14 - EfficientNetB7 libraries 11](#_Toc200448824)

[Figure 15 – Random Forest Classifierlibraries 11](#_Toc200448825)

[Figure 16 - EfficientNetB7Data Augmentation 12](#_Toc200448826)

[Figure 17 - ResNet50 Data Augmentation 12](#_Toc200448827)

[Figure 18 - ResNet50 Model Architecture 13](#_Toc200448828)

[Figure 19 - EfficientNetB7 Model Architecture 13](#_Toc200448829)

[Figure 20 - StandardScaler 14](#_Toc200448830)

[Figure 21 - SMOTE Technique 14](#_Toc200448831)

[Figure 22 - GridSearchCV Optimizer 14](#_Toc200448832)

[Figure 23 - EfficientNetB7Model Strategy 1 15](#_Toc200448833)

[Figure 24 - EfficientNetB7Model Strategy 2 15](#_Toc200448834)

[Figure 25 - ResNet50 Model Strategy 1 15](#_Toc200448835)

[Figure 26 - ResNet50 Model Strategy 2 15](#_Toc200448836)

[Figure 27 - Results of ResNet50 16](#_Toc200448837)

[Figure 28 - Result of Efficient-NetB7 16](#_Toc200448838)

[Figure 29 - Result of Random Forest Classifier 17](#_Toc200448839)

[Figure 30 - Number of Dive Sites Per Cluster 17](#_Toc200448840)

[Figure 31 - K-Means Clustering of Dive Sites By Depth 17](#_Toc200448841)

[Figure 32 - Confusion Metrix 19](#_Toc200448842)

[Figure 33 - EfficientNetB7 Epochs 20 20](#_Toc200448843)

[Figure 34 - ResNet50 Epochs 20 21](#_Toc200448844)

[Figure 35 - Image Classification Model Testing 21](#_Toc200448845)

[Figure 36 - Diving Score Predictor Testing 22](#_Toc200448846)

[Figure 37 - Diving Experience Level Testing 22](#_Toc200448847)

[Figure 38 – Save Model 23](#_Toc200448848)

[Figure 39 - First Interface 24](#_Toc200448849)

[Figure 40 - Sea Creature Identifier 24](#_Toc200448850)

[Figure 41 - Sea Diving Score Predictor 25](#_Toc200448851)

[Figure 42 - Sea Diving Area Predictor 25](#_Toc200448852)

[Figure 43 - Backend Code 1 26](#_Toc200448853)

[Figure 44 - Backend Code 2 26](#_Toc200448854)

# CHAPTER 01 Introduction

# Introduction to BlueTide

The **BlueTide** project uses deep learning and machine learning models. There are **EfficientNetB7** and **ResNet50** to identify sea animals in blurry images and provides detailed descriptions of marine animals through the **Gemini API** and **K-Means Clustering** is used to classify diving sites by experience level. **Random forest classifier** is also used to predict high, medium and low diving scores using multiple datasets. It uses artificial intelligence, machine learning and deep learning to improve diver safety and provide valuable insights into marine biodiversity and diving performance, making it a useful tool for researchers, divers and marine enthusiasts.

# Problem Statement

* Identification of sea animals is slow, requires expert knowledge, and is often inaccurate especially in blurry or low-light underwater images.
* There is no standardized, intelligent system to rate dive sites by difficulty (e.g., beginner, intermediate, pro), which can lead to safety risks for divers.
* Freediving records vary by country, athlete, gender, event, discipline, and year. Manually evaluating whether a performance is high, medium, or low is subjective and inconsistent.
* Large amounts of data on sea animals, diving areas, and shipwrecks exist, but are not being analyzed to uncover trends, insights, or safety guidelines.
* Divers of different skill levels receive generic site suggestions, increasing the chance of accidents or poor dive experiences.
* Marine biologists lack tools that can automatically classify marine animals from underwater images to support fast and large-scale research.
* Without automated image-based identification and data logging, monitoring how species populations change over time is slow and labor-intensive.
* Despite advances in AI, most ocean-related software tools don’t use machine learning to enhance exploration, safety, or biodiversity tracking.

# Machine Learning Solves the Problems

* Use CNNs like **EfficientNetB7** and **ResNet50** to classify marine animals from blurry or unclear underwater images.
* Use **K-Means Clustering** to group dive sites based on depth, accessibility, and risk into Beginner, Intermediate, and Pro levels.
* Apply a **Random Forest Classifier** to predict Low, Medium, or High-performance categories based on historical diving data.
* ML algorithms can uncover **hidden patterns** in large datasets (e.g., species distribution, dive site trends) that would be missed manually.
* ML can offer custom recommendations to divers based on skill level, dive history, and site difficulty clusters.
* Deep learning models allow **automated image analysis** at scale, speeding up research on species classification and biodiversity.
* Unsupervised ML (K-Means) provides an **objective method** to categorize shipwrecks by difficulty and age without labeled data.
* Automated ML-based logging of species presence over time supports **trend analysis and conservation planning**.
* By integrating ML into exploration tools, BlueTide bridges the **gap between technology and marine field research**.

