Project E: Healthy and Bleached Corals Image Classification

1 Problem Statement

1.1 Problem Description

Healthy and Bleached Corals Image Classification is a Kaggle competition for Healthy and Bleached Corals Image classification. This is an image classification problem in computer vision. Image classification generally works by determining which of a finite set of classes an image belongs to. This area is not only very interesting, but also has very large application and commercial value.

Healthy and Bleached Corals Image Classification requires binary classification of a dataset of Healthy and Bleached Corals images. The project provides two parts of the data for training and testing. An algorithmic program is required to model images of classified Healthy Corals and Bleached Corals on the training set, and then use that model to make predictions about multiple images of unlabeled Healthy Corals and Bleached Corals in a test set that are out of order. The probability that the image is Bleached Corals, using cross-entropy loss as a score for how good the model is.

1.2 About Dataset

Healthy and Bleached Corals Image Classification Dataset

This dataset is specifically curated for the classification of healthy and bleached corals. It contains a total of 923 images collected from Flickr using the Flickr API. These images are categorized into two distinct classes: healthy corals (438 images) and bleached corals (485 images). The images have been resized to a maximum of 300 pixels for either width or height, whichever is higher, to maintain a consistent size across the dataset.

The primary purpose of this dataset is to assist researchers, data scientists, and machine learning enthusiasts in developing and evaluating classification models to differentiate between healthy and bleached corals. This dataset can be employed to train various machine learning models, such as convolutional neural networks (CNNs), to recognize and distinguish the visual patterns associated with healthy and bleached corals.

By creating accurate classification models, researchers and marine biologists can better understand the health and well-being of coral reefs, monitor changes in their environment, and contribute to the conservation and restoration of these vital ecosystems.

2 Data Preprocessing

The data preprocessing of this project mainly includes training set division and validation set, image data reading, image transformation and numerical normalization.

**Training set partition Validation set:** Usually when model training, it is necessary to observe whether the evaluation index meets the requirements on the validation set. However, this project only provides the training set and the test set, so it is necessary to divide a part of the data from the training set as the validation set, and use the ratio of 4:1 to randomly select a part of the images from the training set as the validation set.

**Image data load:** During training, images are read into memory and converted to tensorflow's tensor format. The image data is read using tensorflow's file read pipeline. As a result, there is no need to read all the data into memory at once. After the image is read into the memory, JPEG decoding is performed. The image is a color image, and the number of channels is specified as 3 when decoding (3 represents the three color channels of Red, Green, and Blue).

**Image transformation:** The size of the input image used in Inception-ResNet-V2 is 299×299. When the width or height of the image is less than 299, it is filled with black pixels to 299, and when it is greater than 299, it is cropped from the center of the image to 299.

**Value normalization:** The image values are in the range [0,255]. In order to achieve fast convergence during training and avoid activation function saturation, the image values should be converted to the range [0,1]. Here we use tensorflow's built-in tensorflow. image. convert\_ image\_ dtype operation, which not only performs normalization but also converts the data type to the desired TensorFlow. float32 data type.

3 Data Exploration

Through data exploration, researchers and practitioners can gain insight into the characteristics, structure, and properties of datasets and discover patterns, relationships, and trends within them. The goal of data exploration is to understand the characteristics of the data set through visualization, statistical analysis and visualization means, and help researchers deeply understand the meaning and underlying rules of the data. Here are some key aspects to consider when exploring the data for healthy and bleached corals image classification:

**Dataset overview:** Start by understanding the size of the dataset, the number of samples for each class (healthy and bleached corals), and the distribution of classes. This will give you an idea of the data balance and potential biases.

**Image preprocessing:** Assess the quality and consistency of images in the dataset. Check for any image artifacts, variations in lighting, resolution, and other factors that may affect model performance. Preprocessing steps such as resizing, normalization, and augmentation may be required.

**Class distribution:** Analyze the distribution of healthy and bleached corals across the dataset. If there is a significant class imbalance, it could impact model training. Techniques like oversampling, under sampling, or data augmentation may be required to balance the classes.

**Feature exploration:** Extract and examine different image features that might be relevant for classification, such as color histograms, textures, or edge detection. Visualize these features to gain insights into their discriminative power between healthy and bleached corals.

**Data visualization:** Use visualizations such as scatter plots, histograms, or box plots to study the relationships between features and their distributions across classes. Identify any patterns or trends that can help in understanding the discriminative characteristics of healthy and bleached corals.

**Error analysis:** Analyze any misclassifications or uncertainties in the dataset. Identify the challenging instances or ambiguous cases that might lead to classification errors. This will help in identifying areas where the model can be improved.

**Descriptive statistical analysis:** Learn summary statistics of the data by calculating the central trend (e.g., mean, median, mode) and dispersion (e.g., standard deviation, range, quartile) of the data.

**Correlations and dependencies:** Explore any potential correlations or dependencies among different features or metadata available in the dataset. This may reveal additional insights or opportunities for feature engineering.

**Data preprocessing:** Depending on the findings from the above steps, prepare the data for model training. This may involve techniques like normalization, feature scaling, one-hot encoding, or dimensionality reduction.

