

Western Governors University

C964 - Computer Science Capstone

Joshua Truong

October 14th, 2024

Table of Contents

Letter of Transmittal	4
 Project Summary:	
Problem Summary	6
Data Product Benefits	6
Data Product Outline	6
Data Description	7
Objective and Hypothesis	7
Project Methodology Outline	7
Funding Requirements	8
Impact on Stakeholders	8
Ethical and Legal Considerations	9
Relevant Expertise	9
 Executive Summary:	
Problem Statement	9
Customer Description	10
Existing Data Products	10
Available Data	10
Project Methodology	11
Deliverables	11
Implementation Plan	11
Meeting Customer Requirements	11
Programming Environments and Human Resources	12
Projected Timeline	12
 Design and Development:	
Descriptive and Non-Descriptive Methods	14
Collected Datasets	15
Decision Support Functionality	16
Featurizing, Parsing, Cleaning, and Wrangling Datasets	16
Data Exploration and Preparation	17
Data Visualization Functionalities	18
Interactive Queries	19
Machine-Learning Methods and Algorithms	19
Accuracy Evaluation	20

Security Features	21
Tools to Monitor	21
User-Friendly and Functional Dashboard	21
Documentation:	
Business Vision	21
Cleaned Datasets	22
Code to Construct Data Product	22
Assessment of Hypothesis	25
Visualizations and Elements	25
Assessment of the Product's Accuracy	28
Data Product Testing, Revisions, and Optimization	28
Source Code and Executable File(s)	29
Quick-Start Guide	30
References	31

Letter of Transmittal

October 1st, 2024

Mr. Jotaro Kujo, Co-Founder and Director

Speedwagon Foundation

Washington D.C. 20202

Dear Mr. Kujo,

As a fellow marine biologist, you're likely familiar with the overwhelming amount of data being collected for marine research every year, as a result of the vast variety of mysterious species that live in the ocean. The vast amount of data collected each year can be challenging for marine workers to manually handle, especially when they need to identify and classify each marine species accurately with minimal error. Contributing to research may also be challenging for newer students or researchers without extensive knowledge or experience because marine research is a uniquely disciplinary field that requires contribution from multiple sciences like physics, biology, chemistry and geology to further advance (Metzke, 2000). For such a reserved field that already has a limited availability of professionals, the low compensation and high workload results in a slowly declining field that hinders scientific research even further. As the volume of data keeps continuing to increase, it is important to support marine biologists with their research by finding a way to make the identification and classification process more efficient. For the past 2 months, my team at Bloop's Boundless Blue Board have been working hard to create a viable solution.

Our solution to streamline the identification and classification of marine species is by developing the Sea Species Sorting System (SSSS). This product is an AI-based system that is designed to automate the entire process. We decided to split this project into two phases with the first phase handling the identification process and the second phase dealing with the classification process. Due to limited resources and budget constraints, we are unable to complete the second phase at this time. The SSSS automates the identification and classification of marine species, significantly reducing the time and effort required for research. By utilizing machine learning and advanced image processing, we ensure accurate species identification while allowing experts to focus on more complex research tasks like exploring the deeper aspects of marine ecosystems. Our first prototype is complete and we are testing it to ensure at least 80% accuracy.

In terms of pricing, I have excluded the expenses related to my team at BBBB and equipment for the first phase, out of respect for my father's friendship with you. Normally, developing this prototype would cost around \$20,000, but I am happy to offer it to you for \$2,000 as a token of appreciation since I've always admired the work of the Speedwagon Foundation. Once we finalize the details, my team can begin working on implementing the second phase. We estimate the cost for equipment for the second phase to be about \$8,000 and the total time to complete to be around 200 hours. If you're interested, we can schedule a time to talk and I can provide more information about the second phase. I look forward to hearing from you soon!

Sincerely,

Joshua Truong

Project Proposal:

Problem Summary

A majority of marine life remains unexplored, with potentially millions of species still unknown. A recent study estimates that there are "at least 750,000 marine species and possibly as many as 25 million marine species" (MarineBio Conservation Society, n.d.). Scientists continue to work on uncovering the secrets of the ocean by accelerating data collection for research each year. However, even known organizations like the SWF struggle to handle the tremendous volume of data and keep up with species identification in a timely manner. As a result, the manual process can not match the speed of data collection, causing delays in marine research advancements.

Data Product Benefits

The Sea Species Sorting System provides value by automating the entire process for the identification of sea species. This automation helps improve the accuracy for species identification and reduces the time and effort needed for manual classification. Through streamlining the process, data accuracy and efficiency increases, while the risk of human errors decreases. By integrating advanced image processing techniques with machine learning, this system supports research experts in making reliable and timely decisions. The SSSS simplifies the research process and accelerates the advancement of scientific knowledge on marine life.

Data Product Outline

The SSSS is an AI-based system designed to predict various marine species from random images with a high degree of accuracy. The system randomly selects 30 images from a pool of 300 and displays them with the actual sea species label on top and the predicted sea species label below.

The label will turn green if the prediction is correct or red if the prediction is incorrect. Each time the system is run, it will display a new set of 30 randomly selected images from the testing dataset with the new labels included. The system also includes visual outputs like an accuracy bar chart and confusion matrix to help with evaluating the model's performance.