# Brief Plan and Justification

1. **Define the Problem:** Understand what specific marine data issue to solve (e.g., species classification, performance grouping).
2. **Collect Data:** Find relevant datasets suitable for the problem.
3. **Prepare Data:** Clean, preprocess, and explore the data to understand it better.
4. **Choose Models:** Use supervised models (like Random Forest or CNN) for labeled data, and clustering methods (like K-Means) for unlabeled data.
5. **Train and Test:** Train the models and evaluate their accuracy using standard metrics.
6. **Deploy:** Build an automated system to apply these models for fast and consistent marine data analysis.

**Justification:**  
Machine learning provides an effective, scalable, and reliable way to handle complex marine data problems, improving accuracy and reducing manual effort.

# Objectives

**Objectives of ResNet-50 and EfficientNetB7**

* Develop a deep learning-based classification model for sea animals.
* Achieve high accuracy using models.
* Preprocess the dataset effectively for better model performance.
* Evaluate the model using standard performance metrics.
* Deploy the model as a user-friendly application.

**Objectives of Random Forest Classifier**

* Divide the dataset into Low, Medium, and High priority.
* Recognize significant features and target variables.
* Preprocess the data for effective model training.
* Transform the data into a suitable format for modeling.
* Balance the dataset for improved prediction accuracy.
* Train the model to classify records.
* Evaluate the model trained with the traditional accuracy metrics.
* Deploy the model as a user-friendly application.

**Objectives of K Means Clustering**

* Split the dataset into distinct clusters based on similarities.
* Determine the optimal number of clusters for effective grouping.
* Preprocess and normalize the dataset for effective clustering operation.
* Apply this for pattern and structure recognition in the data.
* Visualize and comprehend cluster distributions for insight.
* Enable more effective data-driven decision-making by grouping similar records together.
* Compare clustering results with other unsupervised learning techniques

# CHAPTER 02: Methodology

# Dataset Sourcing & Evaluation

**Dataset 1:** [Sea Animal Classification](https://www.kaggle.com/datasets/vencerlanz09/sea-animals-image-dataste)is sourced from Kaggle using EfficientNetB7 and ResNet50. This labeled image dataset contains hundreds of underwater photographs categorized into various marine species such as clams, crabs, dolphins, and more. It is particularly suitable for training and evaluating image classification models in marine biodiversity research, enabling automated identification of sea creatures for educational, scientific, or conservation purposes.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 1 - Sea Animal Classification Dataset

Dataset 2: [Freediving Performance Dataset](https://www.kaggle.com/datasets/igalbronshtein/freediving-aida-world-records-dataset) is sourced from Kaggle. This tabular dataset includes records of international freediving performances, with details such as discipline, gender, country, year, athlete names, record values, and event links. It is valuable for analyzing performance trends, comparing freediving disciplines, and creating visual storytelling tools in sports analytics and athlete tracking.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 2 - Freediving Dataset

Dataset 3: [Shipwreck Dataset](https://drive.google.com/file/d/1AF8pKRrvvpkkv0WMYDffdyEqerbCSFpn/) contains unlabeled information about shipwrecks, including features such as site names, depths, year of sinking, and geographic locations. It is especially useful for mapping dive sites, conducting clustering analysis, and exploring the historical distribution of marine wrecks around Sri Lankan coastal regions.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3 - Shipwreck Dataset

# 2.2 Models

**ResNet50 -** This is a deep CNN with residual connectivity, is used in BlueTide to efficiently classify marine animals. This significantly improves accuracy, reduces training time, and increases efficiency in marine biodiversity research.

**EfficientNetB7 -** This model scales from EfficientNetB0 to EfficientNetB7, with small increases in accuracy for bigger models. Also, this has more layers and parameters, and it helps in recognizing intricate textures, colors, and patterns of dataset images more efficiently.This used to Marine Image Classification.

**Random Forest Classifier -** In BlueTide, analyzes freediving records to predict performance categories (low, medium, high). By evaluating characteristics such as country, discipline, and year, it supports performance tracking and decision making in marine sports analytics.

**K Means Clustering** - This for clustering data points into clusters based on their similarities. In BlueTide, K-Means Clustering groups dive sites together using depth and position data. It predicts experience levels **beginner, intermediate, professional** based on-site characteristics like depth and district.

# Data Preprocessing

## Dataset Loading



Figure 4 - Dataset Loading in K\_Means\_Clustering



Figure 5 - Dataset Loading in Random\_Forest\_Classifier



Figure 6 - Dataset Loading ResNet50

A close-up of a logo

AI-generated content may be incorrect.