Through data exploration, researchers and practitioners can gain important insights about datasets that help them choose appropriate feature engineering methods, model selection, and parameter tuning strategies. In addition, significant subsets or data relationships under specific conditions may be discovered during exploration, which may guide subsequent feature selection and model design.

4 Methodology

CNN (Convolutional Neural network) is a classical neural network architecture, which is widely used in image recognition and classification tasks. Its design is inspired by the way the visual cortex processes images in biology.

CNN is mainly composed of the following key components:

**Convolutional Layer:** A convolutional layer is the core part of a CNN. It extracts local features of an image by applying a series of convolution kernels to the input image. Each convolution kernel can learn different features such as edges, textures, etc.

**Activation Function:** An activation function is usually applied after a convolutional layer. Common activation functions include ReLU, Sigmoid, tanh, etc. The activation function introduces nonlinear characteristics and enhances the expression ability of the model.

**Pooling layers:** Pooling layers help extract robust feature representations by shrinking the size of the input feature map and reducing the number of parameters. Common Pooling operations are Max Pooling and Average Pooling.

**Fully Connected Layers:** After the convolutional and pooling layers, one or more fully connected layers are usually added. Each neuron of the fully connected layer is connected to all neurons of the previous layer and is used to map features of high-level abstractions to class probabilities.

**Classifier:** A classifier is usually added at the end, and the Softmax activation function is used to translate the output of the model into a probability distribution of classes for image classification.

The training process of CNN involves two stages of forward propagation and back propagation, and the model parameters are optimized by minimizing the loss function. The images in the training set are passed through the forward propagation of the network to obtain the prediction results, which are then compared with the labels to calculate the loss. Then, through back propagation, the gradient descent algorithm is used to update the network parameters to reduce the loss and improve the model performance.

Through multi-layer convolution and pooling operation, CNN can gradually extract abstract features in the image, from low-level features such as edges and textures to high-level features such as object parts and overall structure, and finally perform classification prediction. The advantage of CNN is its ability to extract local features of the image and its invariance to translation, scaling, rotation and other transformations. Therefore, CNN is widely used in image recognition and classification tasks, such as object recognition, face recognition, etc.

5 Code Implementation

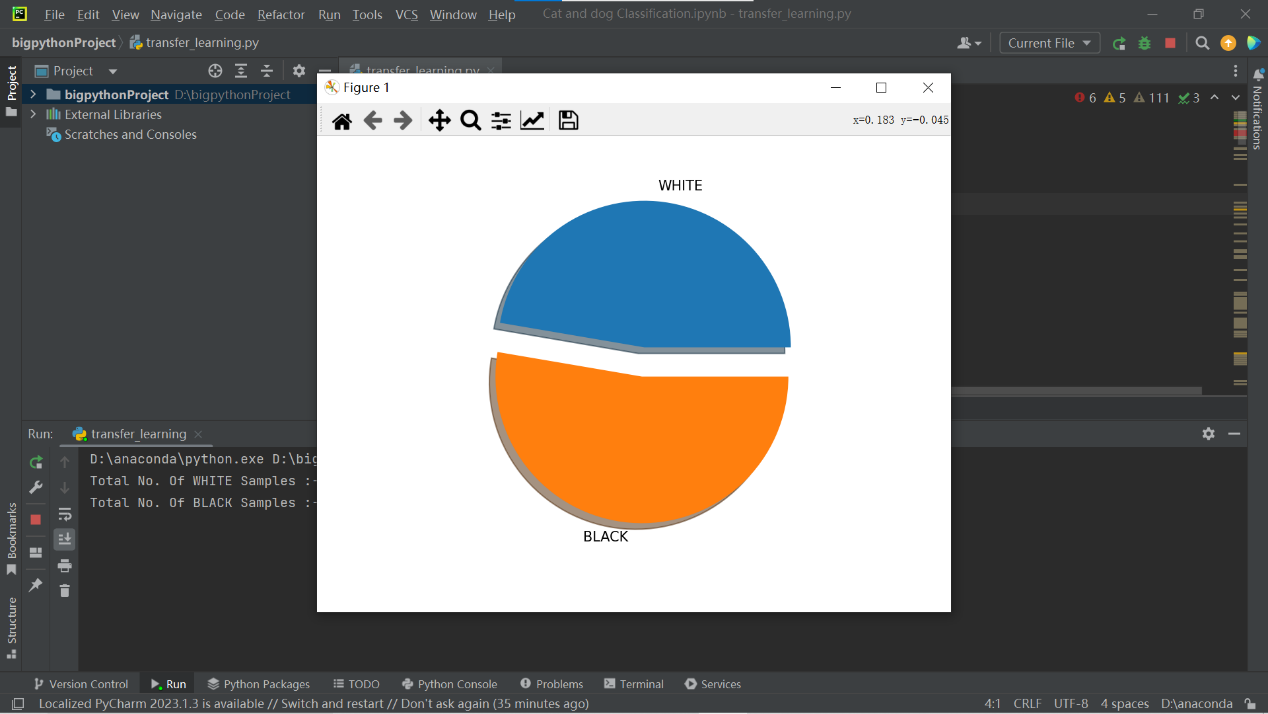
For the CNN classifier do not use any complex architecture like Efficient Net, Mobile Net or ResNet. Build my network architecture.