Data Description

The data that was used for this project includes a total of 32,160 images of 15 different sea creatures. There were 31,110 images from the training dataset, 750 images from the validating dataset and 300 images from the testing dataset. Each dataset had 15 folders to represent a sea creature. The data varied from 1848 to 2202 images for the classes in the training dataset. The validating dataset had exactly 50 images for each class and the testing dataset had exactly 20 images for each class.

Objective and Hypothesis

The main objective of this project is to develop a model that is capable of accurately identifying different sea species based on the given image data. The hypothesis is that the model can correctly identify the sea species from a random set of images with at least 80% accuracy or higher.

Project Methodology Outline

The project follows the Agile methodology to ensure efficiency and flexibility throughout the development process, which is split in two phases. The first phase focuses on delivering a functional prototype for the identification process while gathering feedback from users. The second phase focuses on implementing the classification process and can begin while collecting

feedback from users. The project will be broken down into multiple sprints for specific tasks and will have continuous integration and testing to help reduce any early issues or problems.

Funding Requirements

The original price for developing the first phase for the SSSS was \$20,000 but was reduced to a final price of \$2,000 because of the close relationship between Mr. Kujo and Mr. Truong's father. The cost for managing the team and equipment required for the first phase was also excluded because of this. The price for the second phase is \$8,000 when it finishes development. The development time will take around 200 hours. The development team at Bloop's Boundless Blue Board has 4 members that get paid \$50/hour each. They work full time with 8 hours a day and 5 days a week. The cost to manage the team will be \$40,000. The total cost for completing the first and second phase is \$50,000.

Impact on Stakeholders

The SSSS is beneficial for primary stakeholders like the organizations at SWF and BBBB. The marine researchers and employees at SWF will be able to drastically reduce the time and energy needed when analyzing large datasets and focus this extra time and effort on other important research matters like deeper analysis on marine ecosystems and biomes. The team at BBBB can benefit through exposure from supporting a prestigious organization like SWF and gain precious experience with developing the system. Other third-party stakeholders can also benefit as well by using the system as a tool to teach and educate their students to train them for practical data analysis in the future.

Ethical and Legal Considerations

The Sea Species Sorting System uses a dataset of publicly available images of marine species, with no sensitive or personally identifiable data involved. The development team prioritizes ethical standards by protecting user privacy and being transparent with data usage. The system does not store any user data, ensuring compliance with legal regulations and maintaining confidentiality.

Relevant Expertise

The BBBB development team has 5 highly skilled individuals that have expertise with technology. The team is led by a leader with decades of experience in software development and team management. The remaining 4 members have completed their masters degree in Computer Science and have extensive knowledge with AI. Through several years of working together, the team was able to create a comfortable and healthy work environment by establishing a deep sense of mutual trust and open communication with each other. Their strategic planning and efficient teamwork allow them to consistently and effectively meet project deadlines.

Executive Summary:

Problem Statement

The core issue in marine research is the lack of tools for marine biologists to support further exploration of the abundant life in the ocean. The SSSS provides a solution for this key task by utilizing an automated system to accurately identify various marine species from given images. Implementing this system would help speed up the analysis of large datasets and reduce potential human errors.

Customer Description

The primary customers are the members of the Speedwagon Foundation that are testing and evaluating the first prototype for this product. The SWF is a big organization that focuses on medical research and environmental conservation. This product gives them an opportunity for improving their conservation efforts and enhancing their research efficiency in marine life. It is an efficient and scalable tool for species identification that solves a critical task for environmental agencies and researchers. Using this tool will free up the organization's resources to be allocated to better topics and help maximize their studies and impact on marine life.

Existing Data Products

The current identification process for sea species uses outdated software tools that are unable to efficiently handle and process large datasets. There are also manual methods for identifying species but they are time consuming and susceptible to human error. These current tools and methods lack the ability to improve research productivity or to scale efficiently.

Available Data

The raw dataset that was sourced from Kaggle included a total of 46,700 images that had 20 different classes for each sea creature. This original dataset was cleaned up to use for the project. The cleaned data that was used for this project includes a total of 32,160 images and has 15 different classes for each sea creature.

Project Methodology

The Agile methodology was followed to help develop the data product development and design. The agile and flexible approach allows the model to be refined through constant testing and feedback.

Deliverables

The primary deliverable for this project is a trained model that uses machine learning to accurately identify and predict random sea creatures from an image dataset. Other deliverables include visual tools to evaluate the model's accuracy.

Implementation Plan

The prototype for the first phase will be released to use for the SWF organization. Depending on their feedback, the prototype will be adjusted accordingly until both parties are satisfied. The prototype for the second phase will begin as soon as the prototype for the first phase is finalized. Once finished, the second prototype will be released and adjusted accordingly just like the first prototype. After both prototypes are approved by both parties, they will be combined into a final product that incorporates tools and methods from both phases. This is the final version that will be released to the public after one final testing.

Meeting Customer Requirements

Another training dataset will be used to test the data product to ensure that it is still able to accurately identify sea creatures. This dataset will contain new image data that the model has not yet seen before. Another accuracy bar chart and confusion matrix will be generated to evaluate

the model's new performance on the separate training dataset. Feedback will be collected from the SWF organization to adjust the product accordingly. The product will be tested thoroughly and continuously to ensure it can consistently identify sea creatures with high accuracy to satisfy its users needs.