Figure 7 - Dataset Loading EfficientNetB7

## Data Cleaning

The initial dataset was examined for:

* **Missing values**: No missing values were found in the dataset.
* **Duplicate rows**: 1 duplicate record was identified and removed.
* **Incorrect entries**: Categorical values such as sex, cp, and that were checked for out-of-range or invalid values. No anomalies were detected.

**Data Cleaning in K-Means Clustering Model**

A screenshot of a computer code

AI-generated content may be incorrect.

Figure 8 - Data Cleaning

**Data Cleaning in Random Forest Classifier Model**

A close-up of a missing link

AI-generated content may be incorrect.

Figure 9 - Drop Missing Values

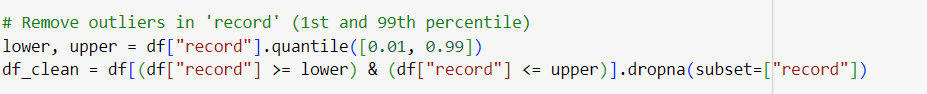


Figure 10 - Remove Outliers

A screenshot of a computer

AI-generated content may be incorrect.

Figure 11 - Drop Unnecessary Columns

**Common libraries and tools include:**

* TensorFlow and Keras - for building and training deep learning models.
* Pandas and numpy - for data manipulation.
* Matplotlib and seaborn - for data visualization.
* Google Collab - As a development environment for interactive, GPU-accelerated experiments.

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 12 – K-Means Clusteringlibraries

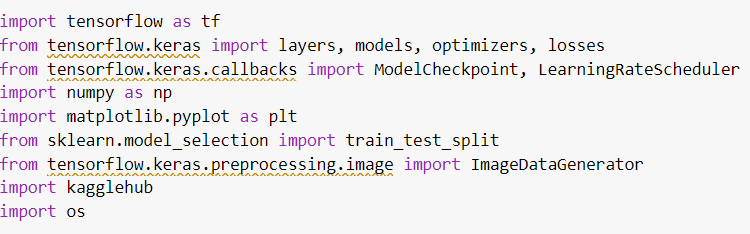


Figure 13 - ResNet50 libraries

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 14 - EfficientNetB7 libraries

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 15 – Random Forest Classifierlibraries

## Image Preprocessing

|  |  |  |
| --- | --- | --- |
| **Steps** | **ResNet-50** | **EfficientNetB7** |
| **Image Resizing** | 300x225 pixels | 224x224 pixels |
| **Data Augmentation** | Random horizontal flipping  Random rotation (15 degrees)  Color jittering (brightness, contrast, saturation adjustments) | Rescaling – Normalizes pixel values to [0,1] range  Random flip – flip images horizontally  Random rotation – rotate images by ±10%  Random Contrast – Adjusts contrast by ±10% |
| **Normalization** | Applied using mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225]. | Test Loss: 0.76889  Test Accuracy: 77.73% |
| **Train Test Split** | 70% Training Data  15% Validation Data  15% Test Data | 75% Training Data  25% Testing Data  20% Validation data |

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 16 - EfficientNetB7Data Augmentation

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 17 - ResNet50 Data Augmentation

# Model Building and Training

**Transfer Learning with ResNet-50**

ResNet-50 is pre-trained on ImageNet, a large-scale dataset containing 1,000 different object categories. However, in our BlueTide project, Dataset only have 23 specific sea animal categories. Since the default ResNet-50 model is designed to classify 1,000 categories,

A diagram of a diagram

AI-generated content may be incorrect.

Figure 18 - ResNet50 Model Architecture

**Transfer Learning with EfficientNetB7**

It is pre-trained on ImageNet, a large dataset with 1,000 object categories. In our BlueTide project, focus on just 23 specific sea animal categories. Since the default EfficientNet-B7 model is set up for 1,000 categories, this need to change the last layer of the network to fit our dataset.

A diagram of different colors

AI-generated content may be incorrect.

Figure 19 - EfficientNetB7 Model Architecture

## Training Strategy and Hyperparameters

**Model Strategy Comparison Table between** **K-Means Clustering & Random Forest Classifier**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **K-Means Clustering** | **Random Forest Classifier** |
| **Use Case** | Clustering dive sites by depth to group by diver experience level  (Beginner, Intermediate, Pro) | Classifying freediving records into performance categories (Low, Medium, High) |
| **Preprocessing** | Standardized using StandardScaler | One-hot encoding for categorical features + standard scaling for numerical features |
| **Special Techniques** | None | Applied **SMOTE** to handle class imbalance |
| **Model Parameters** | n\_clusters = 3 (Beginner, Intermediate, Pro) | Optimized with GridSearchCV and 5-fold cross-validation |

A screenshot of a computer code

AI-generated content may be incorrect.