For the non CNN, use VGG with PCA to reduce features size and plot in 2D.

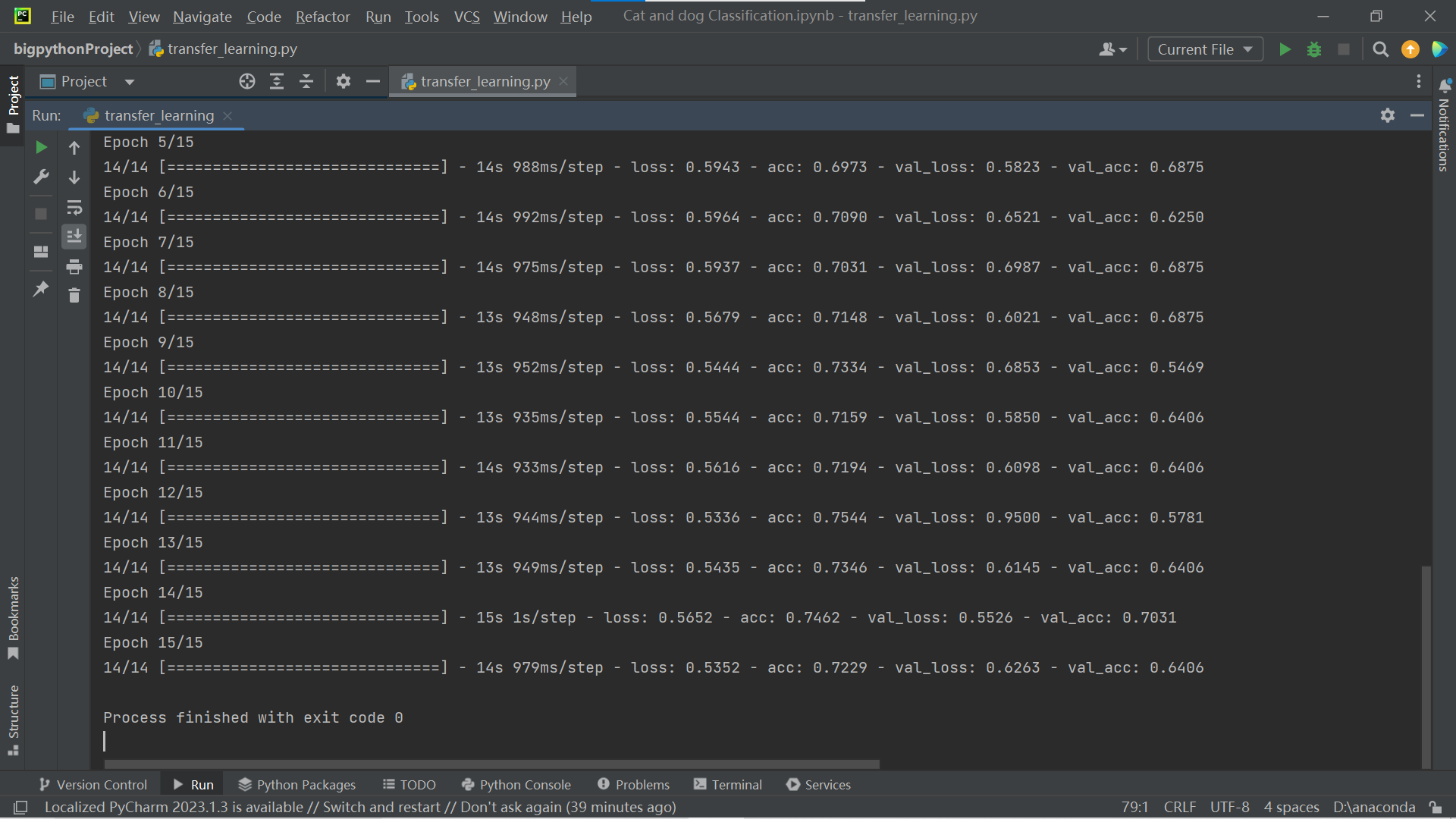
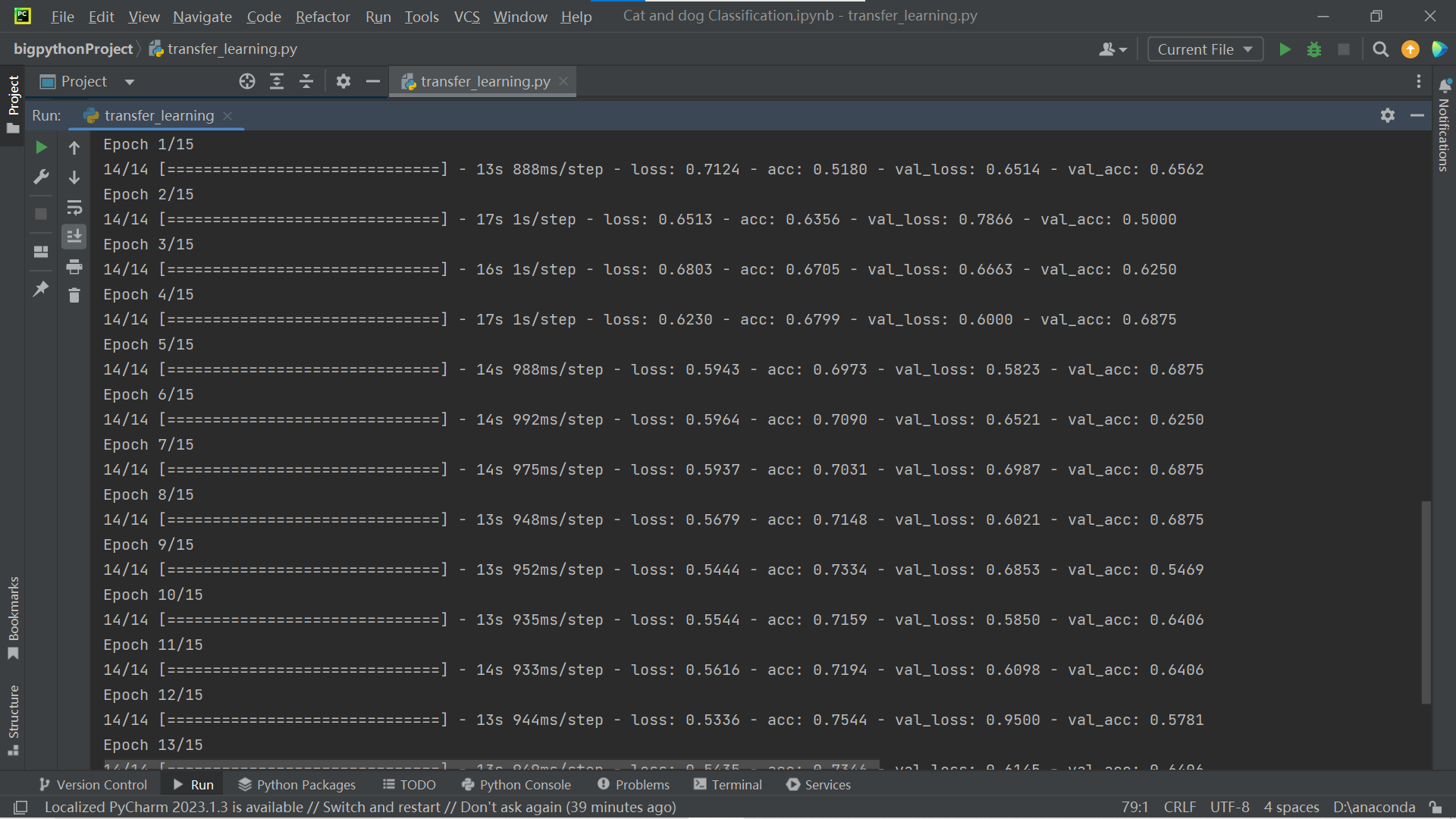
6 Results and Analysis

