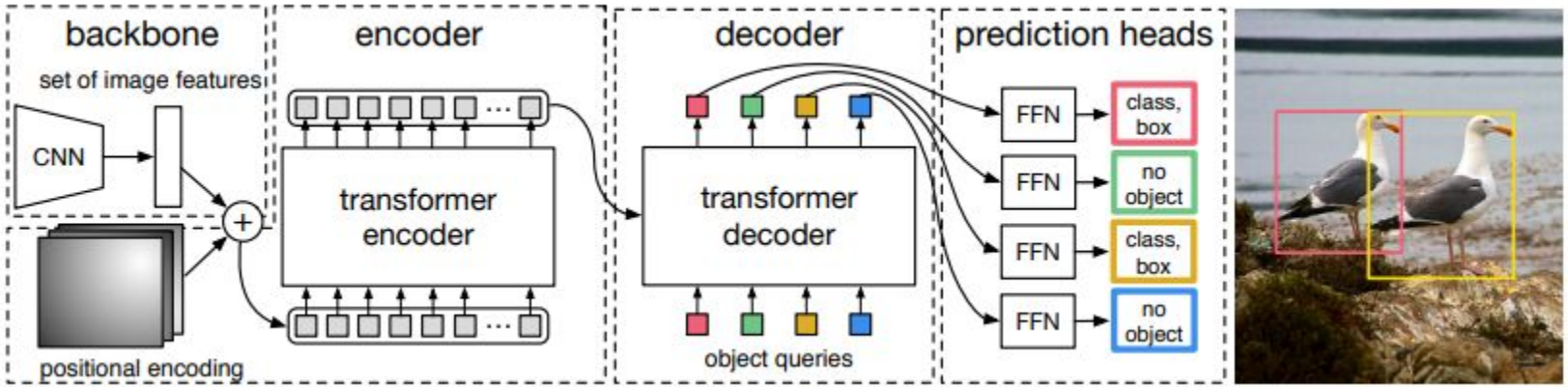
Object Detection with DETR

Group 14:
Tuan Nguyen
Nikhil Darwin Bollepalli
Erum Hooda

DETR Architecture

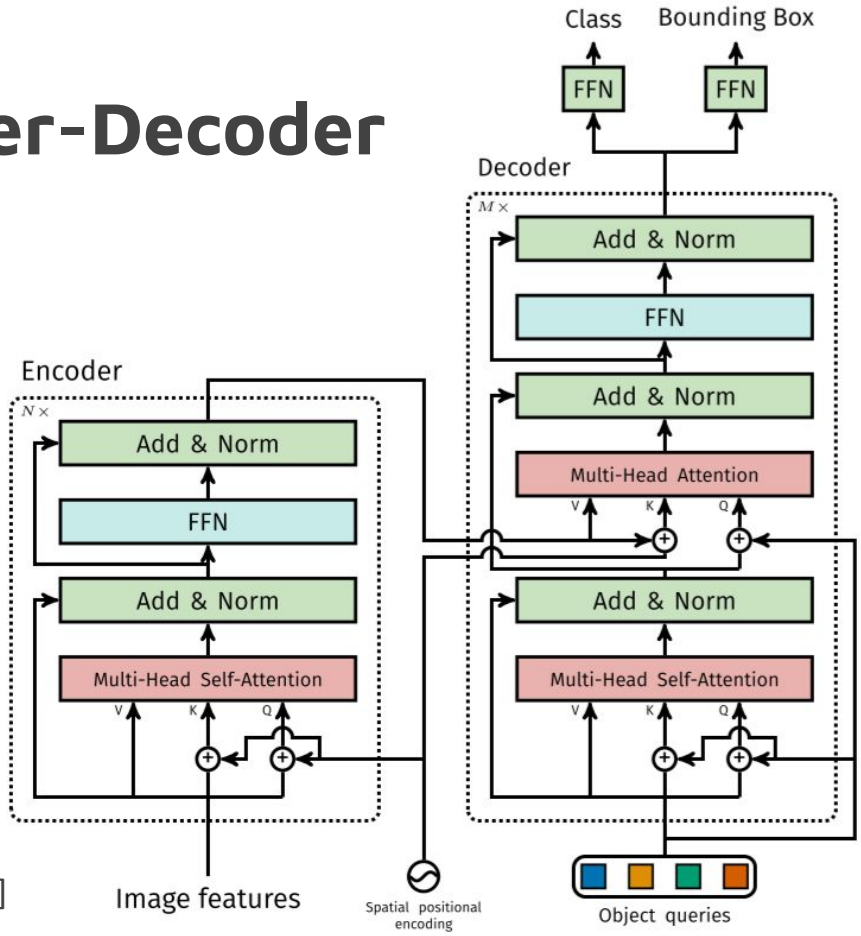
- DETR: Object **DE**tection with **TR**ansformers
- Goal:
  - Evaluate our own implementation of DETR against original DETR implementation by Facebook



