Aquarium animal identification system

1. Abstract

This project focuses on developing an animal identification system using Convolutional Neural Networks (CNNs) for image classification, specifically targeting marine organisms. The RegNet Y800 model from the SuperGradients library, based on PyTorch, was utilized for training the dataset. The model, optimized for computer vision tasks, demonstrated high efficiency and accuracy in image classification.

However, challenges such as computational complexity and task specialization were identified. The system integrated SQLite for database management and utilized OpenCV for image capture. The web application, built with Flask, facilitated user interaction and information retrieval. Experimental studies validated the model's performance, achieving an accuracy of 86.05% with a Top-5 accuracy of 97.25%.

The project successfully addressed the need for accurate animal identification in aquariums, enhancing visitor experiences and promoting marine life conservation. Future enhancements aim to optimize system performance and expand the marine life database for broader recognition capabilities.

2. Introduction

In today's digital age, the rapid development of technology has brought unprecedented exhibition and educational opportunities to cultural institutions. In places like aquariums, visitors are often fascinated by the amazing variety of marine life, but identifying specific animal species can be a challenge. Although the aquarium has an introduction board, it is often difficult for visitors to associate the animal with the introduction board due to the mobility of the animals.

However, there is currently no corresponding product on the market. Therefore, we decided to start researching and developing an animal identification system.Our animal recognition system uses a convolutional neural network (CNN) and the RegNet Y800 model of the SuperGradients library, combined with computer vision and deep learning algorithms, to accurately and quickly identify various marine life in the aquarium. Through mobile apps or specialized identification devices, visitors only need to take photos of animals or point at them, and the system will display the animal's detailed information, ecological habits, and interesting anecdotes in a few seconds.

This new interactive experience not only allows visitors to learn more about marine life, but also provides them with a new way to visit the aquarium, making visiting the aquarium more fun and meaningful. This innovative working method not only enhances the visitor experience, but also demonstrates the cultural institution’s position at the forefront of the digital age. By using CNN and RegNet Y800 models, we achieved efficient image classification and recognition, bringing new possibilities to science education and interactive experiences in aquariums.

3. Methodology

3.1 CNN animal identification system

The task of our project mainly focuses on image classification, and we apply the convolutional neural network to achieve our goal. Based on this, we use the Python library “SuperGradients” in our project. SuperGradients is an open-source training library based on PyTorch, which has many pre-trained models available, and these training recipes are accurate. In our project, we use the RegNet Y800 model of SuperGradients to train our dataset. This model is a machine learning model that can classify images from the Imagenet dataset.
The RegNet Y800 model is a convolutional neural network (CNN). It is designed specifically for computer vision tasks, which are typical applications of CNNs. It is based on the principles of CNN and uses convolutional layers to process and learn image data. Convolutional layers are good at capturing spatial hierarchy and patterns in visual input, making them very effective in tasks such as image classification and object detection. The model architecture systematically explores the design space of possible network architectures to find efficient models. It uses regular patterns of convolutional layers and other operations such as batch normalization and ReLU activations to form a network that is both scalable and performant.

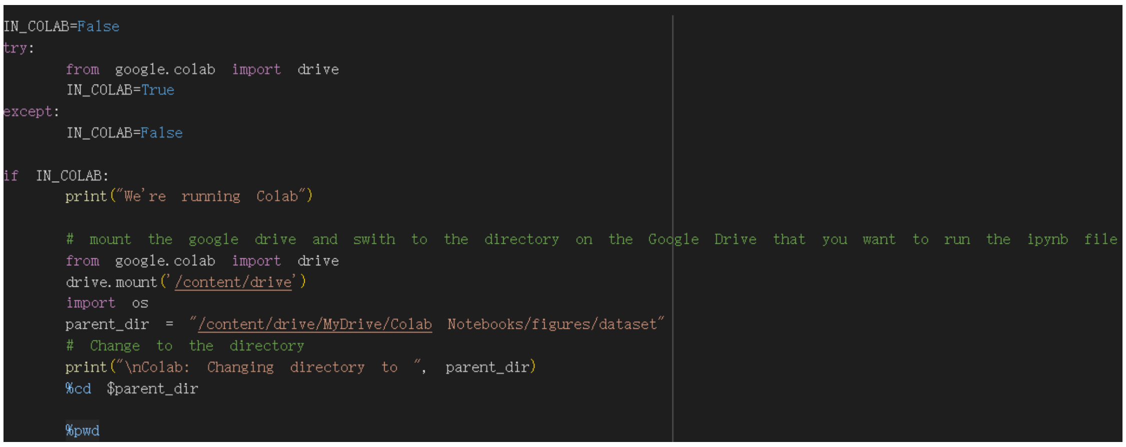
3.2 Pros and Cons of the model

Pros:
1. Embedding early-stop function to avoid over fitted
During the model training phase, if the model's performance on the validation set does not improve within 7 consecutive training epochs (accuracy does not improve or loss does not decrease), training will end early. This aims to prevent overfitting and ensure that the model achieves optimal performance while maintaining generalization.
2. High efficiency for use
The model design space provides a simple and fast network that performs well under various flipping mechanisms. RegNet models outperform the popular EfficientNet model under comparable training settings and flash conditions while running up to 5x faster on GPUs.

Cons:
1. Numerous parameter for training
The model has many built-in parameters for the training process, which requires a lot of computing power to achieve. Ordinary equipment can easily cause training failure due to too many parameters that need to be adjusted during the training process.
2. Specialization to Vision Tasks
This model is designed primarily for computer vision tasks. While this specialization enables them to perform well in this domain, it may limit their applicability to other types of problems, such as natural language processing or time-series analysis.