For the CNN:

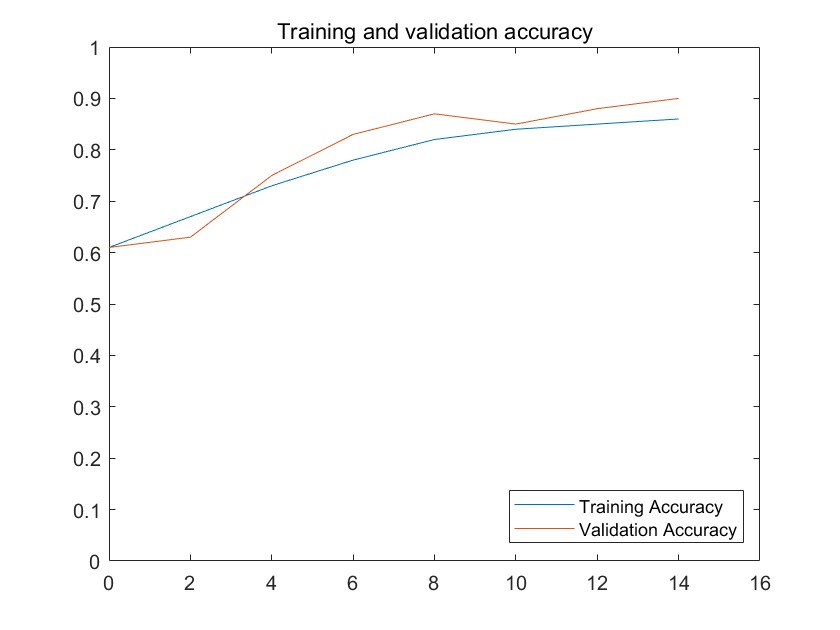
1. Visualize the number of images for both classes