Programming Environments and Human Resources

The programming environments for this project include Windows 10, Anaconda 24.7.1, Jupyter Notebook 7.0.8, Python 3.12.4 and the relevant Python libraries. These programming environments are all open-source and free. The human resources needed to execute the first phase in the development are provided with no cost but the second phase will require \$50,000 to cover the expenses of managing the development team.

Projected Timeline

The development for the first phase will take around 2 months or 500 hours. The development for the second phase will take around 25 days or 200 hours. The project will begin on October 1st and is expected to be completed by the end of this year.

Milestone	Start Date	End Date	Duration	Dependencies	Resources
Phase 1 Planning	October 1st, 2024	October 15th, 2024	2 weeks	None	Project Manager, Stakeholders
Phase 1 Development	October 15th, 2024	November 15th, 2024	1 month	Phase 1 Planning	Development Team

Phase 1 Prototype Release and Gathering Feedback	November 15th, 2024	November 22nd, 2024	1 week	Phase 1 Development	Stakeholders, Development Team
Phase 1 Testing and Final Release	November 22nd, 2024	November 30th, 2024	1 week	Phase 1 Prototype Release and Gathering Feedback	Project Manager, Development Team
Phase 2 Planning	November 30th, 2024	December 5th, 2024	5 days	Phase 1 Testing and Final Release	Project Manager, Stakeholders
Phase 2 Development	December 5th, 2024	December 20th, 2024	15 days	Phase 2 Planning	Development Team
Phase 2 Prototype Release and Gathering Feedback	December 20th, 2024	December 22nd, 2024	2 day	Phase 2 Development	Stakeholders, Development Team
Phase 2	December	December	3 days	Phase 2	Project

Testing and Final Release	22nd, 2024	25th, 2024		Prototype Release and Gathering Feedback	Manager, Development Team
Combining Phase 1 and Phase 2 and Gathering Feedback	December 25th, 2024	December 30th, 2024	5 days	Phase 2 Testing and Final Release	Project Manager, Stakeholders, Development Team, QA Testers
Final Release for Phase 1 and 2 Combined Prototype	December 30th, 2024	December 31st, 2024	1 day	Combining Phase 1 and Phase 2 and Gathering Feedback	Project Manager, Development Team

Design and Development:

Descriptive and Non-Descriptive Methods

My descriptive method includes a summary statistic of the dataset and the visualizations that are in my Jupyter Notebook. The visualizations include the accuracy bar chart, confusion matrix and an image display for showing the predictions with the appropriate labels. My predictive method

is my sea creature classification model which uses machine learning to predict the correct species based on the images.

Collected Datasets

I collected my image dataset for the various sea creatures from Kaggle. The original set already had three divisions for testing, training and validating. Each of these sets had 20 different sea creature classes, each with multiple images. I decided to narrow down the dataset by deleting five of those classes (eel, otter, seahorse, sea urchin, shark) for this project.




Sea Animals Images Dataset

▲
2

New Notebook





















[Data Card](#)
[Code \(1\)](#)
[Discussion \(0\)](#)
[Suggestions \(0\)](#)

train (20 directories)

About this directory
[Add Suggestion](#)

This file does not have a description yet.

 Clam 2090 files	 Crab 2150 files	 Dolphin 1855 files	 Eel 2160 files
 Fish 3164 files	 Jelly Fish 2016 files	 Lobster 2210 files	 Octopus 1939 files
 Otter 2631 files	 Puffer 3271 files	 Sea Horse 1860 files	 Sea Ray 1984 files
 Sea Turtle 2164 files	 Sea Urchin 1784 files	 Seal 2050 files	 Shark 3389 files
 Shrimp 2012 files	 Squid 2158 files	 Starfish 2073 files	 Whale 2344 files

Decision Support Functionality

This model helps with identifying a random variety of sea creatures which can be valuable for marine research and education. The user can decide whether the model is reliable enough to deploy now or if it needs further training based on the overall performance shown through the accuracy bar chart and confusion matrix.

Featurizing, Parsing, Cleaning, and Wrangling Datasets

For featurizing the dataset, I utilized a pre-trained convolutional neural network model to extract key features from the images. I chose MobileNetV2 for this project because of its effectiveness in capturing important details like edges, patterns and textures within the images. After extracting these features, I processed them through custom classification layers to predict the correct sea creature class.

For parsing the dataset, I implemented the ImageDataGenerator from TensorFlow to load and preprocess the images in batches. The images were organized into directories based on species and the labels were parsed from the directory structure to ensure that each image was correctly mapped to its associated class.

For wrangling the dataset, I applied data augmentation techniques to increase the dataset's size and diversity. These techniques included rotating, flipping, and shifting the images which allowed the model to generalize better by exposing it to various orientations of the same sea creatures. Additionally, I normalized the pixel values of the images, and resized them to a fixed dimension to ensure consistency for the model.

