Sakka Mohamad-Mario 1241EB – Subgroup II

SOFTWARE ENGINEERING HW IV

RED PANDA DETECTION USING YOLOV5 AND ROBOFLOW

This report describes the end-to-end pipeline for training and evaluating a custom object detection model to detect red pandas. A small custom dataset was collected, labeled with Roboflow, and used to fine-tune the YOLOv5s model. The final model was evaluated on a held-out test set, and qualitative results are shown below.

1. ENVIRONMENT AND SETUP

The experiment was conducted in a Python environment (for example, Google Colab) using the official YOLOv5 repository from Ultralytics. The following commands were used to clone the repository and install dependencies:

```
!git clone https://github.com/ultralytics/yolov5
%cd yolov5

!pip install -r requirements.txt
!pip install roboflow
```

The first command clones the YOLOv5 codebase, the second changes into the cloned directory, and the remaining commands install the core YOLOv5 dependencies along with the Roboflow Python package used to download the labeled dataset.

2. DATASET CREATION AND LABELING WITH ROBOFLOW

A custom dataset of red panda images was collected from the web and then uploaded to Roboflow for labeling. Each image contains one or more red pandas, and all bounding boxes were annotated with a single class: “Red Panda”. Roboflow was used to manage the project, draw bounding boxes, and export the data in YOLOv5 format.

In total, 78 labeled images were used:
• 69 images for training
• 9 images for testing
Roboflow handled the conversion to YOLOv5 format, generating the images, labels, and a data.yaml file specifying the dataset paths and class names.

The Roboflow SDK was used to programmatically download the dataset as follows:

```
from roboflow import Roboflow
rf = Roboflow(api_key="<API_KEY>")
project = rf.workspace("marioworkspacetest").project("swe_hw4-ni49x")
version = project.version(2)
```

```
dataset = version.download("yolov5")
```

This code connects to the Roboflow workspace and project, selects version 2 of the dataset, and downloads it directly into the YOLOv5 directory in the correct structure.

3. MODEL AND TRAINING CONFIGURATION

The YOLOv5s model (the small, fast variant of YOLOv5) was fine-tuned on the red panda dataset using transfer learning from the pretrained COCO weights. Training was launched with the following command:

```
!python train.py --img 512 \
                 --batch 8 \
                 --epochs 20 \
                 --data /content/yolov5/SWE_HW4-2/data.yaml \
                 --weights yolov5s.pt \
                 --name redpanda_yolov5s
```

Key hyperparameters:
• Image size: 512×512 pixels
• Batch size: 8
• Epochs: 20
• Base weights: yolov5s.pt (pretrained on COCO)
• Dataset configuration: SWE_HW4-2/data.yaml
The training script reads the custom dataset, performs on-the-fly augmentation, and updates the YOLOv5s weights to specialize on the red panda class.

4. INFERENCE ON THE TEST SET

After training, the best model checkpoint (based on validation metrics) was used for inference on the 9-image test set. Detection was run with:

```
!python detect.py \
  --weights runs/train/redpanda_yolov5s/weights/best.pt \
  --img 512 \
  --source /content/yolov5/SWE_HW4-2/test/images \
  --data /content/yolov5/SWE_HW4-2/data.yaml
```

The detect.py script loads the trained model, runs inference on each test image, and saves annotated copies of the images (with bounding boxes and confidence scores) into the runs/detect/ directory.

5. QUALITATIVE RESULTS

The following figures illustrate example detections produced by the trained model on unseen test images. Confidence scores range roughly from 0.29 to 0.50. In most cases, the model correctly localizes red pandas and assigns reasonable confidence scores.

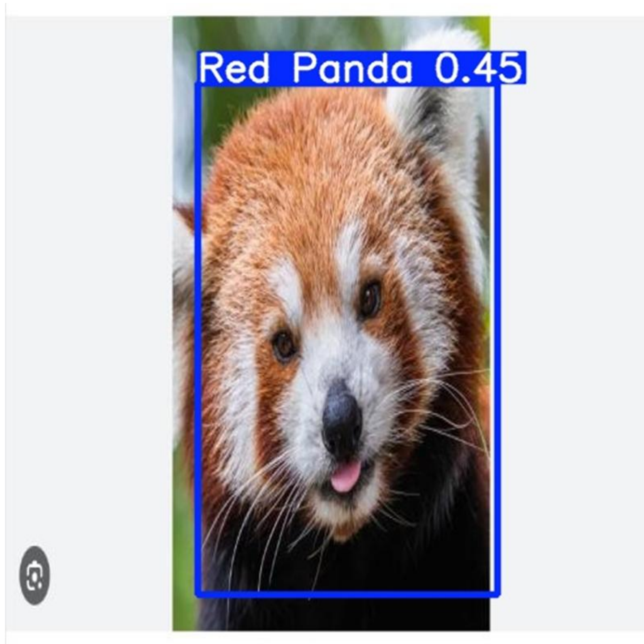

Figure 1. Single red panda detected with confidence ≈ 0.45.