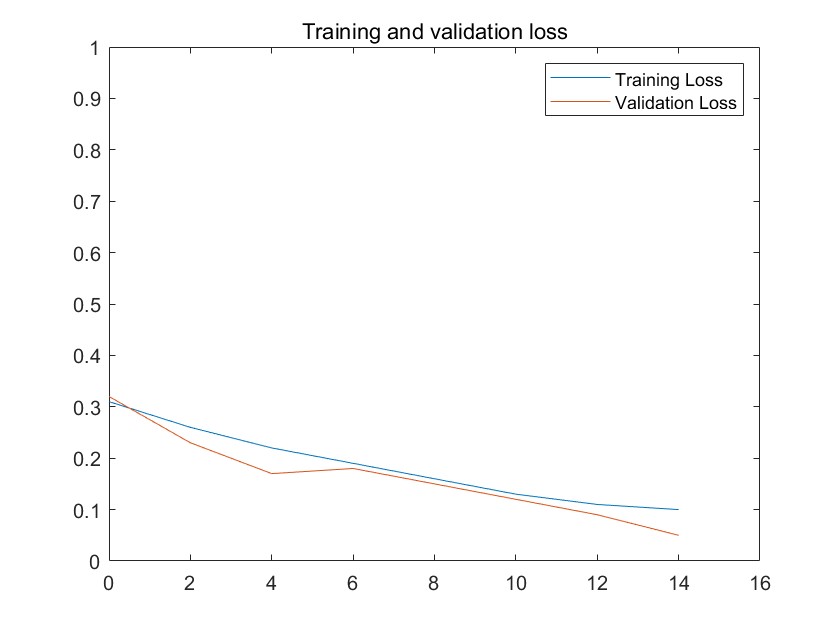


1. Run model results



