Applied AGI (Interview)

Presented By: Gur Amrit Singh

Presentation Outline

Problem Statement	Dataset	CNN Architectures	Results
Bottle Necks	Multi-Viewpoint Object Detection	Data Loader	Previous Work

Problem Statement

- Classify, recognise, and localise target vehicle from the sky against varying back grounds.

- Train off-the shelf CNN models using custom Dataloader class, which can load images infinitely.

- Classify target vehicle from multiple viewpoints.

- Provide a count and accuracies of vehicles classified.

Dataset

- Publicly available Satellite Imagery Multi-Vehicles Dataset (SIMD) [1].
- Consists of 5000 images (1024x768).
- 15 classes
- 45096 total objects.
- Train – 4000, Val/Test – 1000

0 – Car | 2 – Van | 3 – Long Vehicle | 14 - Boat

[1] Haq, Nazeef Ul, et al. "Orientation Aware Weapons Detection In Visual Data: A Benchmark Dataset." *arXiv preprint arXiv:2112.02221* (2021).

CNN Architectures

Faster RCNN

Yolo v3

Retina Net

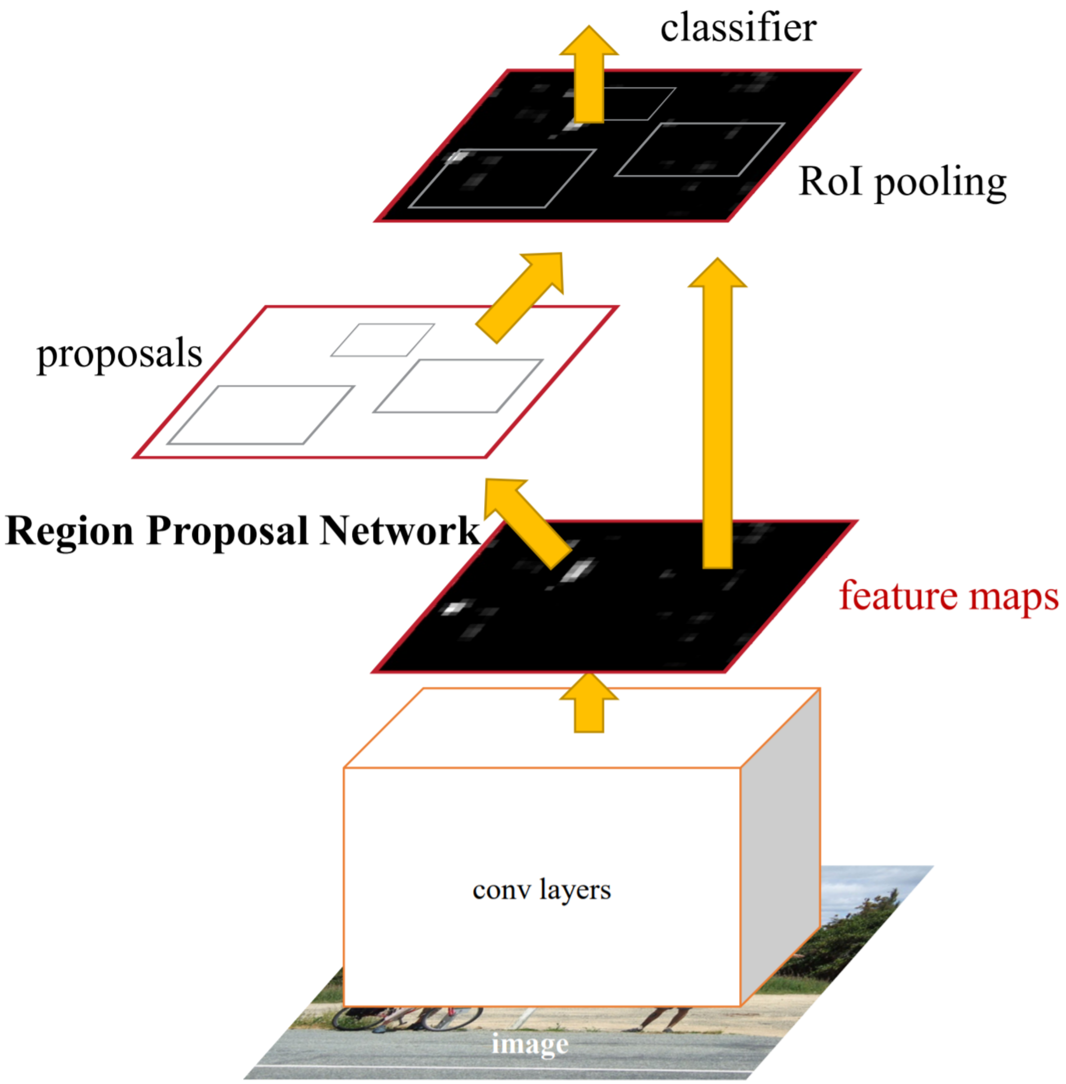
Center Net

DETR

Faster - RCNN

Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks

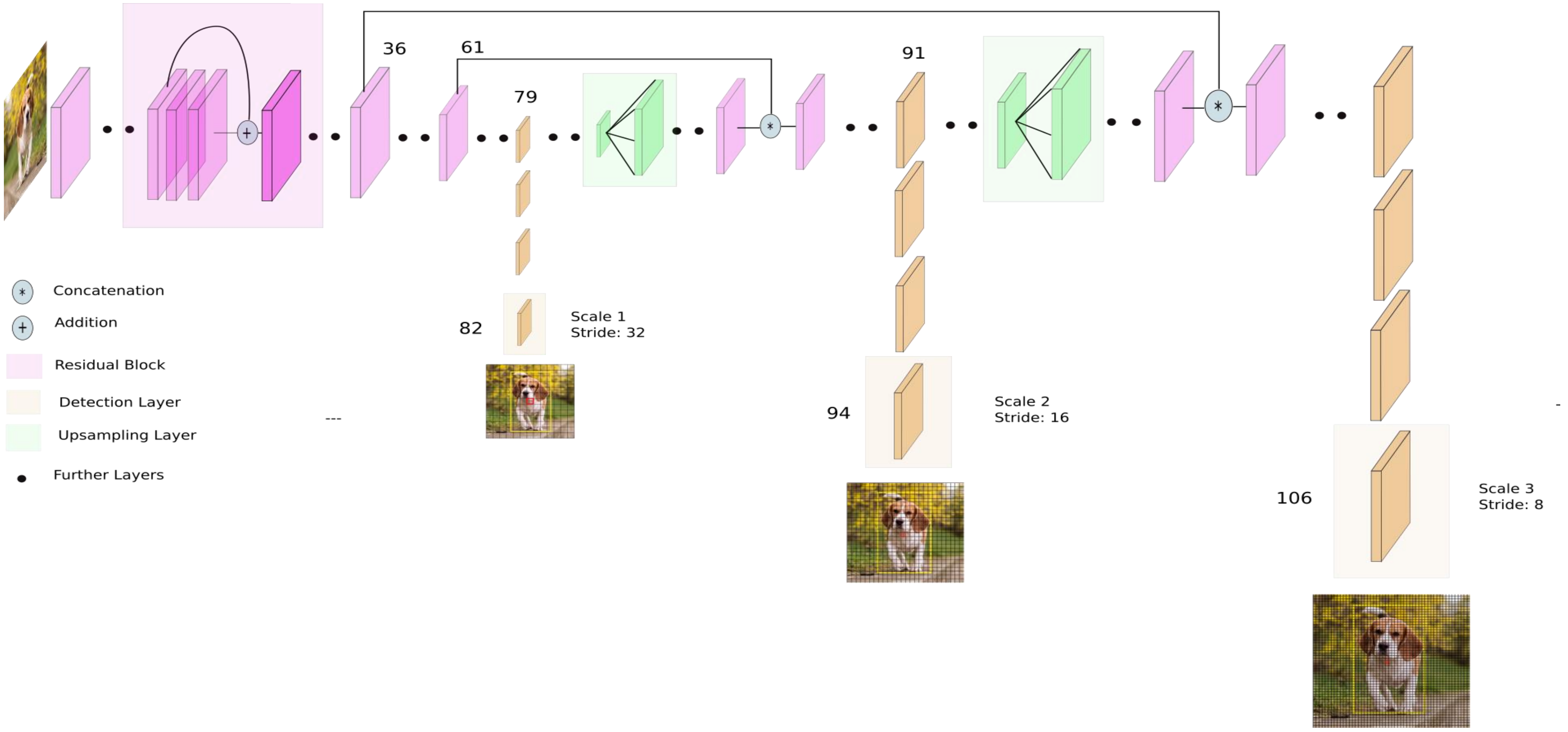
Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun



[2] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems* 28 (2015): 91-99.