[1]

Transformer Encoder-Decoder

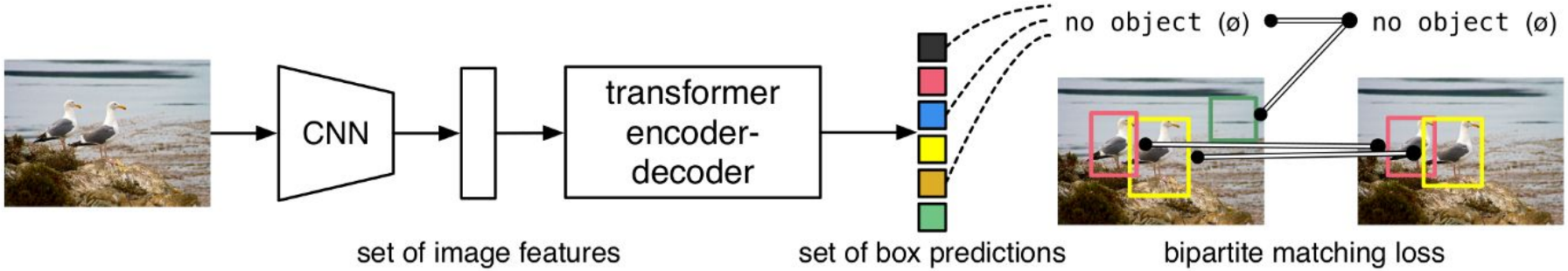
- Encoder
  - Input = feature map + positional encodings
  - Has multi-head self-attention module and feed forward network
  - Encodes image features
- Decoder
  - Input = encoder output + queries
  - N queries learned in training
  - Each query results in bounding box + class label
  - Some queries map to no object
  - Output fed into feed forward network



[3]

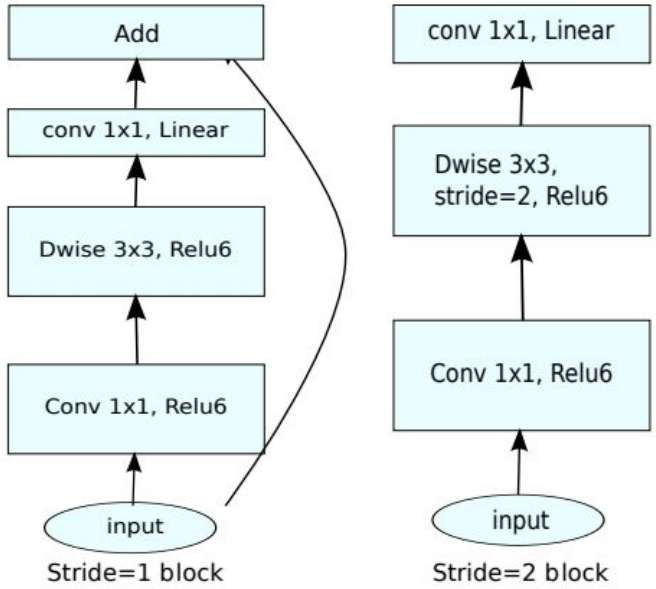
Loss Function

- Bipartite Matching
  - Assign each predicted bounding box + class label to a ground truth bounding box + class label
  - 1:1 matching
- Hungarian Algorithm
  - Finds optimal bipartite matching
  - Minimize total loss