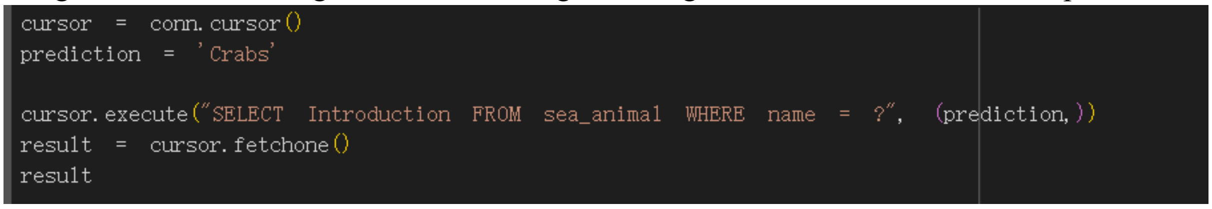
3.3 Database creation (SQLite):
First we collected the required information on marine organisms(name&introduction) based on the model training database and organized it into a CSV file named “sea_animal_introduction”. Subsequently, we imported this information into SQLite to generate a DB file. And firstly, we imported the sqlite3 library into python.

Next, the required libraries for SQLite are imported in Python and the DB file is uploaded to a Google Drive folder and the DB file is read via Python.



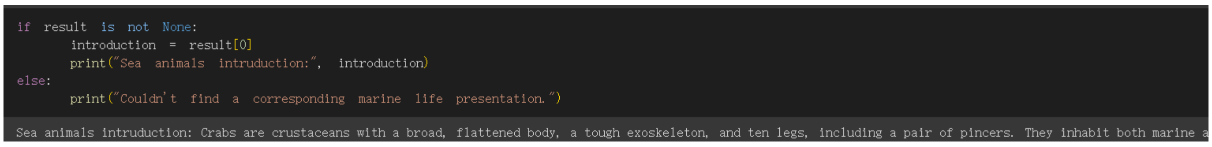
```
IN_COLAB=False
try:
        from  google.colab  import  drive
        IN_COLAB=True
except:
        IN_COLAB=False

if  IN_COLAB:
        print("We're  running  Colab")

        #  mount  the  google  drive  and  swith  to  the  directory  on  the  Google  Drive  that  you  want  to  run  the  ipynb  file
        from  google.colab  import  drive
        drive.mount('/content/drive')
        import  os
        parent_dir  =  "/content/drive/MyDrive/Colab  Notebooks/figures/dataset"
        #  Change  to  the  directory
        print("\nColab:  Changing  directory  to  ",  parent_dir)
        %cd  $parent_dir

        %pwd
```

Then we create a cursor object to execute the SQL query, followed by executing the model recognition and obtaining the results, storing the recognition results in the variable prediction.



```
cursor  =  conn.cursor()
prediction  =  'Crabs'

cursor.execute("SELECT  Introduction  FROM  sea_animal  WHERE  name  =  ?",  (prediction,))
result  =  cursor.fetchone()
result
```

Finally, the cursor object is used to execute the query to obtain the corresponding marine life introduction.



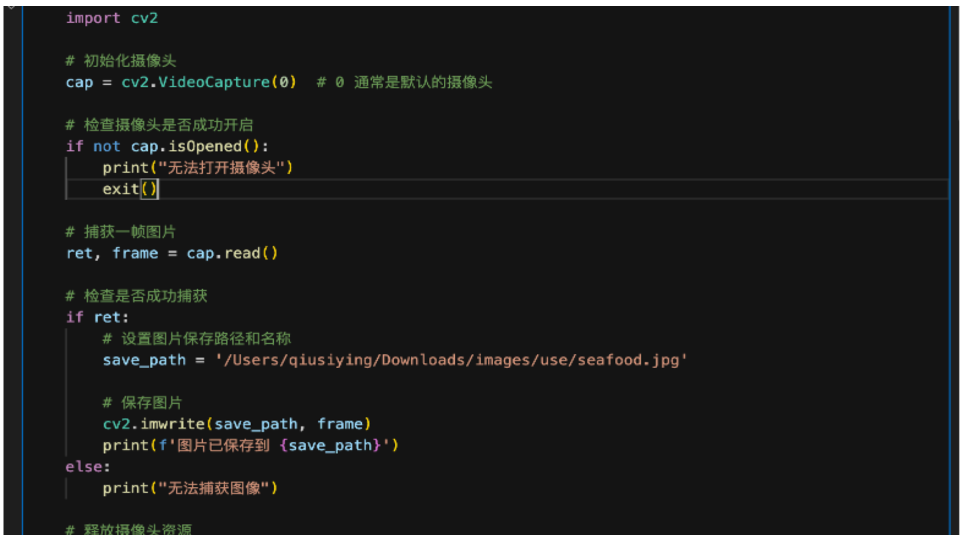
```
if  result  is  not  None:
        introduction  =  result[0]
        print("Sea  animals  intruduction:",  introduction)
else:
        print("Couldn't  find  a  corresponding  marine  life  presentation.")

Sea animals intruduction: Crabs are crustaceans with a broad, flattened body, a tough exoskeleton, and ten legs, including a pair of pincers. They inhabit both marine a
```

3.4 Market application

3.4.1 Image capture

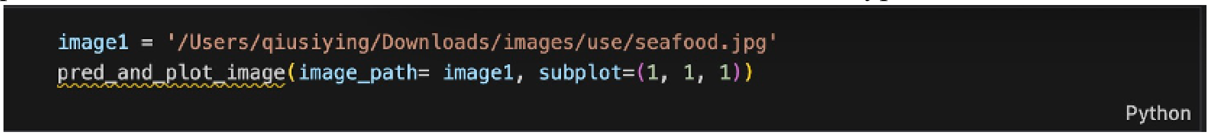
Visitors often see many unusual animals in aquariums, but it can be difficult to identify specific species. Although aquariums often provide plaques, due to the mobility of aquatic animals, it is often difficult for visitors to associate a specific animal with the corresponding plaque.Therefore, tourists can upload their own animal pictures to our system through image shooting, but how do the pictures taken by the lens connect to the system.