YOLOv3: An Incremental Improvement

Joseph Redmon Ali Farhadi
University of Washington

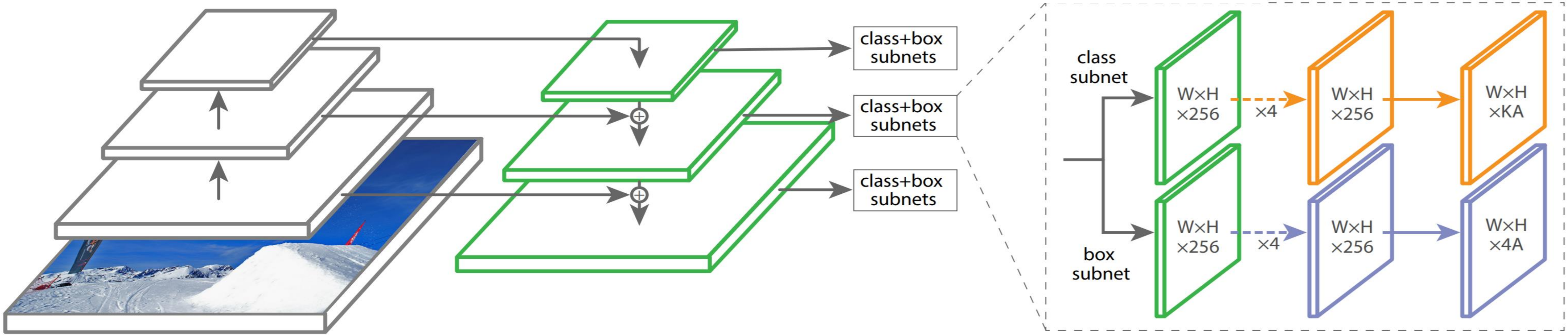


[3] Kathuria, A., 2022. *What's new in YOLO v3?*. [online] Medium. Available at: <https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b#:~:text=First%2C%20YOLO%20v3%20uses%20a,v3%20compared%20to%20YOLO%20v2.> [Accessed 19 January 2022].

Retina Net

Focal Loss for Dense Object Detection

Tsung-Yi Lin   Priya Goyal   Ross Girshick   Kaiming He   Piotr Dollár
Facebook AI Research (FAIR)



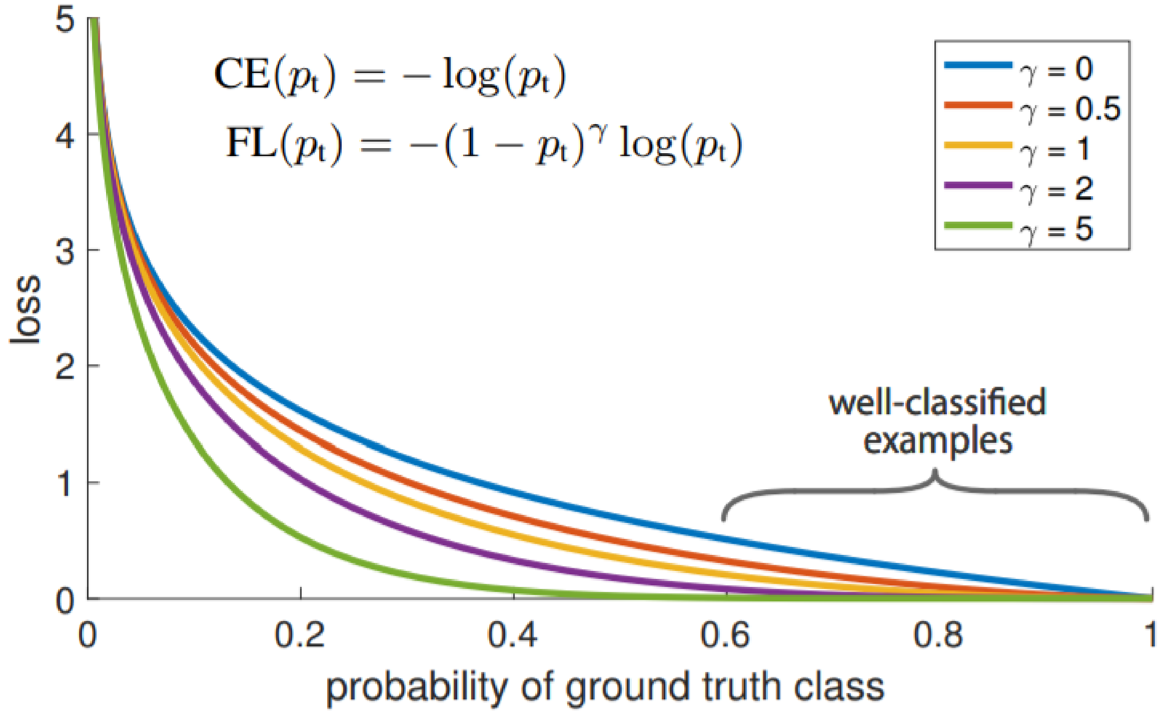
(a) ResNet   (b) feature pyramid net   (c) class subnet (top)   (d) box subnet (bottom)

[4] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.