Figure 20 - StandardScaler

A screen shot of a computer code

AI-generated content may be incorrect.

Figure 21 - SMOTE Technique

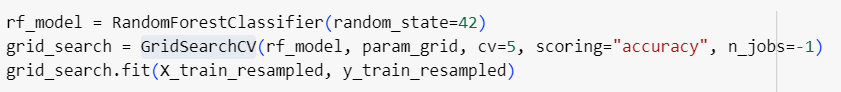


Figure 22 - GridSearchCV Optimizer

**Model Strategy Comparison Table between ResNet-50 & EfficientNetB7**

|  |  |  |
| --- | --- | --- |
| **Component** | **ResNet-50 Strategy** | **EfficientNetB7 Strategy** |
| **Base Model** | (include\_top=False, input\_shape=(224, 224, 3)) | (include\_top=False, input\_shape=(224, 224, 3)) |
| **Image Input Size** | 224 × 224 | 600 × 600 |
| **Data Augmentation** | Yes, using ImageDataGenerator | Yes, using ImageDataGenerator |
| **Optimizer** | Adam() | Adam() |
| **Loss Function** | SparseCategoricalCrossentropy | categorical\_crossentropy |
| **Evaluation Metric** | Accuracy | Accuracy |

A computer screen shot of a computer code

AI-generated content may be incorrect.

Figure 23 - **EfficientNetB7** Model Strategy 1

A close-up of a code

AI-generated content may be incorrect.

Figure 24 - **EfficientNetB7** Model Strategy 2



Figure 25 - ResNet50 Model Strategy 1

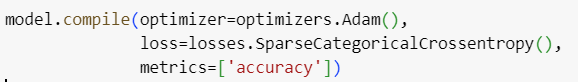


Figure 26 - ResNet50 Model Strategy 2

# CHAPTER 03: Result

# Result Analysis

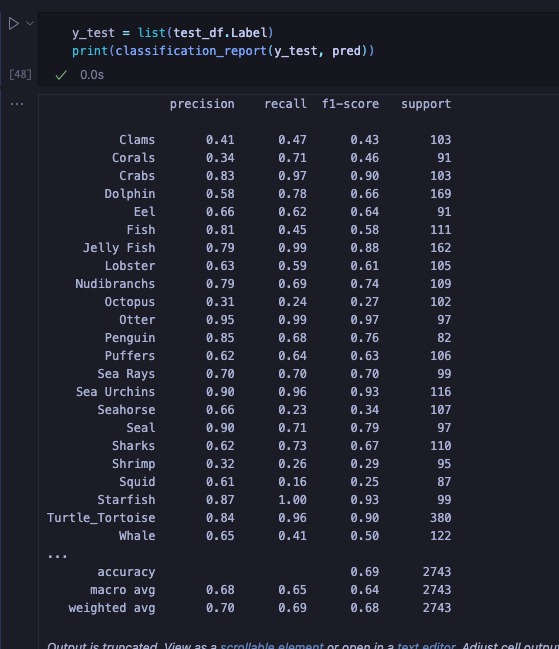
After training, both models were tested on a held-out validation/test dataset to measure generalization performance.

A screenshot of a computer

AI-generated content may be incorrect.

**ResNet-50:**

* Accuracy and loss were tracked across all epochs.
* The learning rate was dynamically adjusted using a **scheduler** to prevent overfitting and promote convergence.

Figure 27 - Results of ResNet50

**EfficientNetB7:**

* Similar evaluation using TensorFlow’s built-in methods.
* Visual inspection of predictions was also done using custom pred\_and\_plot() utility.

Figure 28 - Result of Efficient-NetB7

A screenshot of a computer screen

AI-generated content may be incorrect.**Random Forest Classifier**

* **Test Set Evaluation:**
  + The model was tested on the preprocessed and encoded test set.
  + Predictions were compared against ground truth using:
    - Accuracy
    - Precision, Recall, F1-score

Figure 29 - Result of Random Forest Classifier

* **Observations:**
  + The model achieved high accuracy.
* The class distribution after SMOTE helped balance predictions across Low, Medium, and High categories.
* Misclassifications may have occurred due to overlaps in record values between categories.

**K-Means Clustering**

**Cluster Assignments:**

* Clusters were assigned to each dive site and visualized using seaborn.

**Result:**

* A moderately high silhouette score indicates well-separated clusters.
* A graph of a number of diving sites

  AI-generated content may be incorrect.A graph of a number of dive sites per cluster

  AI-generated content may be incorrect.The model effectively grouped sites based on depth, useful for suggesting dive spots by skill level

Figure 30 - Number of Dive Sites Per Cluster

Figure 31 - K-Means Clustering of Dive Sites By Depth

# Evaluation Metrics Description

To evaluate the performance of the deep learning models—**ResNet-50** and **EfficientNetB7**—several standard metrics were used:

* **Accuracy**: Measures the percentage of correctly classified samples.
* **Loss**: Cross-entropy loss was used for classification tasks to evaluate model prediction error.
* **Precision, Recall, and F1-Score**: Provide detailed insight into model performance, especially for imbalanced datasets.
* **Confusion Matrix**: Visual tool to assess the number of true positives, false positives, true negatives, and false negatives.