Based on market research, we finally chose to use Cv2. 'Cv2 is the Python interface to OpenCV, an open source computer vision and image processing library. It provides a rich set



```
import cv2

# 初始化摄像头
cap = cv2.VideoCapture(0)  # 0 通常是默认的摄像头

# 检查摄像头是否成功开启
if not cap.isOpened():
    print("无法打开摄像头")
    exit()

# 捕获一帧图片
ret, frame = cap.read()

# 检查是否成功捕获
if ret:
    # 设置图片保存路径和名称
    save_path = '/Users/qiusiying/Downloads/images/use/seafood.jpg'

    # 保存图片
    cv2.imwrite(save_path, frame)
    print(f'图片已保存到 {save_path}')
else:
    print("无法捕获图像")

# 释放摄像头资源
```

of functions and tools for processing image and video data, including image processing, feature detection, object recognition, target tracking and other functions.
Based on data study and repeated experiments, we finally wrote this code. When the code is running, the system will automatically turn on the camera to capture images, capture animal pictures in real time and store them in the path we set (save_path). The file of this path will be imported into the animal recognition system through the following code, and a series of pictures taken of a certain animal will be used to detect the animal type.

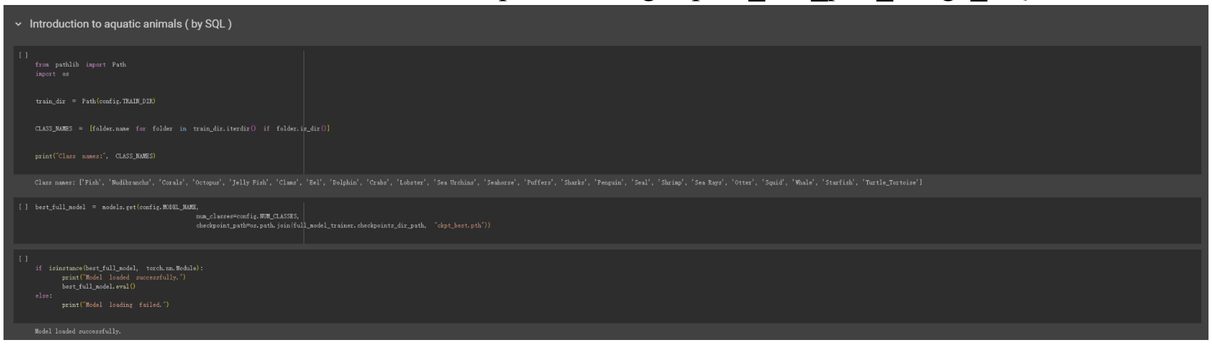


```python
image1 = '/Users/qiusiying/Downloads/images/use/seafood.jpg'
pred_and_plot_image(image_path= image1, subplot=(1, 1, 1))
```

Python

3.4.2 Specific applications of database

In order to allow the audience to better understand marine life, we established a database of marine life-related information and output it through ‘pred_and_plot_image_SQL‘.



Introduction to aquatic animals (by SQL)