Retina Net

Focal Loss for Dense Object Detection

Tsung-Yi Lin Priya Goyal Ross Girshick Kaiming He Piotr Dollár
Facebook AI Research (FAIR)



[4] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.

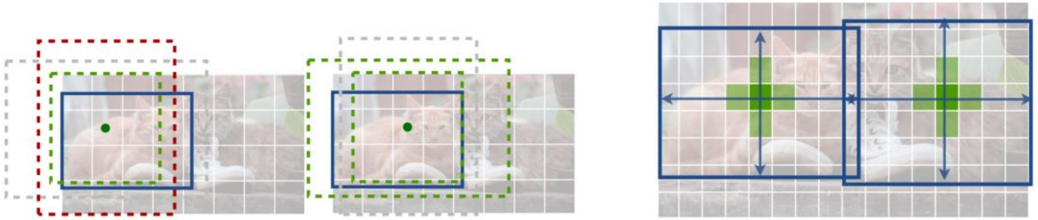
Center Net

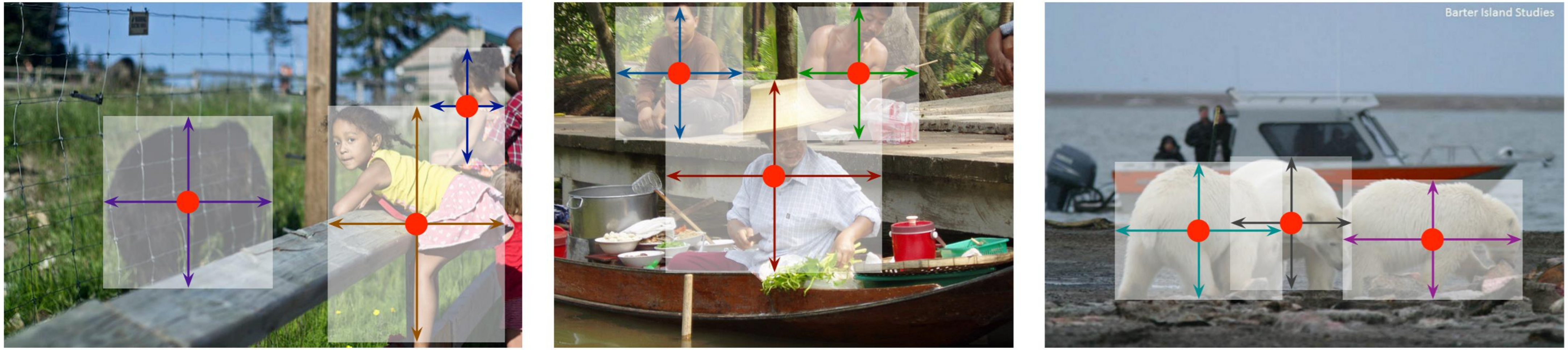
Objects as Points

Xingyi Zhou
UT Austin
zhouxy@cs.utexas.edu

Dequan Wang
UC Berkeley
dqwang@cs.berkeley.edu

Philipp Krähenbühl
UT Austin
philkr@cs.utexas.edu



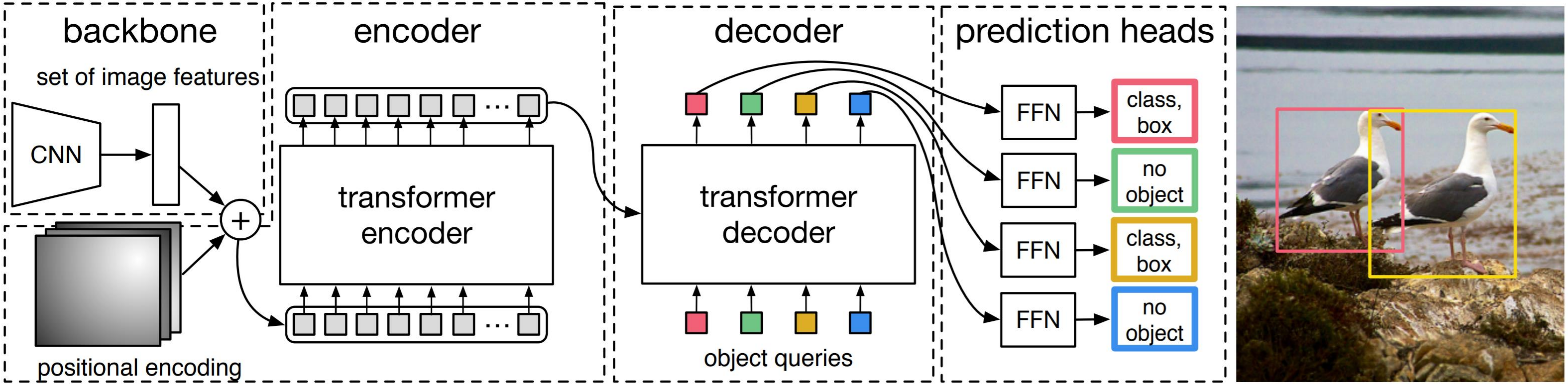