These metrics help us understand how well the models generalize to unseen data and where may need further improvement.

**The performance of ResNet50 model was evaluated using:**

* Accuracy: Measures overall correct classifications.

**The performance of EfficientNetB7 model was evaluated using:**

**Classification Metrics:**

* Accuracy
* Precision, Recall, F1-Score
* AUC-ROC, PR Curve

**Loss Function:**

* Categorical Crossentropy (for multi-class)
* Binary Crossentropy (for binary classification)
* Training vs. Validation Loss

**Optimizer Selection:**

* Adam, RMSprop, or SGD with learning rate scheduling

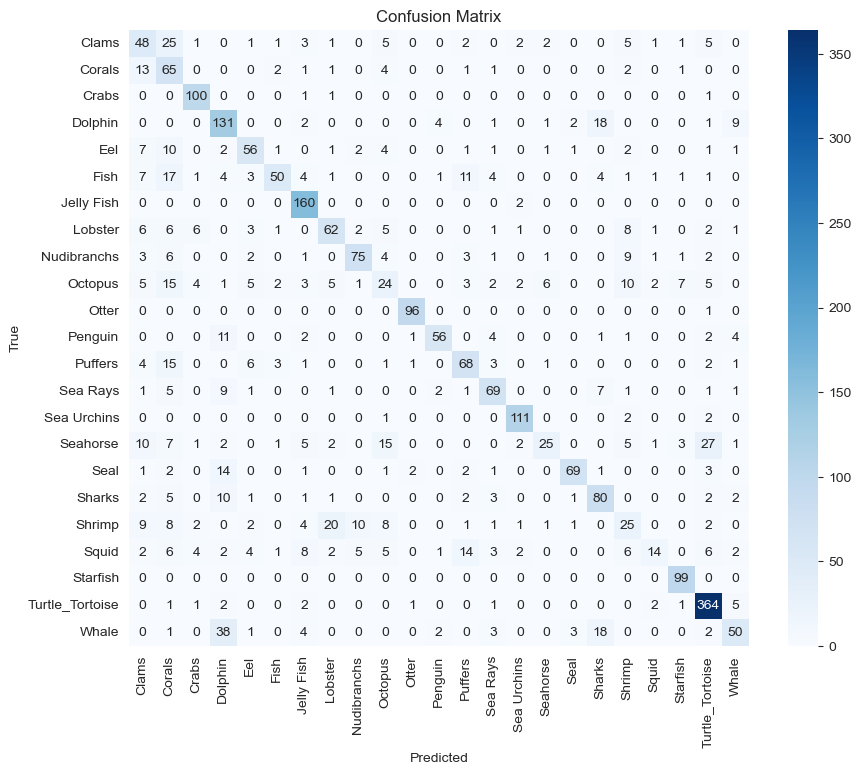


Figure 32 - Confusion Metrix

A confusion matrix shows how well a classification model is performed by comparing the actual and predicted labels in Efficient-NetB7 model. Correct predictions appear diagonally; Its higher values mean better accuracy. Misclassifications are off diagonal. For example, "dolphin" was correctly predicted 131 times but confused with "whale" 38 times, showing an overlap. The matrix helps identify strong and weak class predictions, guiding improvements in model accuracy and strategy.

# Evaluation Procedure using ResNet50 & EfficientNetB7

**ResNet50**

* The model was trained for 20 epochs using CrossEntropyLoss.
* A cosine annealing learning rate scheduler was implemented to adjust learning rates dynamically.
* The best model was saved based on the highest validation accuracy.

**EfficientNetB7**

* The model was trained for 20 epochs using categorical\_crossentropy.
* The optimizer used was Adam (learning rate of 0.00001).

**Model Performance Evaluation:**

* Using validation/test dataset
* Generating confusion matrix
* Computing classification report (precision, recall, F1-score)

**Visualization:**

* Training/Validation loss and accuracy plots
* ROC Curve, Precision-Recall Curve

A screenshot of a computer

AI-generated content may be incorrect.

Figure 33 - EfficientNetB7 Epochs 20

A close up of a text

AI-generated content may be incorrect.