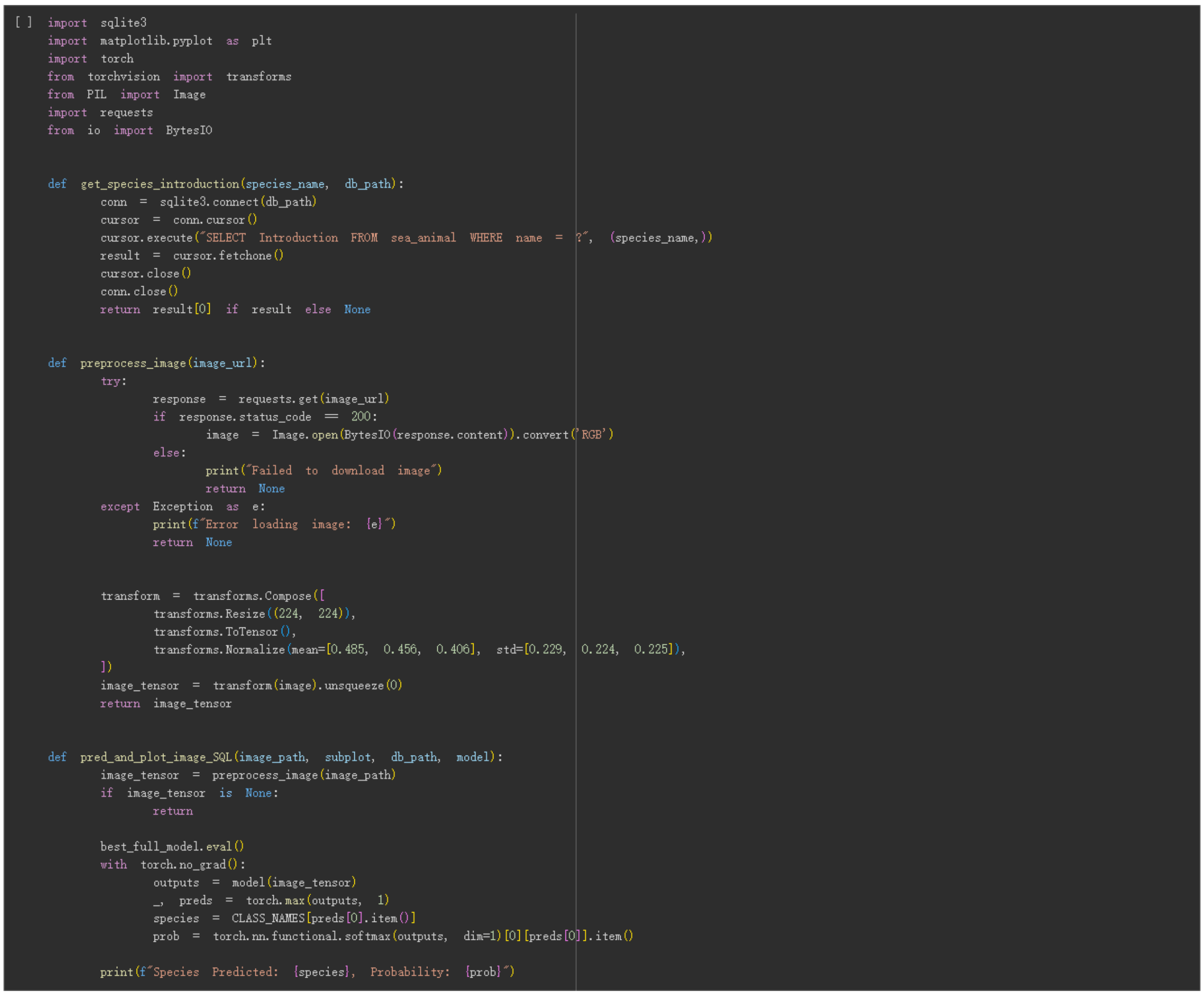
```
[ ]
    from pathlib import Path
    import os


    train_dir = Path(config.TRAIN_DIR)


    CLASS_NAMES = [folder.name for folder in train_dir.iterdir() if folder.is_dir()]


    print("Class names:", CLASS_NAMES)
```

```
    Class names: ['Fish', 'Nudibranchs', 'Corals', 'Octopus', 'Jelly Fish', 'Clams', 'Eel', 'Dolphin', 'Crabs', 'Lobster', 'Sea Urchins', 'Seahorse', 'Puffers', 'Sharks', 'Penguin', 'Seal', 'Shrimp', 'Sea Rays', 'Otter', 'Squid', 'Whale', 'Starfish', 'Turtle_Tortoise']
```