Data Exploration and Preparation

For the initial cleaning, I narrowed down the dataset by removing five classes (eel, otter, seahorse, sea urchin, shark) from the original 20 classes. The training set consisted of around 1,800 to 3,200 images per species, while the validation set had exactly 50 images per species, and the testing set had exactly 20 images per species. To make the dataset fair and balanced, I manually deleted excess images from certain classes in the training set until each class had approximately 2,000 images. I wrote the Python code below to verify the number of images in each training directory.

```
training_dir = 'C:\\Users\\JoshyBoi\\WGU C964 Final Capstone\\Dataset\\Training'
validating_dir = 'C:\\Users\\JoshyBoi\\WGU C964 Final Capstone\\Dataset\\Validating'
testing_dir = 'C:\\Users\\JoshyBoi\\WGU C964 Final Capstone\\Dataset\\Testing'

species_images = {}

for species in os.listdir(training_dir):
    species_class = os.path.join(training_dir, species)
    images = len([img for img in os.listdir(species_class)])
    species_images[species] = images

for species, count in species_images.items():
    print(f"{species}: {count} images")
```

In this code, I created three variables to define the directories where the images for the different datasets are located. I initialized a dictionary for storing the total amount of images for each species. Then I looped through each species to get the total count and display the results to make sure that my dataset is balanced.

I also deleted bad images from the testing dataset if they were blurry, low quality, or not accurate for representing the sea species. These images were replaced with better-quality images that I manually selected from the training dataset. In addition, I renamed some of the files for better

organization and clarity. For preprocessing the data, I used the ImageDataGenerator from TensorFlow to load, batch and augment the data to help enhance the model's performance. Lastly, I applied preprocessing techniques like normalization and image resizing to ensure uniformity and allow the model to process the data efficiently.

Data Visualization Functionalities

I displayed the class distribution for each sea creature in the training dataset by printing the number of available images of each class to the terminal. This output helped me to verify that the dataset was balanced and check for any imbalances to adjust the data as needed.

```
Clam: 2078 images
Crab: 2138 images
Dolphin: 1848 images
Fish: 2154 images
Jellyfish: 2007 images
Lobster: 2198 images
Octopus: 1927 images
Puffer: 2198 images
Seal: 2039 images
Shrimp: 2000 images
Squid: 2145 images
Starfish: 2065 images
Stingray: 1968 images
Turtle: 2143 images
Whale: 2202 images
```

I generated an accuracy bar chart to show the number of correct and incorrect predictions and assess the model's overall performance . This simple visual provides an easy way to measure the model's accuracy and spot any potential issues.

I incorporated a confusion matrix to show both the correct and incorrect predictions for each class to diagnose any misclassifications. This visual highlights areas that may need further improvement by identifying which classes the model may be struggling with.

Interactive Queries

This project uses Jupyter Notebook as a fully interactive platform. The real-time interaction with the code gives users complete control over the various processes. Users are able to explore different sections of the data and the model training, modify the inputs, and execute specific cells without the need for a separate GUI. The interactive functionality of the notebook creates a dynamic and user-friendly environment, making it easy for users to engage with the project as needed.

Machine-Learning Methods and Algorithms

MobileNetV2 is used as the base model for developing the CNN because its efficient and lightweight architecture allows for high accuracy in image classification tasks which make it ideal for devices with limited resources (Sharma, 2024).

The first step is to initialize the pre-trained model with weights learned from training on ImageNet which is a large dataset that consists of 1.4M images and 1000 classes (TensorFlow, n.d.). The classification layers at the top are excluded to allow for custom classification layers to be added. The pre-trained base layers are frozen to retain the knowledge acquired from ImageNet and to prevent their weights from changing during the training process.

Now custom layers can be added on top of the pre-trained model. The first custom layer is a dense layer with 1024 units, to help the model learn complex patterns from the extracted features. Following this, a dropout layer with a rate of 0.5 is added to prevent overfitting by randomly dropping half of neurons during training. A second dense layer with 512 units is added to refine the model's learning even further. A batch normalization layer is included to stabilize and speed up the training. After this is another dropout layer to further prevent overfitting.

Finally, the output layer with softmax activation is added to convert the output to a probability distribution, to allow the model to predict the correct sea creature class.

The model is compiled using the Adam Optimizer to dynamically adjust the learning rate and ensure efficient model convergence. Since this is a multi-class classification task, categorical cross-entropy is used to calculate how well the model's predictions match the true class. The accuracy metric is set to track the model's performance.

Early stopping is implemented to track validation loss and the patience level is set to 5. To avoid overfitting, the model will stop training if the validation loss does not improve after 5 consecutive epochs. An option to restore the best-performing model weights is included and checkpoints are used to save the model's weights whenever the validation accuracy improves. The model is trained for 50 epochs, although training may end earlier if the validation loss does not improve, due to early stopping. The callbacks for early stopping and checkpointing ensures that the best version of the model is saved while preventing overfitting.

Accuracy Evaluation

To evaluate the accuracy of my data product, I utilized an accuracy bar chart to clearly display the number of correct and incorrect predictions made by the model for the testing dataset. This functionality shows a simple representation of the model's performance in a way that is easy to understand.

I also implemented a confusion matrix to further break down the results by class. This functionality shows how accurate the model's predictions are for each sea creature class and provides a more in-depth analysis of where the model performs as expected or where improvements may be needed.

Security Features

In the future, the application will integrate security features to protect user privacy and ensure data integrity. One feature is having a password login system to ensure that only authorized users can access the application. This feature will require users to create strong passwords with a minimum of 8 characters that include uppercase and lowercase letters as well as at least 1 number or special character. The system will support two factor authentication and the passwords will be hashed and stored with encryption techniques to further enhance the overall security.