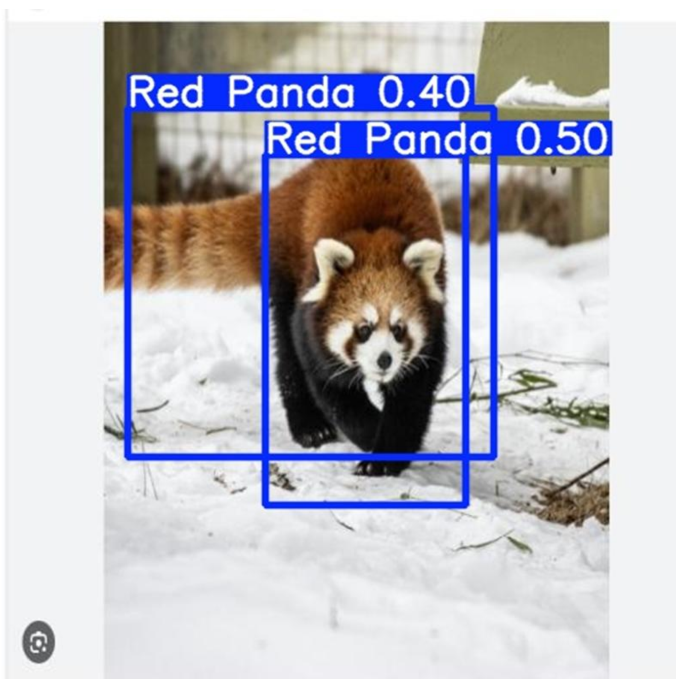

Figure 2. Red panda walking in snow with two overlapping detections (0.40 and 0.50).

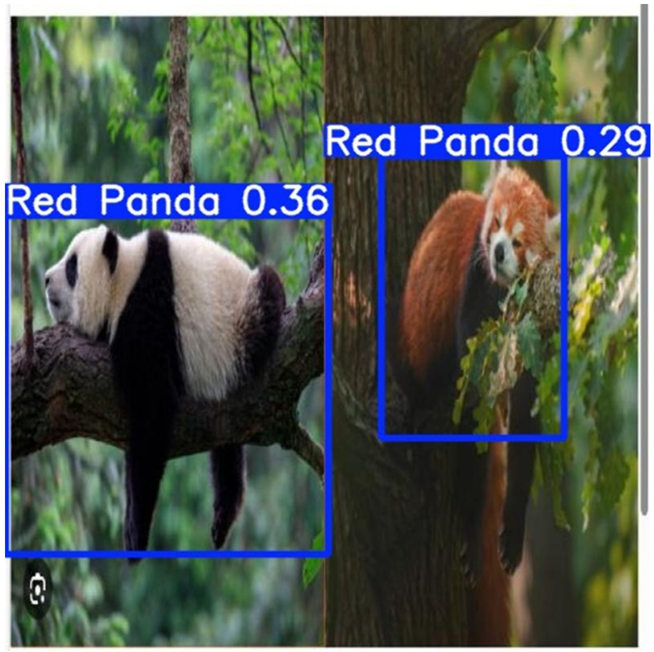

Figure 3. Mixed image with a giant panda and a red panda; the model predicts both as 'Red Panda', revealing some over-generalization.

Figure 4. Young red panda detected in an indoor environment (confidence ≈ 0.42) while ignoring nearby plush toys.

Overall, the model displays good localization ability on the small dataset. However, the prediction on the giant panda in Figure 3 shows that the model can produce false positives on visually similar animals due to the single-class training and limited data diversity.

6. DISCUSSION AND FUTURE WORK

Given the small dataset size (69 training and 9 testing images), the YOLOv5s model achieves promising qualitative performance. The bounding boxes are generally accurate, and the confidence scores are consistent across different poses and backgrounds. Nevertheless, there is room for improvement:

- Expanding the dataset with more red panda images and more varied backgrounds.
- Adding negative examples (other animals, especially giant pandas) to reduce false positives.
- Increasing the number of training epochs and tuning hyperparameters.
- Experimenting with larger YOLOv5 variants or alternative architectures.

Despite these limitations, this experiment successfully demonstrates a complete custom-object-detection pipeline: data collection and labeling with Roboflow, training a YOLOv5 model, and evaluating qualitative results on a held-out test set.