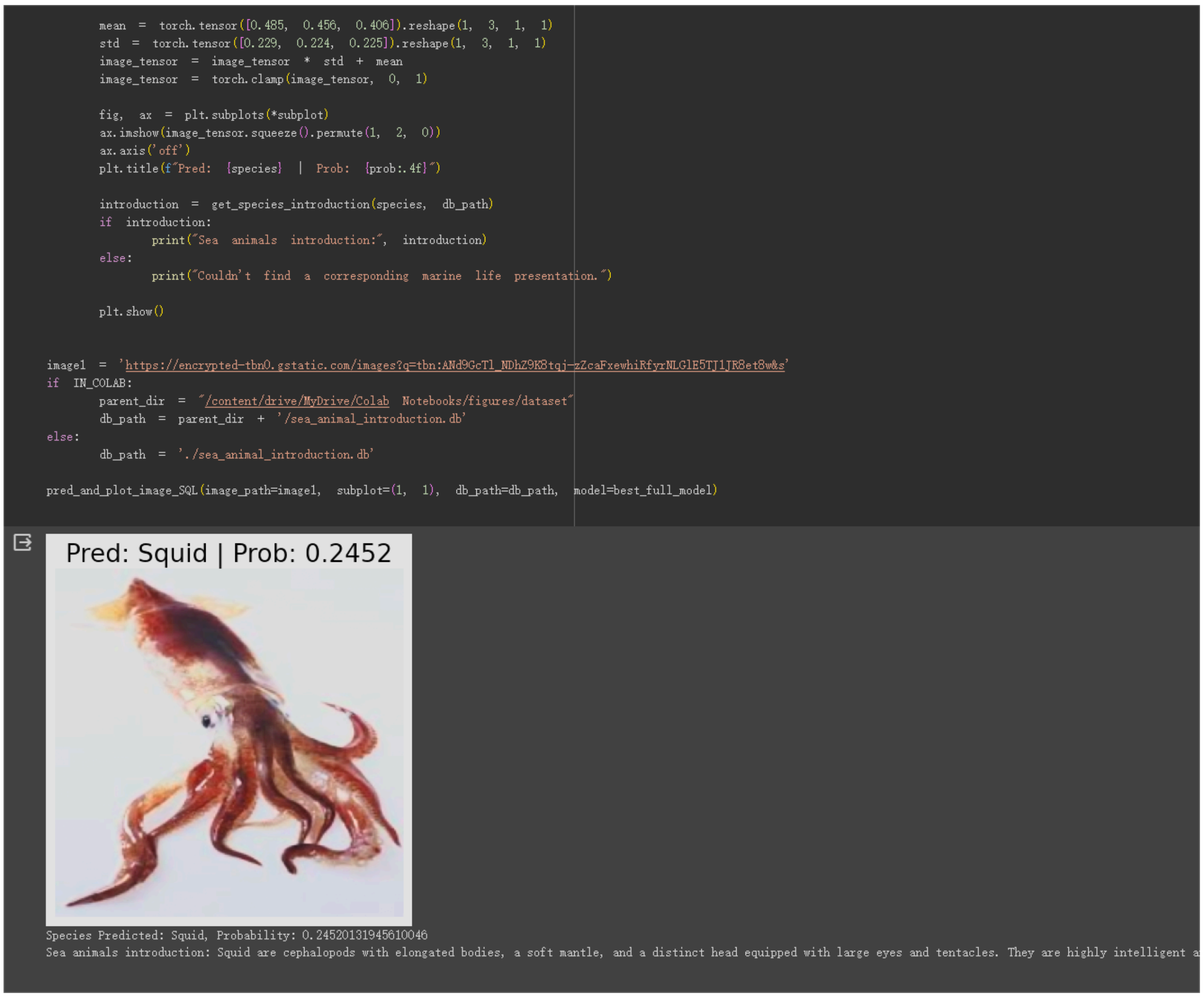
```
[ ] best_full_model = models.get(config.MODEL_NAME,
                                 num_classes=config.NUM_CLASSES,
                                 checkpoint_path=os.path.join(full_model_trainer.checkpoints_dir_path, "ckpt_best.pth"))
```

```
[ ]
    if isinstance(best_full_model, torch.nn.Module):
        print("Model loaded successfully.")
        best_full_model.eval()
    else:
        print("Model loading failed.")
```

```
    Model loaded successfully.
```



```python
[ ] import sqlite3
    import matplotlib.pyplot as plt
    import torch
    from torchvision import transforms
    from PIL import Image
    import requests
    from io import BytesIO


    def get_species_introduction(species_name, db_path):
        conn = sqlite3.connect(db_path)
        cursor = conn.cursor()
        cursor.execute("SELECT Introduction FROM sea_animal WHERE name = ?", (species_name,))
        result = cursor.fetchone()
        cursor.close()
        conn.close()
        return result[0] if result else None


    def preprocess_image(image_url):
        try:
            response = requests.get(image_url)
            if response.status_code == 200:
                image = Image.open(BytesIO(response.content)).convert('RGB')
            else:
                print("Failed to download image")
                return None
        except Exception as e:
            print(f"Error loading image: {e}")
            return None


        transform = transforms.Compose([
            transforms.Resize((224, 224)),
            transforms.ToTensor(),
            transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
        ])
        image_tensor = transform(image).unsqueeze(0)
        return image_tensor


    def pred_and_plot_image_SQL(image_path, subplot, db_path, model):
        image_tensor = preprocess_image(image_path)
        if image_tensor is None:
            return

        best_full_model.eval()
        with torch.no_grad():
            outputs = model(image_tensor)
            _, preds = torch.max(outputs, 1)
            species = CLASS_NAMES[preds[0].item()]
            prob = torch.nn.functional.softmax(outputs, dim=1)[0][preds[0]].item()

        print(f"Species Predicted: {species}, Probability: {prob}")
```



```
        mean  =  torch.tensor([0.485,  0.456,  0.406]).reshape(1,  3,  1,  1)
        std  =  torch.tensor([0.229,  0.224,  0.225]).reshape(1,  3,  1,  1)
        image_tensor  =  image_tensor  *  std  +  mean
        image_tensor  =  torch.clamp(image_tensor,  0,  1)

        fig,  ax  =  plt.subplots(*subplot)
        ax.imshow(image_tensor.squeeze().permute(1,  2,  0))
        ax.axis('off')
        plt.title(f"Pred:  {species}  |  Prob:  {prob:.4f}")

        introduction  =  get_species_introduction(species,  db_path)
        if  introduction:
                print("Sea  animals  introduction:",  introduction)
        else:
                print("Couldn't  find  a  corresponding  marine  life  presentation.")

        plt.show()


image1  =  'https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTl_NDhZ9K8tqj-zZcaFxewhiRfyrNLGlE5TJ1JR8et8w&s'
if  IN_COLAB:
        parent_dir  =  "/content/drive/MyDrive/Colab  Notebooks/figures/dataset"
        db_path  =  parent_dir  +  '/sea_animal_introduction.db'
else:
        db_path  =  './sea_animal_introduction.db'

pred_and_plot_image_SQL(image_path=image1,  subplot=(1,  1),  db_path=db_path,  model=best_full_model)
```

```
Species Predicted: Squid, Probability: 0.24520131945610046
Sea animals introduction: Squid are cephalopods with elongated bodies, a soft mantle, and a distinct head equipped with large eyes and tentacles. They are highly intelligent a
```

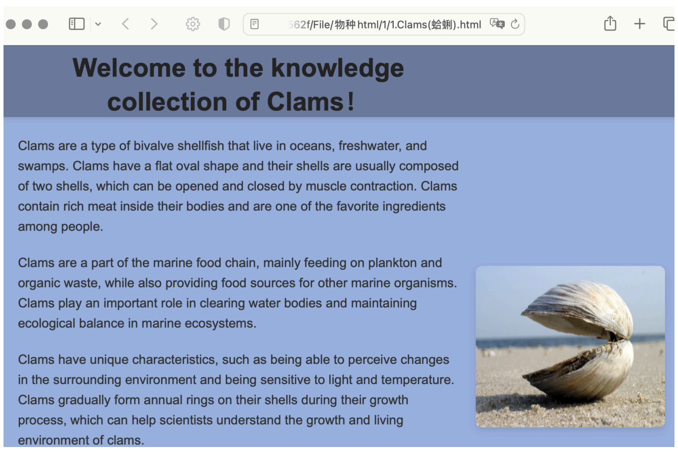
When the analysis and scanning results are for the corresponding marine species, the corresponding marine life information will pop up.