Tools to Monitor

In the future, the application will implement monitoring tools like Prometheus to track system metrics such as server load and response times to ensure the application operates efficiently. The application will also have periodic automatic security patches to ensure that the application is up to date and can handle vulnerabilities.

User-Friendly and Functional Dashboard

The application features a user-friendly and interactive notebook that is easy and intuitive to navigate. Users have the option to adjust any input from whatever section they want and can run code for certain cells with real-time response.

Documentation:

Business Vision

The vast majority of mysterious species existing in the ocean still remain undocumented, causing a challenge for even experienced experts to fully comprehend marine life and existence. A

fascinating study reveals that only “91 percent of ocean species have yet to be classified, and that more than 80 percent of our ocean is unmapped, unobserved, and unexplored” (National Ocean Service, n.d.). The primary purpose of this project is to provide marine workers an accurate and efficient tool for classifying sea creatures based on images. To aid in identifying sea species faster, the model implements advanced machine learning techniques to predict the specific type of sea creature from a given image. Through automating the identification process, the time and effort required for manual classification is significantly reduced and the accuracy and efficiency for marine research is drastically improved.

Cleaned Datasets

The original dataset that I sourced from Kaggle consisted of 20 classes to represent a different sea creature. I personally removed 5 of the classes I believed were less relevant which were the eel, otter, seahorse, sea urchin and shark species. Some of the files were renamed for organizational consistency. These adjustments were made manually, so there is no scraping or cleaning code applicable. For preprocessing the data, I used the TensorFlow and Keras libraries to create scripts that normalized the pixel values and resize all images to 224x224 pixels for uniformity. All the preprocessing code is available in the Jupyter Notebook.

Code to Construct Data Product

```
training_paths = []
training_labels = []

for label in os.listdir(training_dir):
    species_class = os.path.join(training_dir, label)
    for image in os.listdir(species_class):
        training_paths.append(os.path.join(species_class, image))
        training_labels.append(label)

training_df = pd.DataFrame({'ImagePath': training_paths, 'Label': training_labels})
training_df.head()
```

```

validating_paths = []
validating_labels = []

for label in os.listdir(validating_dir):
    species_class = os.path.join(validating_dir, label)

    for image in os.listdir(species_class):
        validating_paths.append(os.path.join(species_class, image))
        validating_labels.append(label)

validating_df = pd.DataFrame({'ImagePath': validating_paths, 'Label': validating_labels})
validating_df.head()

```

```

testing_paths = []
testing_labels = []

for label in os.listdir(testing_dir):
    species_class = os.path.join(testing_dir, label)

    for image in os.listdir(species_class):
        testing_paths.append(os.path.join(species_class, image))
        testing_labels.append(label)

testing_df = pd.DataFrame({'ImagePath': testing_paths, 'Label': testing_labels})
testing_df.head()

```

I created two empty lists to store the paths to the training images and for the matching class labels. I looped through each folder in the training directory to create a path for each class folder and used a nested loop to iterate through each image in that class folder to create the full path file and the matching label for that image. This data is added to the two empty lists that were created earlier. After this, I created a dataframe to store all the data for the paths and labels. Then I checked to see if the data was stored correctly by displaying the first few lines from the dataframe. I repeated the same code for both the validating and testing datasets.

The next 3 sections of code have already been explained in Task C (implementation of machine learning methods and algorithms) so I will not explain them again for the sake of brevity.

```

training_dataset = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    zoom_range=0.2,
    shear_range=0.2,
    fill_mode='nearest',
    horizontal_flip=True)

validation_dataset = ImageDataGenerator(rescale=1./255)
testing_dataset = ImageDataGenerator(rescale=1./255)

training_images = training_dataset.flow_from_directory(
    training_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32)

validation_images = validation_dataset.flow_from_directory(
    validation_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32)

testing_images = testing_dataset.flow_from_directory(
    testing_dir,
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32)

```

```

mobile = MobileNetV2(
    input_shape=(224, 224, 3),
    include_top=False,
    weights='imagenet',
    pooling='avg')

mobile.trainable = False

model = Sequential([
    mobile,
    Dense(1024, activation='relu'),
    Dropout(0.5),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dropout(0.5),
    Dense(15, activation='softmax')])

model.compile(
    optimizer=Adam(learning_rate=0.00001),
    loss='categorical_crossentropy',
    metrics=['accuracy'])

model.summary()