[5]

Backbone: MobileNetV2

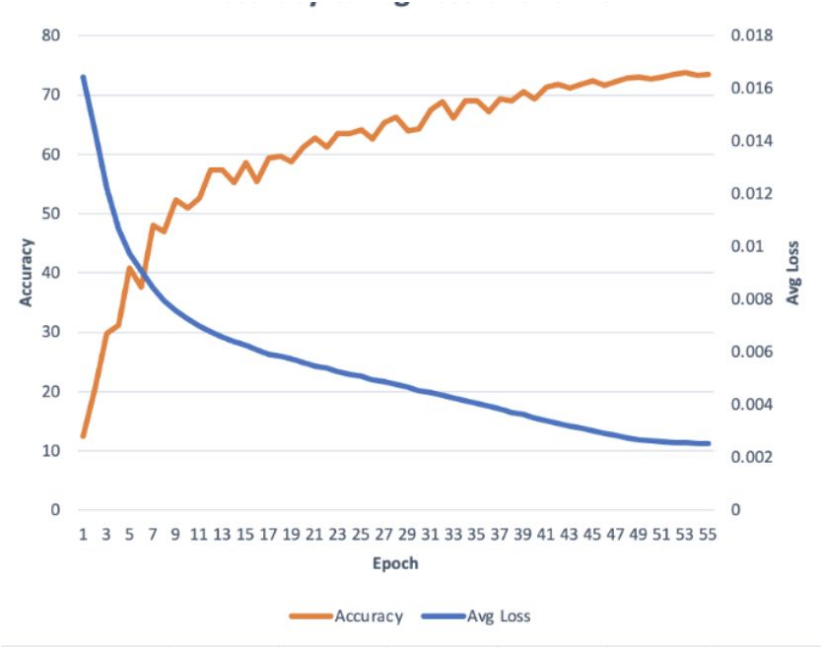


(d) Mobilenet V2

[2]

- Backbone:
  - pretrained CNN
  - Outputs feature representation of input image
- MobileNetV2 chosen over ResNet50 because it is smaller, which means:
  - shorter running time
  - Less memory used
- Outputted feature map fed into Encoder-Decoder