3.4.3 Web pages and jumps

In the construction and jump design of web pages, we use Flask, a lightweight Python web application framework that can quickly build web applications.

3.4.4 HTML

We designed a page introduction of the 23 types of marine life we predicted and stored it in the template folder of the Flask application for Python and JavaScript to implement web page jumps.

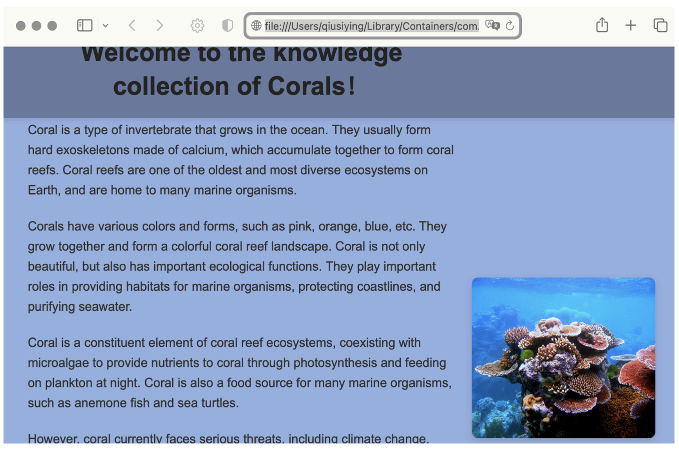


Welcome to the knowledge collection of Clams!

Clams are a type of bivalve shellfish that live in oceans, freshwater, and swamps. Clams have a flat oval shape and their shells are usually composed of two shells, which can be opened and closed by muscle contraction. Clams contain rich meat inside their bodies and are one of the favorite ingredients among people.

Clams are a part of the marine food chain, mainly feeding on plankton and organic waste, while also providing food sources for other marine organisms. Clams play an important role in clearing water bodies and maintaining ecological balance in marine ecosystems.

Clams have unique characteristics, such as being able to perceive changes in the surrounding environment and being sensitive to light and temperature. Clams gradually form annual rings on their shells during their growth process, which can help scientists understand the growth and living environment of clams.



Welcome to the knowledge collection of Corals!

Coral is a type of invertebrate that grows in the ocean. They usually form hard exoskeletons made of calcium, which accumulate together to form coral reefs. Coral reefs are one of the oldest and most diverse ecosystems on Earth, and are home to many marine organisms.

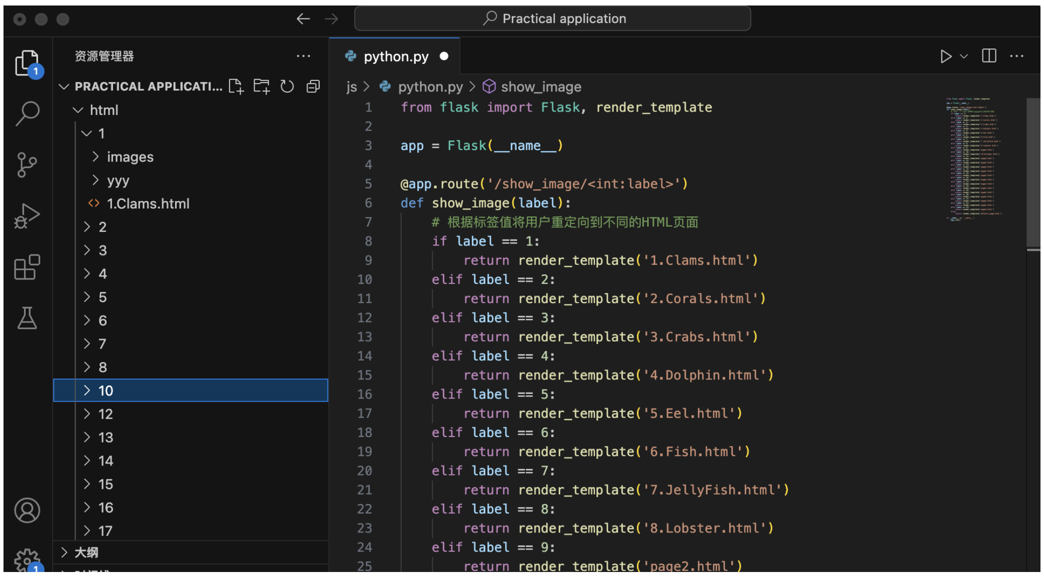
Corals have various colors and forms, such as pink, orange, blue, etc. They grow together and form a colorful coral reef landscape. Coral is not only beautiful, but also has important ecological functions. They play important roles in providing habitats for marine organisms, protecting coastlines, and purifying seawater.

Coral is a constituent element of coral reef ecosystems, coexisting with microalgae to provide nutrients to coral through photosynthesis and feeding on plankton at night. Coral is also a food source for many marine organisms, such as anemone fish and sea turtles.

However, coral currently faces serious threats, including climate change.