```

```

early_stop = EarlyStopping(
    monitor='val_loss',
    patience=5,
    restore_best_weights=True)

checkpoint = ModelCheckpoint(
    'updated_model.h5',
    monitor='val_accuracy',
    save_best_only=True)

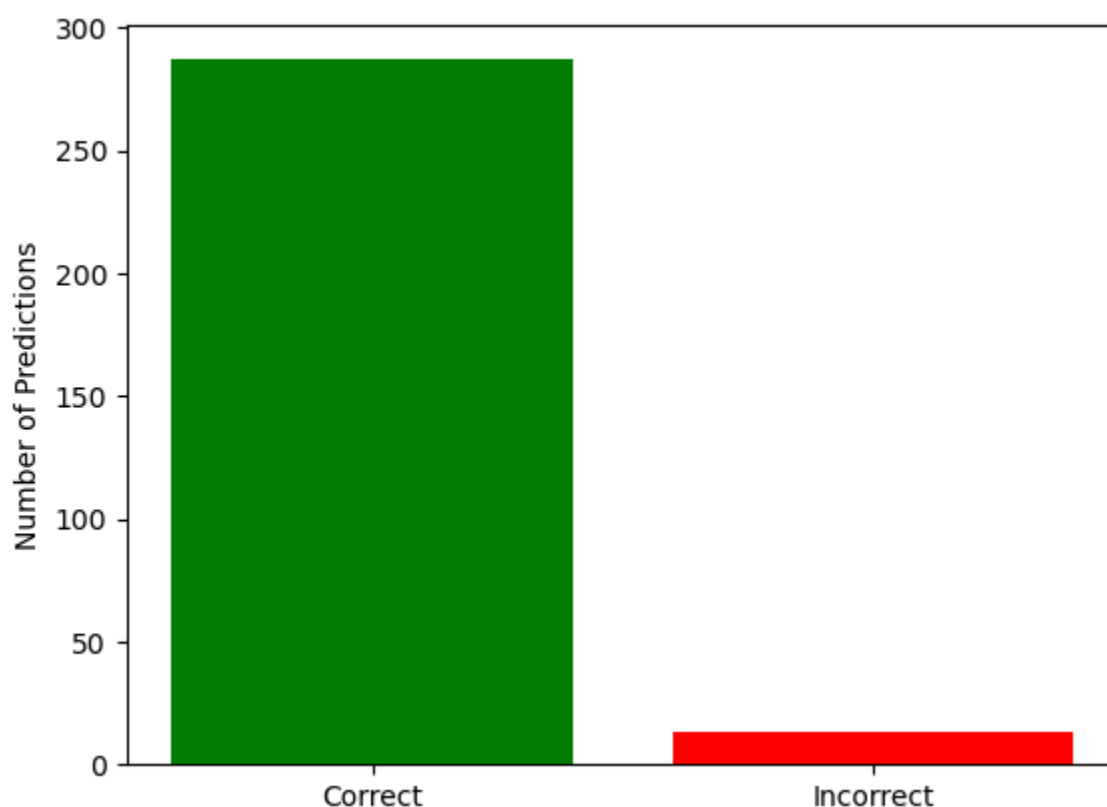
history = model.fit(
    training_images,
    validation_data=validation_images,
    epochs=50,
    callbacks=[early_stop, checkpoint])

```

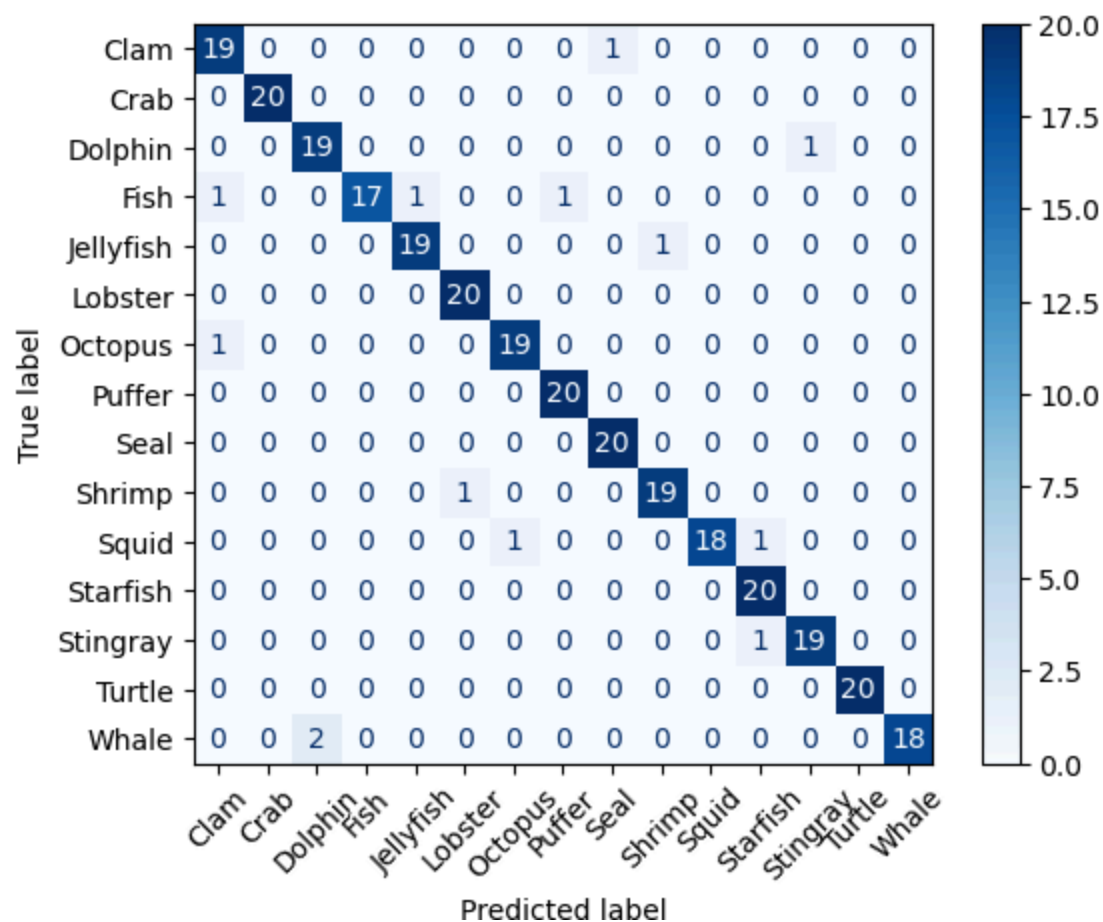

Assessment of Hypothesis

The hypothesis of this project was that the data product model will achieve at least 80% accuracy or more when identifying images. After testing, the hypothesis was confirmed with a final accuracy score of 95.67%. This result proves that the model is capable of reliably identifying sea creatures accurately, to help support researchers in their studies.