Figure 34 - ResNet50 Epochs 20

# Testing

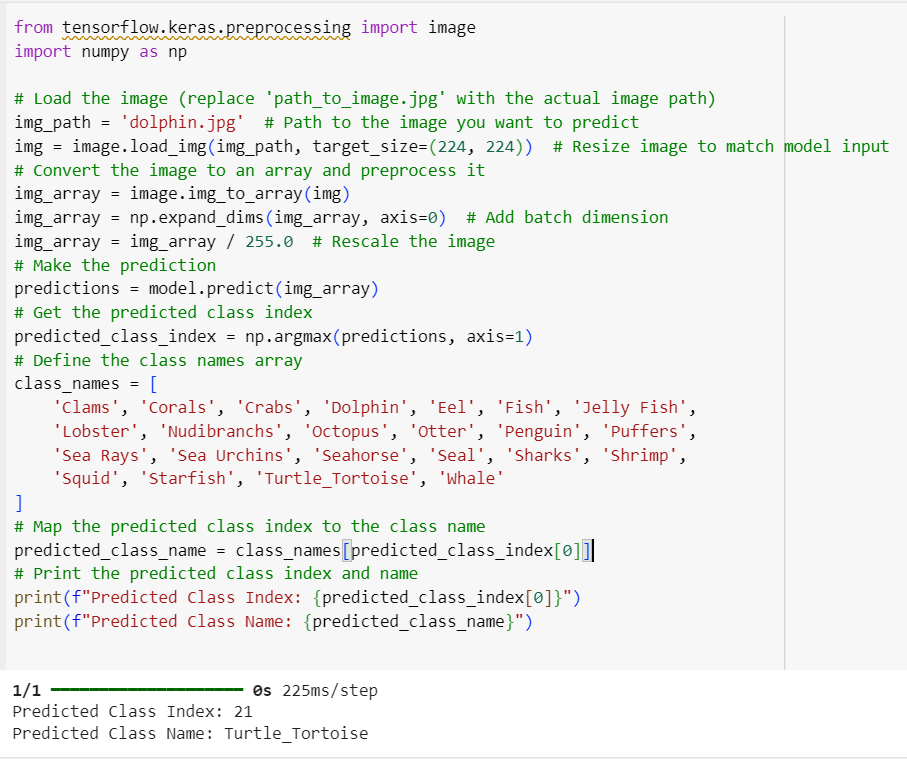


Figure 35 - Image Classification Model Testing

A screenshot of a computer

AI-generated content may be incorrect.

Figure 36 - Diving Score Predictor Testing

A screenshot of a computer

AI-generated content may be incorrect.

Figure 37 - Diving Experience Level Testing

# Deployment

**Deployment of ResNet50**

* The trained model was converted into a deployable format.
* A web-based interface was built for users to upload images and receive predictions.
* The model was optimized for real-time classification using PyTorch and Flask.

**Deployment of EfficientNetB7**

* **Deployment Options:**
* Flask/FastAPI for API deployment
* TensorFlow Serving for scalable deployment
* **Inference Pipeline:**
* Loading model and preprocessing input
* Running inference and post-processing results

## Web Interface

This section describes the deployment strategy for integrating all four machine learning models into a unified web application. The models were made accessible through a user-friendly frontend, enabling interaction with predictive and descriptive marine data.

**Model Integration with Web App**

All models were trained in **Google Colab,** saved as **.pkl files**, and stored **in Google Drive**. The Flask backend loads these models at runtime to deliver real-time predictions through the web interface.

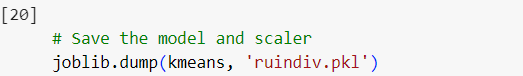


Figure 38 – Save Model

**User Interface Design**

The frontend is developed using **Next.js** and provides three main interactive sections:

