Enhanced DETR for Yellow Leaf Disease Detection

Xiaoning Li, Yaochen Song

Chalmers University of Technology
Electrical Engineering Department

CHALMERS UNIVERSITY OF TECHNOLOGY

Introduction to DETR

DETR mixes CNNs and Transformers to detect objects, seeing it as a straight-up guessing game without needing any area hints or past info like anchors. It does the whole object spotting in one go, [4]making the spotting process much simpler, and has four main parts to it.

- **Backbone Networ)**: Extracts image features via CNN.
- **Transformer Network**:
  - Encoder: Processes features from the backbone network.
  - Decoder: Predicts object bounding boxes and classes.
- **Feed Forward Network (FFN)**: Provides final predictions.

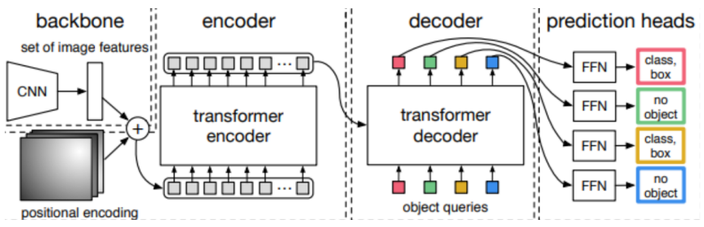

Figure 1: The architecture diagram of DETR.

The image below showcases an example processed by DETR. It annotates the objects with bounding boxes, labels, and confidence scores.

Figure 2: An example processed by DETR.

Theoretical background

- **Comparasion of three models**:
  - DETR(ResNet-50): CNN architecture known for its performance in image classification tasks, featuring 50 layers with skip connections.
  - DETR(MobileNet): A lightweight deep learning model designed for efficient on-device image and vision processing
  - DETR(EfficiencyNet): A family of neural network architectures that systematically scales

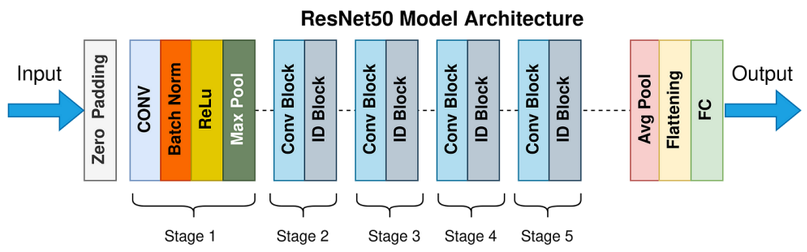

Figure 3: DETR(ResNet-50)[1]

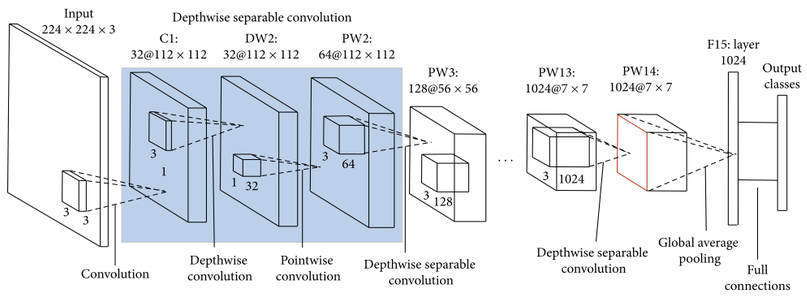

Figure 4: DETR(MobileNet)[2]

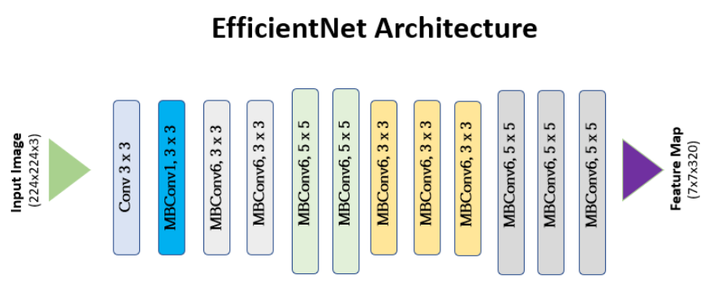

Figure 5: DETR(EfficiencyNet).[3]

Dataset

This project analyzes small data sets. We use the Yellow Leaf Disease Dataset :COCO standard data set format; 378 train images; 98 val images; 28 test images.

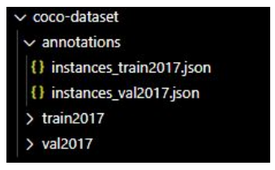

Figure 6: Dataset structure

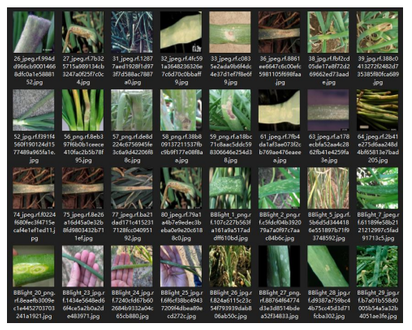

Figure 7: Part of the images in the training set

Results and Conclusions

In this project we use Precision/Recall; Score/Recall; AP across scales:large, medium, small; AP; APIoU50%; APIoU75% to evaluate the backbone.

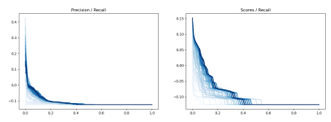

Figure 8: Image 1

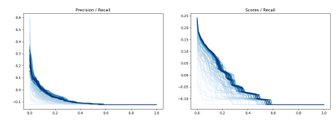

Figure 9: Image 2

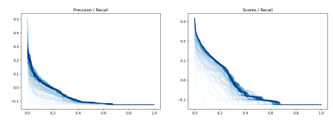

Figure 10: Image 2

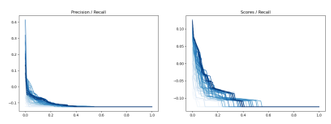

Figure 11: Image 1

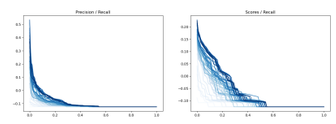

Figure 12: Image 2

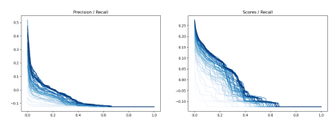

Figure 13: Image 2

Figure 14: Image 1

Figure 15: Image 2

epoch	ResNet-50	MobileNet	EfficientNet
50epochs	48min	38min	59min
100epochs	1h4min	1h17min	2h16min

Table 1: Training time based on GTX2080

In general, Mobilenet runs faster and has lower deployment costs. Resnet50 has better performance in terms of accuracy.
In addition, in the image of the recall rate, we found that for this data set, the difference between different backbones when training for 50 epoch is greater than that for 100 epoch.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun.
Deep residual learning for image recognition.

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam.
Mobilenets: Efficient convolutional neural networks for mobile vision applications.

[3] M. Tan and Q. V. Le.
Efficientnet: Rethinking model scaling for convolutional neural networks.

[4] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai.
Deformable detr: Deformable transformers for end-to-end object detection.

xiaoning@student.chalmers.se; yaochen@student.chalmers.se