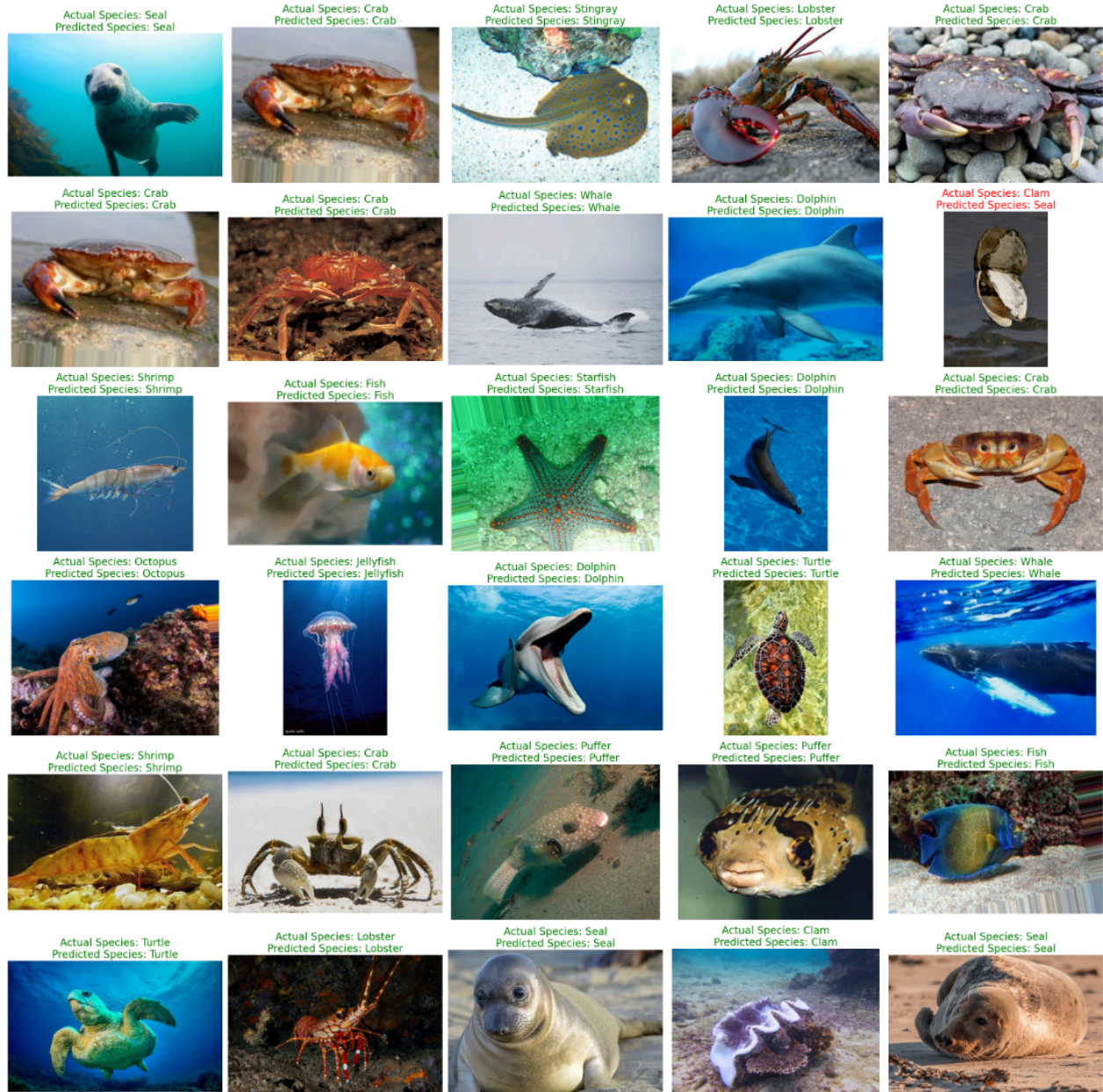
Visualizations and Elements



This is the visualization of the accuracy bar chart for the model's overall performance on the testing dataset. It displays the number of correct and incorrect predictions with the green bar representing the correct predictions and the red bar representing the incorrect predictions. The accuracy bar chart provides a quick visual understanding for the accuracy of the model.



This is the visualization of the confusion matrix for the model's performance across the different classes. The vertical labels represent the actual species from the testing dataset and the horizontal labels represent the predicted species. This matrix highlights accurate predictions (diagonal elements) and identifies misclassifications (non-diagonal elements) to provide insight in certain classes that the model may struggle with.