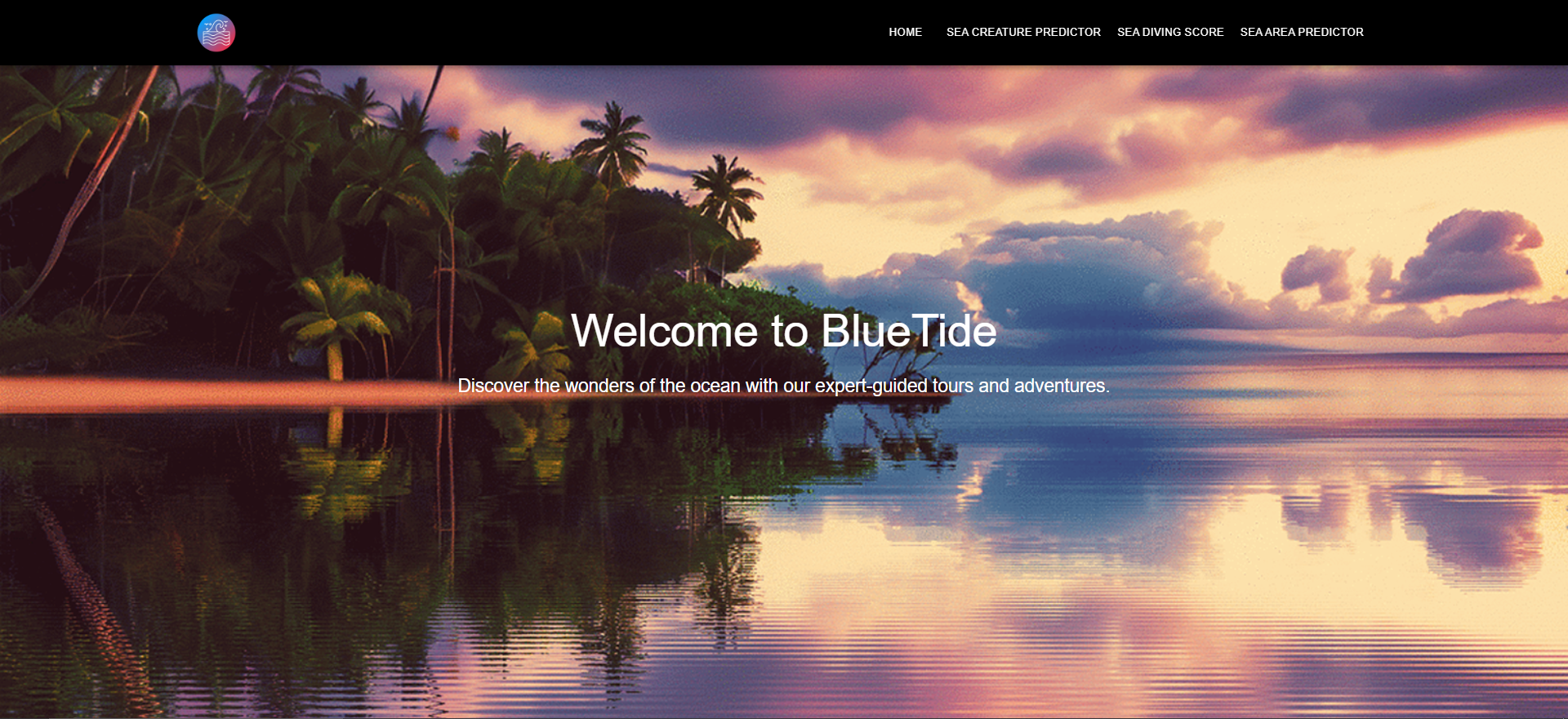


Figure 39 - First Interface

**Sea Creature Identifier**

* + Allows users to upload an image.
  + Displays predicted species name (via ResNet-50 or EfficientNetB7).
  + Fetches additional species information via Gemini API.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 40 - Sea Creature Identifier

**Sea** **Diving Score Predictor**

* + A form where users input discipline, event, country, gender, and year.
  + Displays the predicted diver effort level (Low, Medium, High).

A screenshot of a computer

AI-generated content may be incorrect.

Figure 41 - Sea Diving Score Predictor

**Sea** **Diving Area Predictor**

* + Accepts depth value.
  + Returns diver category: **Beginner**, **Intermediate**, or **Pro**.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 42 - Sea Diving Area Predictor

## Backend code

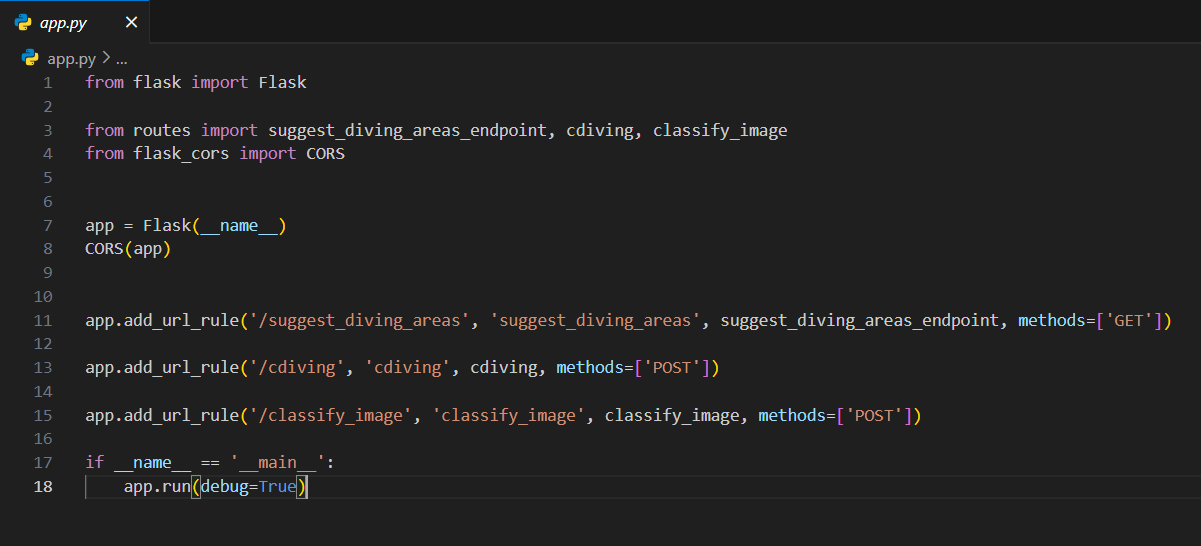


Figure 43 - Backend Code 1

A screenshot of a computer program

AI-generated content may be incorrect.