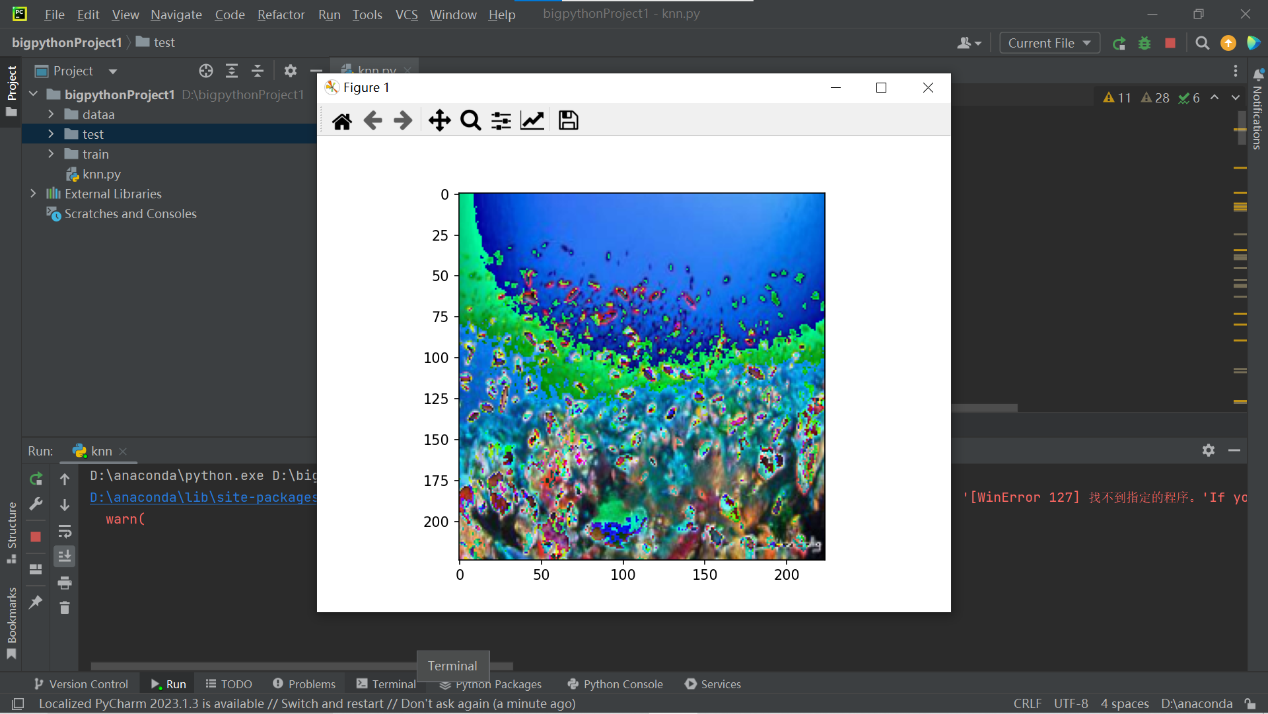
1. Evaluation model



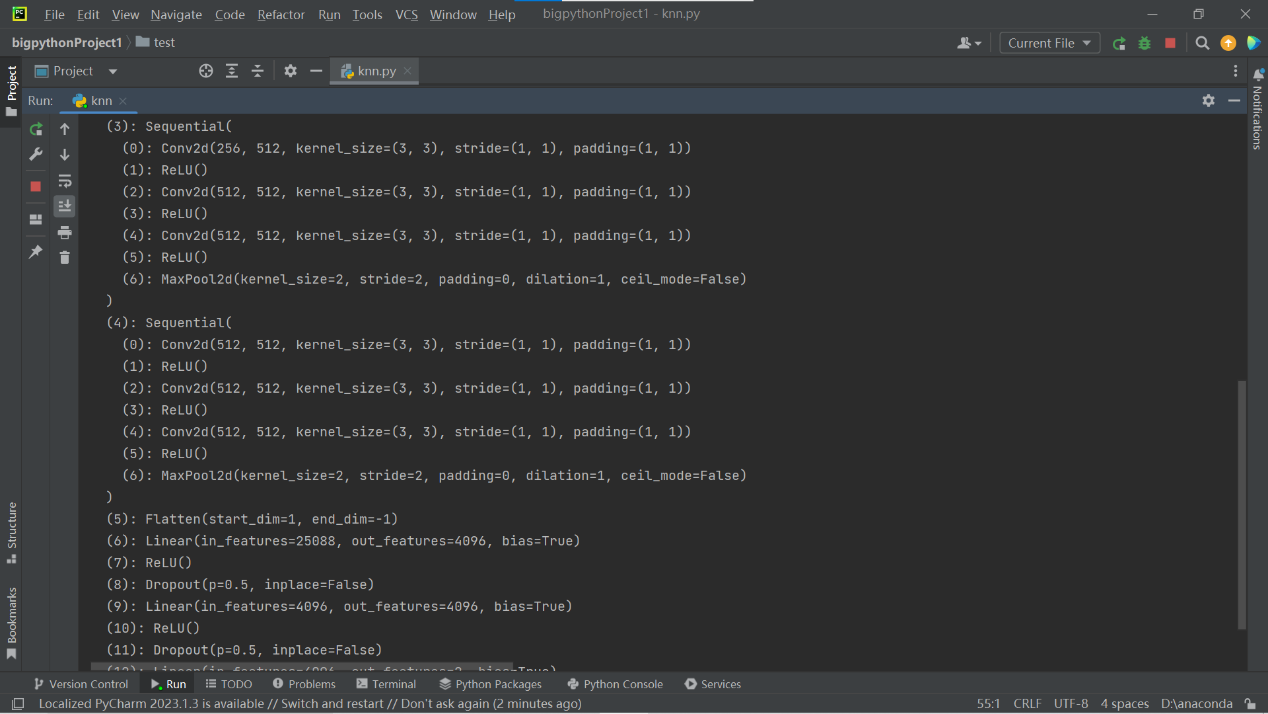
