Chen:

For our project, we tried two different method to achieve the tasks, that is, Detection and Classification. The first idea that I come up with is to remove the back ground of the pics. And find the leftmost, upmost, rightmost and downmost pixels which are not totally transparent (alpha-channel don’t equal to 255). This finishes the detecting task. And after that we could just use K-NN to complete the classify task. And the second plan also the final chosen plan is that we train a yolov8 model from scratch. We first made some augmentation to the pics which are in the train folder. And split the dataset to 450 train pics and 50 valid pics and use the provided valid pics as test pics.

Long:

I have tried three different segmentation models. The Deeplabv3 didn’t perform well and MobileViT returned a set of whole black pics. I finally chose yolov8-seg to finish the background removing task, and the provided labels are not satisfy the model, so I have to reformat it. To split the long list into dictionaries and extract the bboxes, reformat it to [x\_center, y\_center, width, height], and normalize the data between 0-1. And then us a GPU server to train the model and let the model make predictions on the valid set.

Puff:

I have done the KNN match, here is the steps:

1.Distance Calculation: Calculate the distances between the test sample and each sample in the training set.

2.K Selection: Determine the number K of nearest neighbors to be used for voting or averaging. Choosing an appropriate value for K is important and often requires cross-validation.

3.Neighbor Determination: Select the K samples with the shortest distances as the neighbors of the test sample.

4.Voting or Averaging: For classification problems, the labels of the K neighbors are used for voting to determine the predicted class of the test sample. For regression problems, the average of the labels of the K neighbors is taken as the predicted value of the test sample.

But after applying K-NN to the valid set, the acc is only about 53%, not a very good result. So we decided not to use the background removing – K-NN method.

Rui:

COSINE SIMILARITY:

In the KNN algorithm, cosine similarity is a commonly used measure of similarity that evaluates the similarity between two vectors. Cosine similarity measures the cosine value of the angle between two vectors in a multidimensional space. It can be used to compute similarity in domains such as text, images, and more.

The calculation of cosine similarity between two vectors can be done through the following steps:

1.Normalize the two vectors: To eliminate the influence of vector lengths in calculations, it is common to normalize the two vectors to unit vectors. This is achieved by dividing each vector by its length, making them both length.

2.Compute the dot product of the vectors: Multiply the normalized vectors element-wise and sum the results to obtain their dot product.

3.Calculate the cosine similarity: Divide the dot product by the product of the lengths of the vectors.

Caim:

The final method we decide to use is to finish the two tasks at the same time using yolov8.

After reading the document of yolo, we find that yolov8 could finish detection and classification at the same time by training the model with a correct class label.

At first we also have to reformat the labels, just split, normalize and put each line in the independent .txt files(Using python, source code will be provided).

Then we need to write a .yaml file to tell the model the shape of the dataset.

Then we just put the code and the dataset on the GPU server. After build up the environment, we could just start the training. For each epoch, the model took about 1 minute to learn and valid.

Long:

After training, the model will automatically save the last.pt and best.pt. We could choose the best.pt as the model and let the model make predictions and the valid set. The returned labels will be saved in the folder. In some cases the model will make prediction more than once for each picture, so we choose the highest confidence as the final result.