[5] Zhou, Xingyi, Dequan Wang, and Philipp Krähenbühl. "Objects as points." *arXiv preprint arXiv:1904.07850* (2019).

DETR

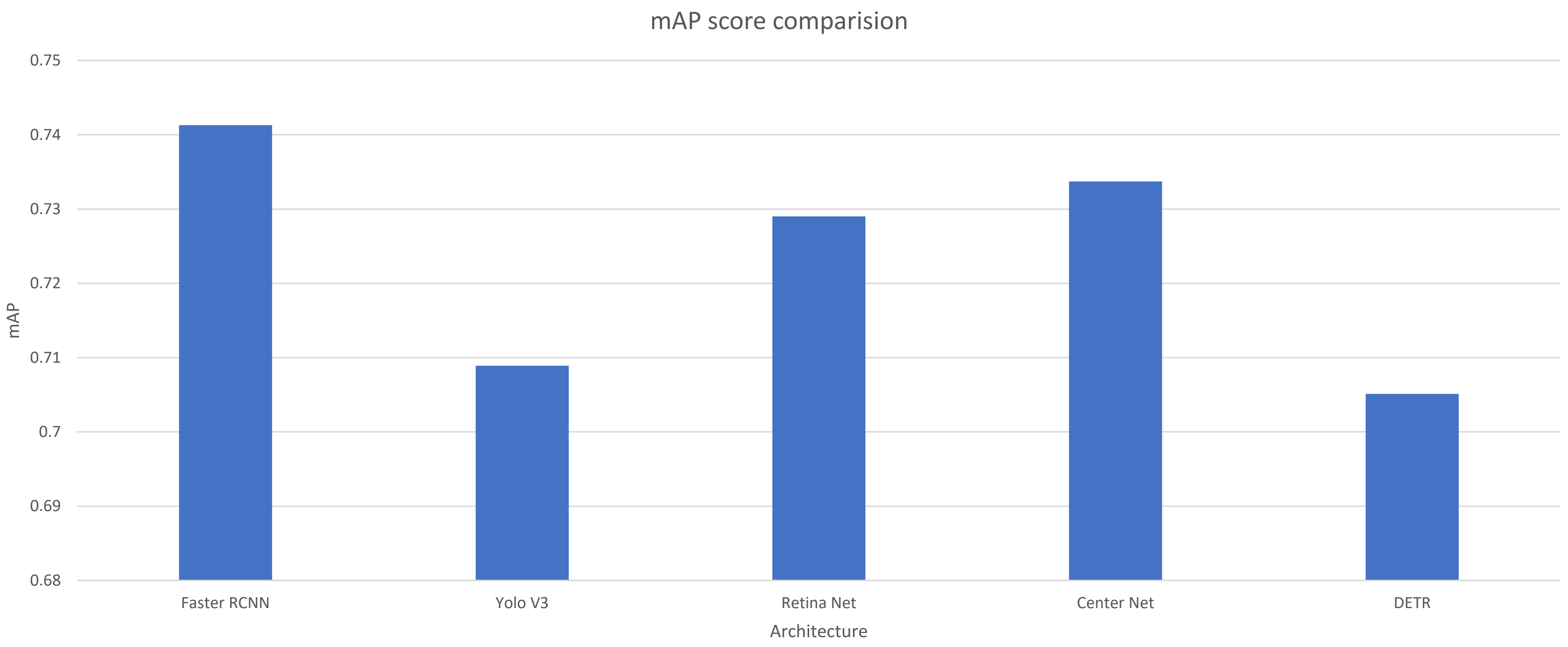
End-to-End Object Detection with Transformers

Nicolas Carion*, Francisco Massa*, Gabriel Synnaeve, Nicolas Usunier,
Alexander Kirillov, and Sergey Zagoruyko



[6] Carion, Nicolas, et al. "End-to-end object detection with transformers." European Conference on Computer Vision. Springer, Cham, 2020.