Loss vs Accuracy Curve: MobileNetV2



Training time: 8+ hours

Top-1 accuracy : 75%

Epochs trained: 200

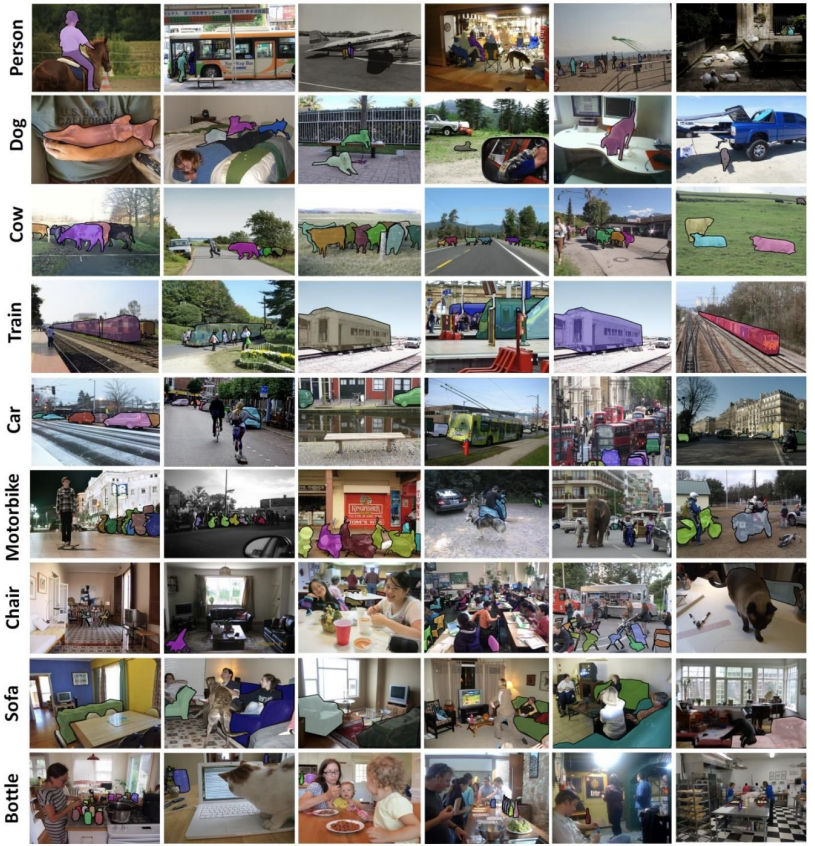
Batch size: 128

Dataset: ImageNet Subset

Dataset

COCO dataset

- 91 classes, including “N/A”
- 328K images



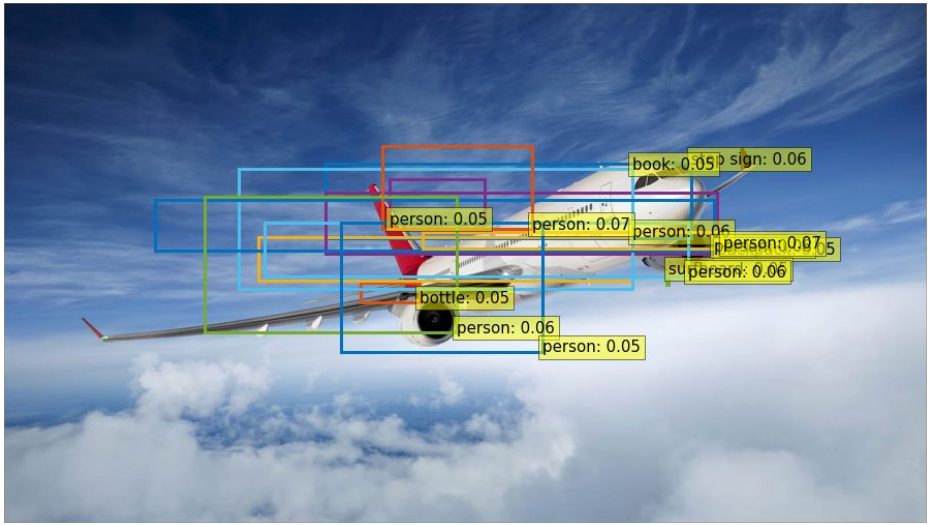
[4]

Training

- Training DETR is extremely resource intensive even using a smaller backbone
  - 41M parameters VS 16M parameters
- Impractical to train on consumer hardware
  - Original paper trained on 16 V100 GPUs
  - We had 1 P100 (Kaggle)
  - 1 epoch of full transformer took 10 hours on Kaggle
  - 1 epoch of the scaled down transformer took 4 hours

Hyperparameters								
	Queries	Hidden	Heads	Encoder	Decoder	Feedforward	Learning Rate	Batch Size
Original	100	512	8	6	6	2048	1.00E-04	64
1st Attempt	100	512	8	6	6	2048	0.1	1
2nd Attempt	50	512	1	1	1	1024	0.1	1

Results



- Ability to train was limited
  - Kaggle kept timing out
  - 1 epoch on full transformer
  - 5 epochs on scaled down transformer
- Transformer proved to be a much bigger bottleneck compared to the backbone

PyTorch

Questions?

References

[1] https://www.analyticsvidhya.com/blog/2020/05/facebook-detection-transformer-detr-a-transformer-based-object-detection-approach/

[2] https://paperswithcode.com/method/mobilenetv2

[3] https://medium.com/swlh/object-detection-with-transformers-437217a3d62e

[4] https://paperswithcode.com/dataset/coco

[5] https://medium.com/analytics-vidhya/up-detr-unsupervised-pre-training-for-object-detection-with-transformers-paper-explained-84611e27a144

[6] https://arxiv.org/pdf/1709.01507.pdf