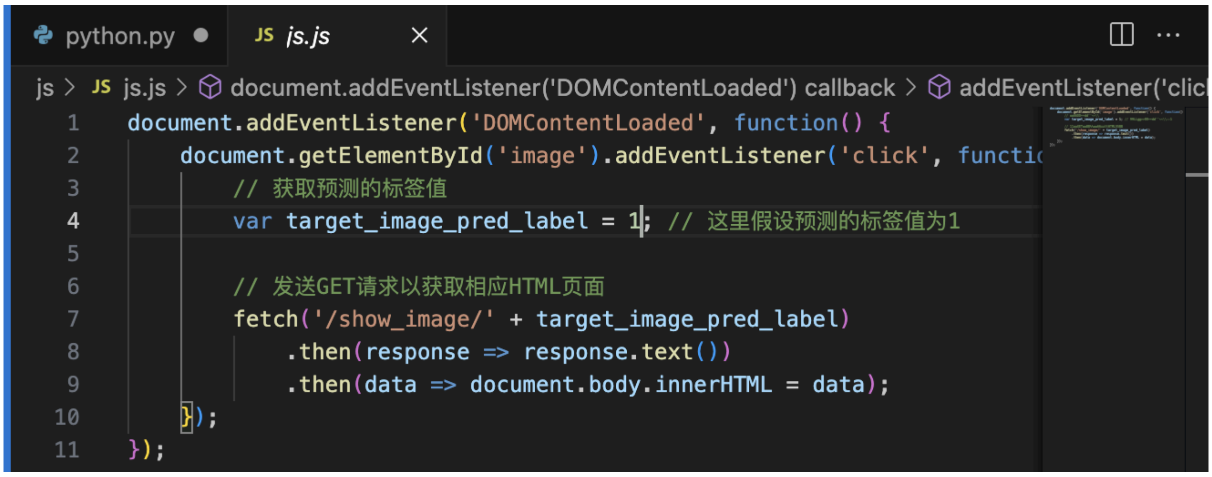
3.4.5 Python and JavaScript

We use python and JavaScript to implement web page jumps. The Python code creates jumps to marine life with different labels, and will jump to the corresponding web page according to different types of marine life.



```python
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/show_image/<int:label>')
def show_image(label):
    # 根据标签值将用户重定向到不同的HTML页面
    if label == 1:
        return render_template('1.Clams.html')
    elif label == 2:
        return render_template('2.Corals.html')
    elif label == 3:
        return render_template('3.Crabs.html')
    elif label == 4:
        return render_template('4.Dolphin.html')
    elif label == 5:
        return render_template('5.Eel.html')
    elif label == 6:
        return render_template('6.Fish.html')
    elif label == 7:
        return render_template('7.JellyFish.html')
    elif label == 8:
        return render_template('8.Lobster.html')
    elif label == 9:
        return render_template('page2.html')
```

The `fetch` request in the JavaScript code sends the correct path to the route of the Python Flask application and sets the path of the `fetch` request to match the route defined in the Python Flask application. When the routes match, when the project program is implemented and the request is sent through python, the web page jump can be realized.


python.py
js.js
js > js.js > document.addEventListener('DOMContentLoaded') callback > addEventListener('clic

```javascript
document.addEventListener('DOMContentLoaded', function() {
    document.getElementById('image').addEventListener('click', functi
        // 获取预测的标签值
        var target_image_pred_label = 1; // 这里假设预测的标签值为1

        // 发送GET请求以获取相应HTML页面
        fetch('/show_image/' + target_image_pred_label)
            .then(response => response.text())
            .then(data => document.body.innerHTML = data);
    });
});
```

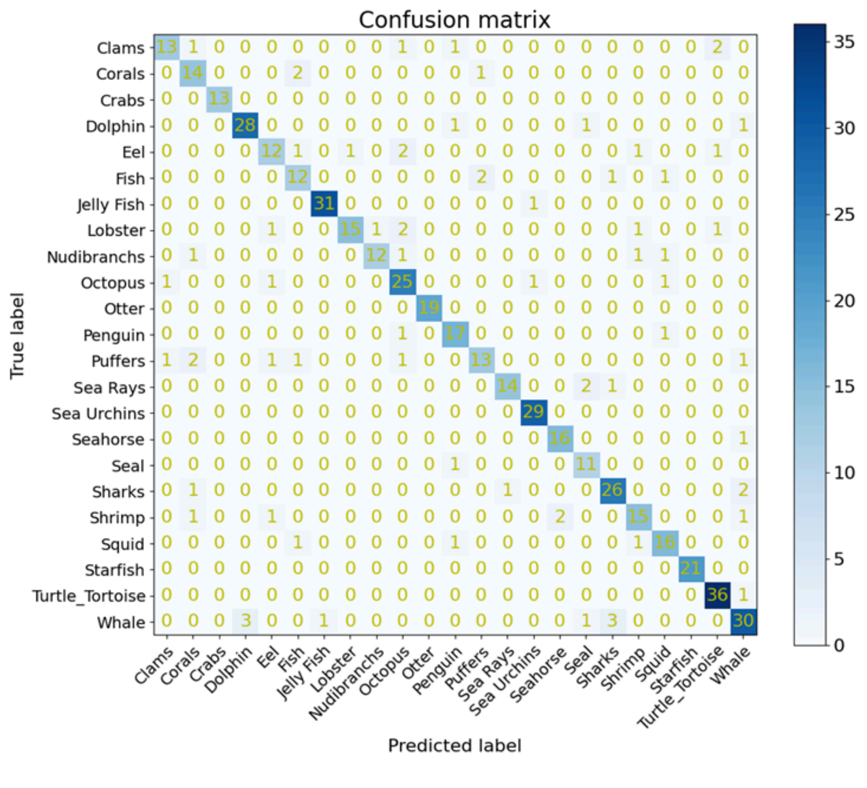
4. Experimental Study

4.1Experimental procedure

First, we installed the Super Gradients deep learning library, which focuses on classification, recognition tasks, and target detection in images, and uses the pre-trained model "regnetY800" to help us improve the efficiency and performance of model training. Second, we set the parameter ratio to split the data into 80% training, 10% validation, and 10% testing. Third, by randomly looking at a set of images and their category labels, check whether the dataset is loaded correctly and whether the annotation of the images after identification is accurate, and again check the distribution of the number of each marine animal category. Fourth, use the data enhancement function to define a series of transformations to be applied to the training set to improve the generalization ability of the model. Finally, we call the training method of the Trainer instance, passing in the model, training parameters, training data loader, and validation data loader. During this stage, the model uses training data to update its weights and validation data to evaluate accuracy and other performance.