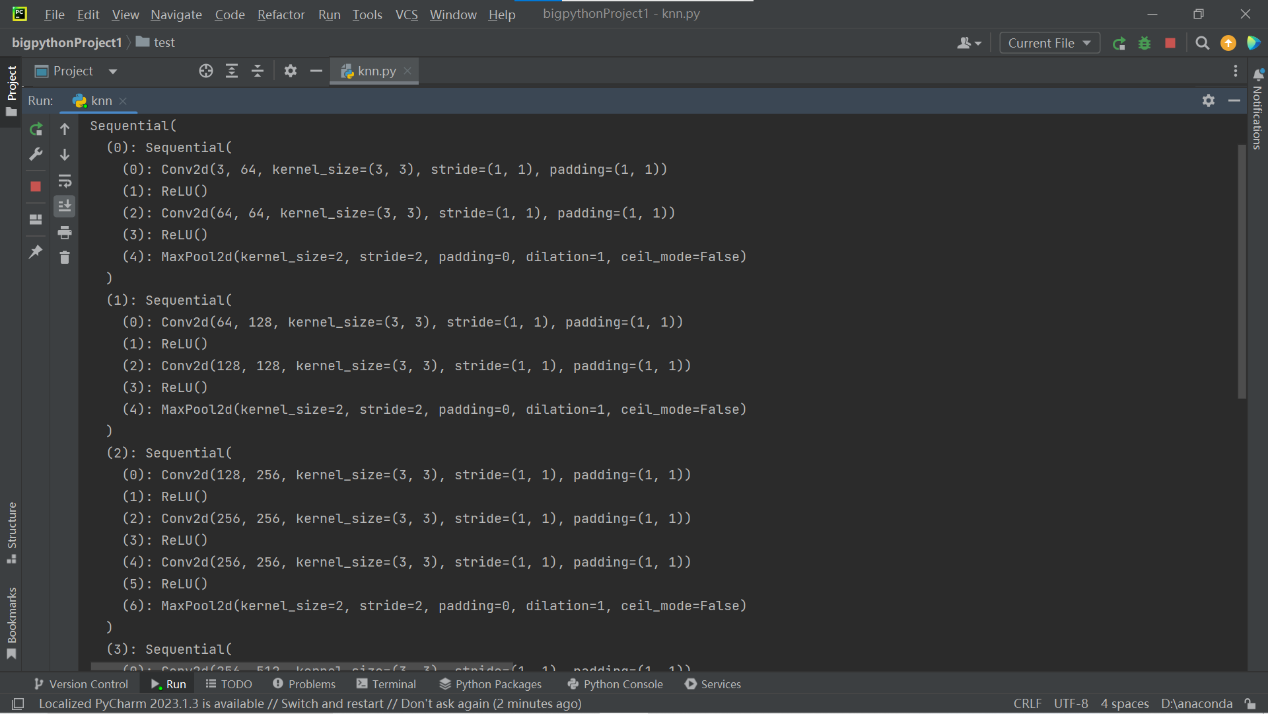
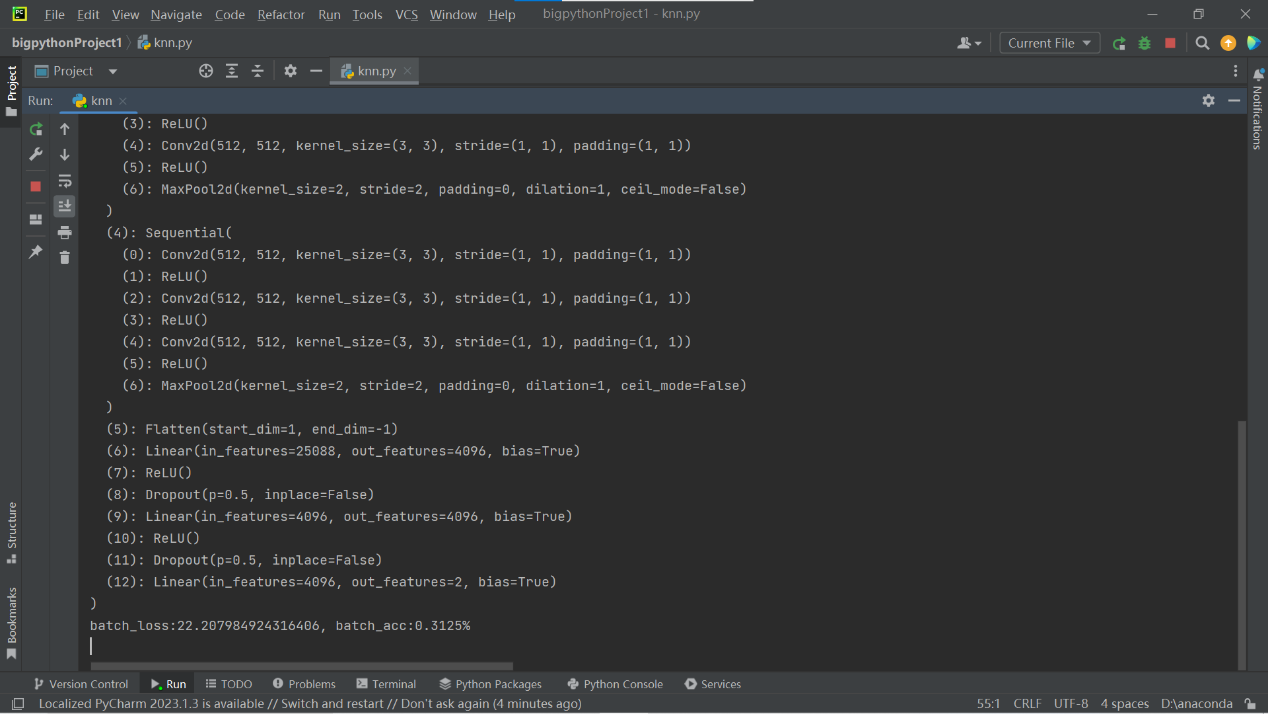