Figure 44 - Backend Code 2

## Tools & Frameworks Used

|  |  |
| --- | --- |
| **Component** | **Technology / Tool** |
| **Frontend** | Next.js, Tailwind CSS, React Icons |
| **Backend API** | Python |
| **Model Training** | Google Colab |
| **Storage** | Google Drive (Model files) |
| **Deployment** | VS Code (development) |
| **ML Libraries** | TensorFlow, Scikit-learn, Pandas |
| **Integration** | Gemini API (animal descriptions) |

**Video Demonstrate**

This is video for the project. Here display how to work this project. [Click\_to\_view\_video](https://drive.google.com/file/d/1kPDBXTNZl_GyhUkTzhnu0mmeS5m-FYDk/view)

**Project Frontend** [Click](https://drive.google.com/file/d/1OHe1nkS9p3guysgfNGfBXa_ylDafCOMh/view)- **Install** Node modules and **run** npm run dev

**Project Backend** [Click](https://drive.google.com/file/d/1b-ewYk5i2X9szuHhJx7p_n_uQrq5pRHB/view) – **Install** Python and **run** python app.py

# CHAPTER 04: Discussion

The BlueTide Project achieved an 87.8% accuracy on test data, demonstrating strong classification capabilities. However, some misclassifications were observed, primarily due to visually similar species and poor image quality. This highlights the challenge of distinguishing marine species with subtle differences, where even small variations in lighting, pose, or background can impact predictions.

To enhance model generalization, data augmentation techniques played a crucial role by improving the model’s ability to recognize diverse image variations. However, further refinements could further enhance performance.

Potential improvements include:

* Expanding the dataset to include more diverse and high-quality images.
* Fine-tuning the model with additional real-world images to increase robustness and accuracy in practical applications.

These enhancements will ensure that BlueTide continues to evolve into a more reliable and scalable solution for marine biodiversity research and conservation efforts.

# CHAPTER 05 Conclusion

The BlueTide project successfully developed an AI-powered image classification model capable of identifying 23 different species of sea animals. By leveraging deep learning techniques, specifically the ResNet-50 effectively automated the process of marine species identification, making it more accessible to researchers, conservationists, and marine enthusiasts.

Through transfer learning, the model benefited from the vast knowledge acquired from ImageNet pretraining, allowing it to generalize well to marine species classification. Additionally, extensive data preprocessing and augmentation techniques helped enhance model performance by improving its ability to recognize diverse image variations. The evaluation metric, accuracy, provided valuable insights into the model’s strengths and limitations.

Our experimental results showed an 85.3% accuracy on test data, demonstrating the model's strong classification capabilities. Furthermore, the successful deployment of the model as a web-based application allows real-time classification, making it a practical tool for marine research and educational purposes. With further refinements, BlueTide has the potential to contribute significantly to biodiversity conservation efforts by enabling quick and reliable species identification, assisting marine biologists, and supporting ecological studies.

In summary, this project highlights the transformative impact of machine learning in real-world applications and lays the foundation for future advancements in AI-driven marine biodiversity research.

# References

* Bronshtein, I. (2021). *Freediving World Records Dataset*. Retrieved from Kaggle: https://www.kaggle.com/datasets/igalbronshtein/freediving-aida-world-records-dataset
* Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017, February). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, pp. 115–118.
* He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE.
* Jordan, M. I., & Mitchell, T. M. (2015, July). Machine learning: Trends, perspectives, and prospects. *Science*, pp. 255–260.
* Khushi, H. M. T., Masood, T., Jaffar, A., Rashid, M., & Akram, S. (2023). Improved multiclass brain tumor detection via customized pretrained EfficientNetB7 model. *IEEE Access*, pp. 117210–117225.
* LeCun, Y., Bengio, Y., & Hinton, G. (2015, May). Deep learning. *Nature*, pp. 436–444.
* *PyTorch documentation*. (2024). Retrieved from PyTorch: https://pytorch.org/docs/stable/index.html
* Rahul, V. V. (2023). *Image classification using EfficientNetB7*. Retrieved from Kaggle: https://www.kaggle.com/rahulvv/image-classification
* Raijin. (2023). *Sea Animals Image Dataset*. Retrieved from Kaggle: https://www.kaggle.com/datasets/vencerlanz09/sea-animals-image-dataste/
* Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, pp. 85–117.