This visualization displays a 6x5 grid of 30 random sea creature images pulled from the testing dataset. Each image includes labels for the actual species and the model's predicted species, with the label turning green for the correct predictions or red for the incorrect predictions. This visualization provides clear examples of both successful and unsuccessful classifications, demonstrating the model's ability to identify various sea creatures.

Assessment of the Product's Accuracy

The model correctly identified 287 sea creatures out of 300 sea creatures from the testing dataset with an overall accuracy of 95.67%. The confusion matrix identifies that the fish class was the most misclassified with 3 incorrect predictions out of 20 images. It also highlights classes with perfect predictions like the crab, lobster, puffer, seal, starfish and turtle species. From these results, the model successfully demonstrates a high level of accuracy.

Data Product Testing, Revisions, and Optimization

Old Code for Training Earlier Model:

```
# Data augmentation to enhance generalization during training
training_dataset = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    zoom_range=0.2,
    fill_mode='nearest',
    horizontal_flip=True,
    rescale=1./255)

# Rescaling pixel values for validation and testing dataset
validation_dataset = ImageDataGenerator(rescale=1./255)
testing_dataset = ImageDataGenerator(rescale=1./255)

# Load and preprocess the datasets
training_images = training_dataset.flow_from_directory(
    'C:\\Users\\JoshyBoi\\New Sea Creature Dataset\\training',
    target_size=(224, 224),
    class_mode='categorical',
    batch_size=32)

validation_images = validation_dataset.flow_from_directory(
    'C:\\Users\\JoshyBoi\\New Sea Creature Dataset\\validation',
    target_size=(224, 224),
    class_mode='categorical',
    batch_size=32)

testing_images = testing_dataset.flow_from_directory(
    'C:\\Users\\JoshyBoi\\New Sea Creature Dataset\\testing',
    target_size=(224, 224),
    class_mode='categorical',
    batch_size=32)
```

```
mobile = MobileNetV2(
    input_shape=(224, 224, 3),
    include_top=False,
    weights='imagenet',
    pooling='avg')

mobile.trainable = False

model = Sequential([
    mobile,
    Dense(1024, activation='relu'),
    Dropout(0.5),
    Dense(15, activation='softmax')])

model.compile(
    loss='categorical_crossentropy',
    metrics=['accuracy'])

model.summary()
```

When training the model for the first time, the accuracy for correct predictions was lower than expected because of specific misclassifications from similar species. After analyzing the results, I made some adjustments like tuning the hyperparameters and replacing some low-quality images from the testing dataset, to ensure high-quality representations of each species. For the adjustments, I modified the rotation range from 20 to 40 and added shear range to expose the model to a greater variety of rotations and reduce overfitting. I also set 'rgb' as the color mode to help capture subtle color differences. Additionally, I added 3 new custom layers to help the model generalize better and capture more complex features. Finally, I fine-tuned the learning rate to enhance the model precision with weights. All of these improvements help the model achieve a 95% accuracy level.

Source Code and Executable File(s)

The “Trained Model.h5” file is the final sea species classification model after training that is used to load the trained model in Jupyter Notebook. The “Sea Species Sorting System.ipynb” file

contains the code for the data product and for evaluating the model with visualizations in Jupyter Notebook. The “Computer Science Capstone.pdf” file contains all the project proposal, executive summary, design and development process, documentation and references. The “Model Training.ipynb” file contains the code for preprocessing, building and training the model. The “README.txt” file contains an explanation for why the training dataset is excluded as well as the user guide. The “Dataset” folder contains the testing dataset used for model predictions.

Quick-Start Guide

1. Make sure to have Python version 3.8 or newer on your system
2. If you don’t already have Anaconda, install it from <https://www.anaconda.com/download>
3. Unzip the project files from “WGU C964 Final Capstone” folder and place them into a new folder
4. Open Anaconda Prompt to type “jupyter notebook” and hit enter
5. Once the notebook launches, click on the “upload” button on the top right and navigate to the folder where you unzipped the project files
6. Upload the “Trained Model.h5”, “Model Training.ipynb” and “Sea Species Sorting System.ipynb” files
7. Once uploaded, open the new “Model Training.ipynb” file to see the model training process or “Sea Species Sorting System.ipynb” file to see the data product and visualizations
8. Run the notebook cells in order from top to bottom
9. Install any missing dependencies if the files fail to run by using “!pip install <package>” (e.g., !pip install tensorflow)

References

MarineBio Conservation Society. (n.d.). *Marine life facts*. MarineBio Conservation Society.

<https://www.marinebio.org/creatures/facts/#:~:text=So%2C%20there%20are%20at%20least,97%25%20of%20the%20Earth%27s%20water>

Metske, R. (2000, June 16). *Marine research: Challenges and chances*. Science.

<https://www.science.org/content/article/marine-research-challenges-and-chances>

National Ocean Service. (n.d.). *How many species live in the ocean?* NOAA.

<https://oceanservice.noaa.gov/facts/ocean-species.html>

Sharma, N. (2024, June 4). *What is MobileNetV2? Features, Architecture, Application and More*.

Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2023/12/what-is-mobilenetv2/>

TensorFlow. (n.d.). *Transfer learning with TensorFlow*. TensorFlow.

https://www.tensorflow.org/tutorials/images/transfer_learning