4.2Evaluation method

We randomly selected five photos of marine life from the Internet (excluding data sets) to evaluate the accuracy of the deep learning model. All five pictures accurately matched the actual image types and had a high accuracy rate. In addition to this, we also use a confusion matrix, which shows the relationship between the actual categories and the model-predicted categories. We find that values along the diagonal represent the number of correct classifications by the model, while values off the diagonal represent the number of incorrect predictions. The confusion model not only shows the accuracy of each class but also visualizes the classification performance of each class, which helps evaluate the model.



True label \ Predicted label	Clams	Corals	Crabs	Dolphin	Eel	Fish	Jelly Fish	Lobster	Nudibranchs	Octopus	Otter	Penguin	Puffers	Sea Rays	Sea Urchins	Seahorse	Seal	Sharks	Shrimp	Squid	Starfish	Turtle_Tortoise	Whale
Clams	13	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	2	0
Corals	0	14	0	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Crabs	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Dolphin	0	0	0	28	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
Eel	0	0	0	0	12	1	0	1	0	2	0	0	0	0	0	0	0	0	1	0	0	1	0
Fish	0	0	0	0	0	12	0	0	0	0	0	0	2	0	0	0	0	1	0	1	0	0	0
Jelly Fish	0	0	0	0	0	0	31	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Lobster	0	0	0	0	1	0	0	15	1	2	0	0	0	0	0	0	0	0	1	0	0	1	0
Nudibranchs	0	1	0	0	0	0	0	0	12	1	0	0	0	0	0	0	0	0	1	1	0	0	0
Octopus	1	0	0	0	1	0	0	0	0	25	0	0	0	0	1	0	0	0	0	1	0	0	0
Otter	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0
Penguin	0	0	0	0	0	0	0	0	0	1	0	17	0	0	0	0	0	0	0	1	0	0	0
Puffers	1	2	0	0	1	1	0	0	0	1	0	0	13	0	0	0	0	0	0	0	0	0	1
Sea Rays	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	0	2	1	0	0	0	0	0
Sea Urchins	0	0	0	0	0	0	0	0	0	0	0	0	0	0	29	0	0	0	0	0	0	0	0
Seahorse	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	1
Seal	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	11	0	0	0	0	0	0
Sharks	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	26	0	0	0	0	2
Shrimp	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0	15	0	0	0	1
Squid	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	16	0	0	0
Starfish	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21	0	0
Turtle_Tortoise	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	1
Whale	0	0	0	3	0	0	1	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	30

4.3Analysis of results

The results show that the model performed well on the test set, with an accuracy of about 86.05%. It is worth mentioning that the Top-5 accuracy is as high as 97%, which means that even if the first choice of the model is incorrect, the correct answer will still be among the top five most likely options. In addition, the average loss of the model on the test set is about 1.43. The smaller the loss value, the closer the model's predictions are to the actual labels and the better the performance.

5. Conclusion

Our project aims to solve the difficult problem aquarium visitors face: to help them accurately identify specific animals among many unfamiliar animal species in an aquarium through our system so that they can recognise and understand various animals more efficiently.

Our group members completed our system through a series of steps such as developing a proposal, data collection, importing libraries, data processing, Data Augmentation, Training model, Test model, connecting SQLite, HTML design and so on. Combining computer vision and deep learning algorithms， after nearly 30 rounds of training, our model reached a high level of accuracy. Now, users can quickly get the names and detailed descriptions of marine organisms just by taking or scanning photos.

The system helps improve visitors' knowledge and awareness of sea animals and promotes awareness of sea life conservation. For researchers and conservation organizations, the system can also provide valuable data and insights for animal population monitoring, behavioural studies, and the development of conservation measures.

In the future, we can further improve and optimize the system's performance to increase accuracy and response time. As well as expanding our marine life database to recognise a greater diversity of marine organisms.

Workload Distribution

Members	Train Model& Experimental Study	SQL query& Template	Market Application &Interface	Report	Slide
Luo Fan 23435968	√		√	√	√
Zhu Xiaoyu 23422041	√			√	√
Zhang Sining 23451637		√		√	√
Pan Yier 23462671		√		√	√
Qiu Siying 23467274			√	√	√
Jiang Nanhezi 23473452			√	√	√

Reference:

1.“How to capture a image from webcam in Python?” (2023,Jan 03). How to capture a image from webcam in Python? - GeeksforGeeks. Retrieved from:https://www.geeksforgeeks.org/how-to-capture-a-image-from-webcam-in-python/

2.“Python OpenCV: Capture Video from Camera” (2023,Jan 04). Python OpenCV: Capture Video from Camera - GeeksforGeeks. Retrieved from:
https://www.geeksforgeeks.org/python-opencv-capture-video-from-camera/