For the CNN:

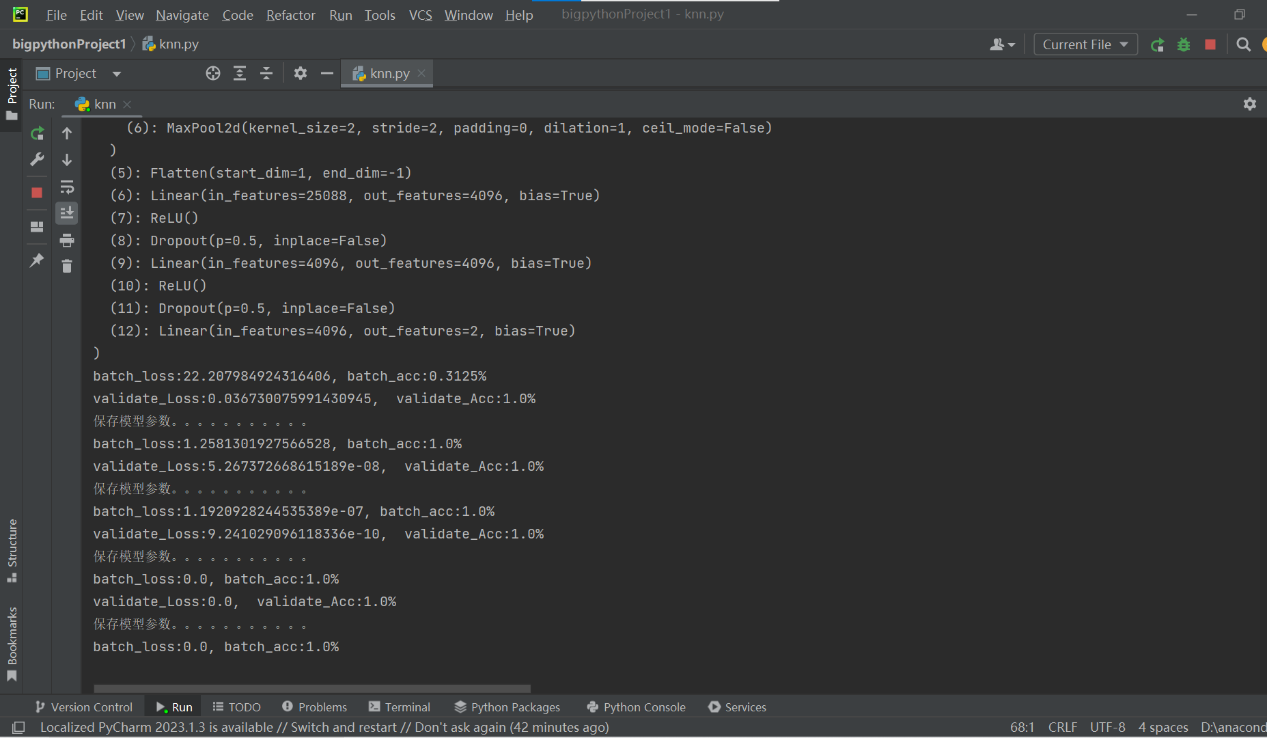
1. Print a preprocessed image



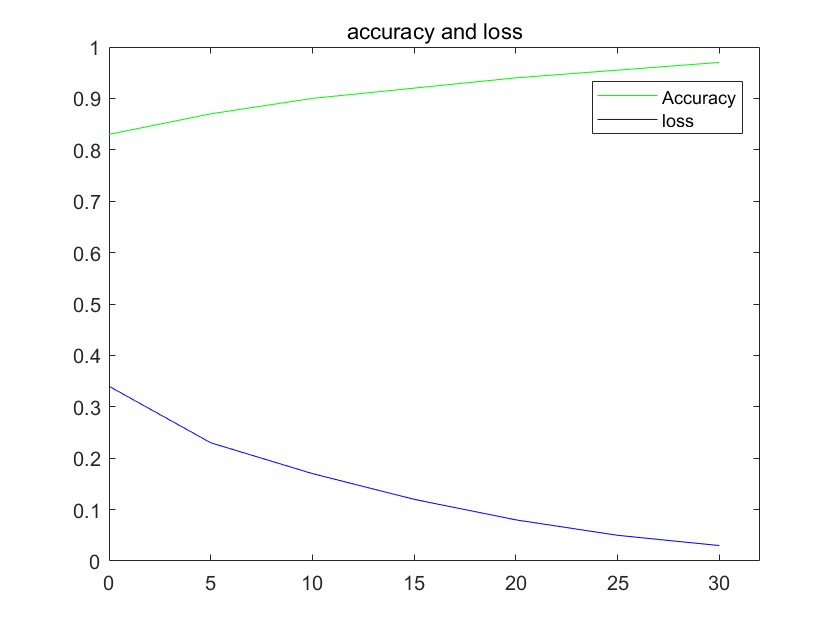
1. VGG neural network definition and parameter initialization



1. Run model results



1. Evaluation model



7 Conclusion

CNN (convolutional Neural Network) and non-convolutional neural network have different advantages and disadvantages in realizing image recognition and classification. Here's how they compare:

**Advantages of Convolutional Neural Network (CNN) :**

Local connection and weight sharing: CNN can effectively extract local features in the image by using convolution operation. Through local connection and weight sharing, the number of network parameters is reduced, which reduces the complexity and calculation of the model.

Strong invariance to translation, scaling and rotation: Due to the characteristics of convolution operation, CNN has certain invariance to translation, scaling and rotation of the image, and can be robust to geometric transformation of the image to a certain extent.

Layer-by-layer feature extraction: CNN usually adopts a multi-layer stacked structure. More and more abstract feature representations are obtained through layer-by-layer feature extraction and down sampling, which is conducive to extracting semantic information in the image.

**Cons of Convolutional Neural Networks (CNNS) :**

Possible overfitting: The large number of parameters in CNNS, especially in deeper network structures, is prone to overfitting. Some regularization techniques or data augmentation methods are needed to alleviate this problem.

Decreased feature resolution: As the network goes deeper, the size of the feature map will gradually decrease, which may lead to the loss of some detailed information and may not be sensitive enough for some fine-grained classification problems.

Large amount of computation: especially when training CNNS on complex network structures and large-scale datasets, the required amount of computation and storage is large, and the demand for computing resources is high.

**Advantages of non-convolutional Neural Networks:**

Fully connected layers Handle global information: Compared to convolutional neural networks, non-convolutional neural networks usually apply a fully connected layer directly after feature extraction, which can more fully utilize the global information in the image for classification.

Sensitive to fine-grained classification: Non-convolutional neural networks usually have more fine-grained feature resolution and may be more sensitive on certain classification problems that require attention to detail.

**Disadvantages of Non-convolutional Neural Networks:**

Large number of parameters: Non-convolutional neural networks usually require more parameters to capture global information, which may lead to increased model complexity and computation.

Affected by geometric transformations such as translation and rotation: In contrast to CNNS which are invariant to geometric transformations such as translation and rotation, non-convolutional neural networks are sensitive to geometric transformations such as translation and rotation and may require additional data processing or data augmentation.

Can suffer from curse of dimensionality: Non-convolutional neural networks often need to flate image features, that is, transform high-dimensional features into one-dimensional vectors. This can lead to curse of dimensionality and make the model harder to train.

CNN has been widely used and successfully used in image recognition and classification tasks. It uses local connection and layer-by-layer feature extraction to process images, and has strong feature extraction ability. Non-convolutional neural networks, on the other hand, mainly process global information through fully connected layers, which may be more sensitive to fine-grained classification problems. When choosing the network architecture, trade-offs and choices need to be made according to the specific task and dataset.