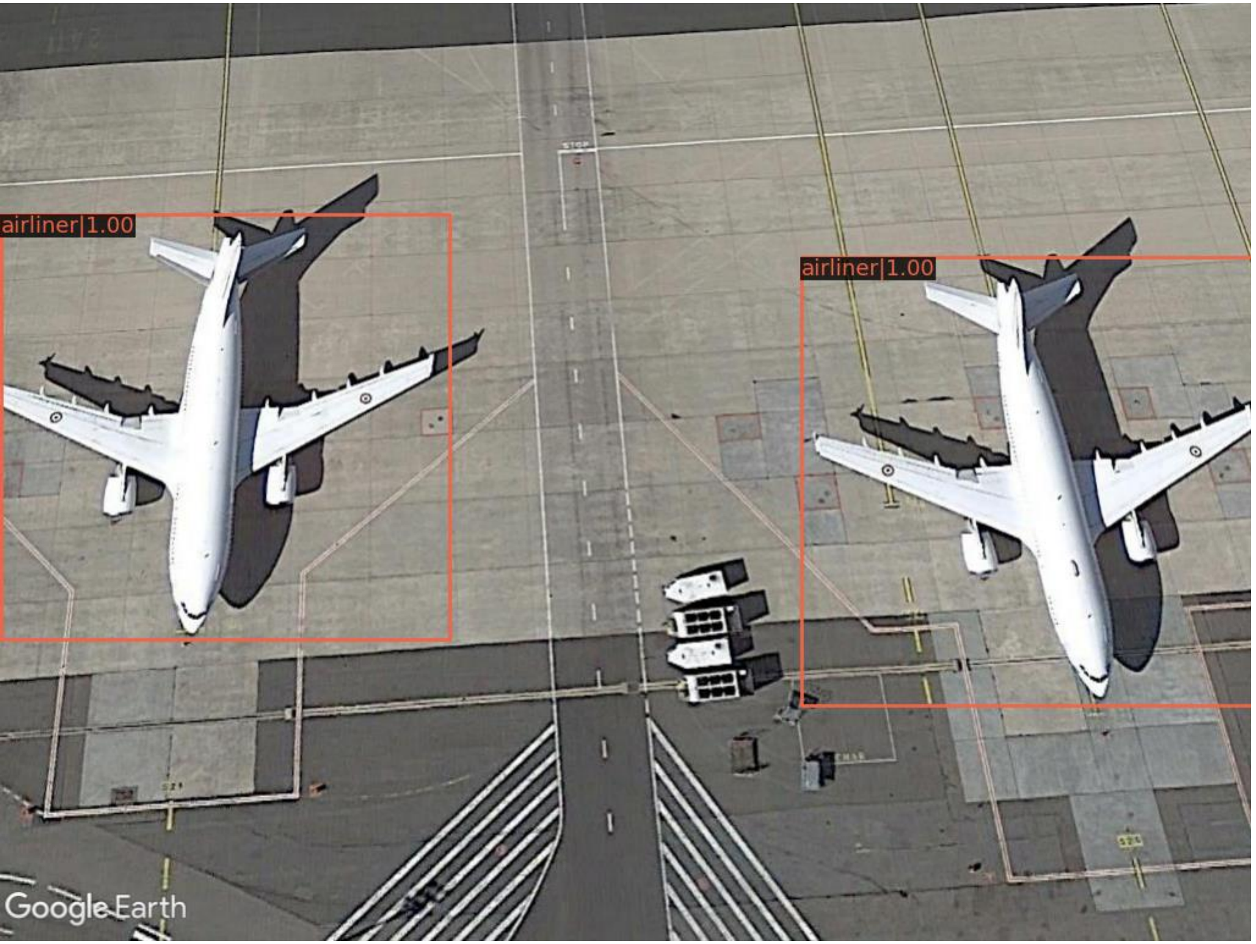
Results



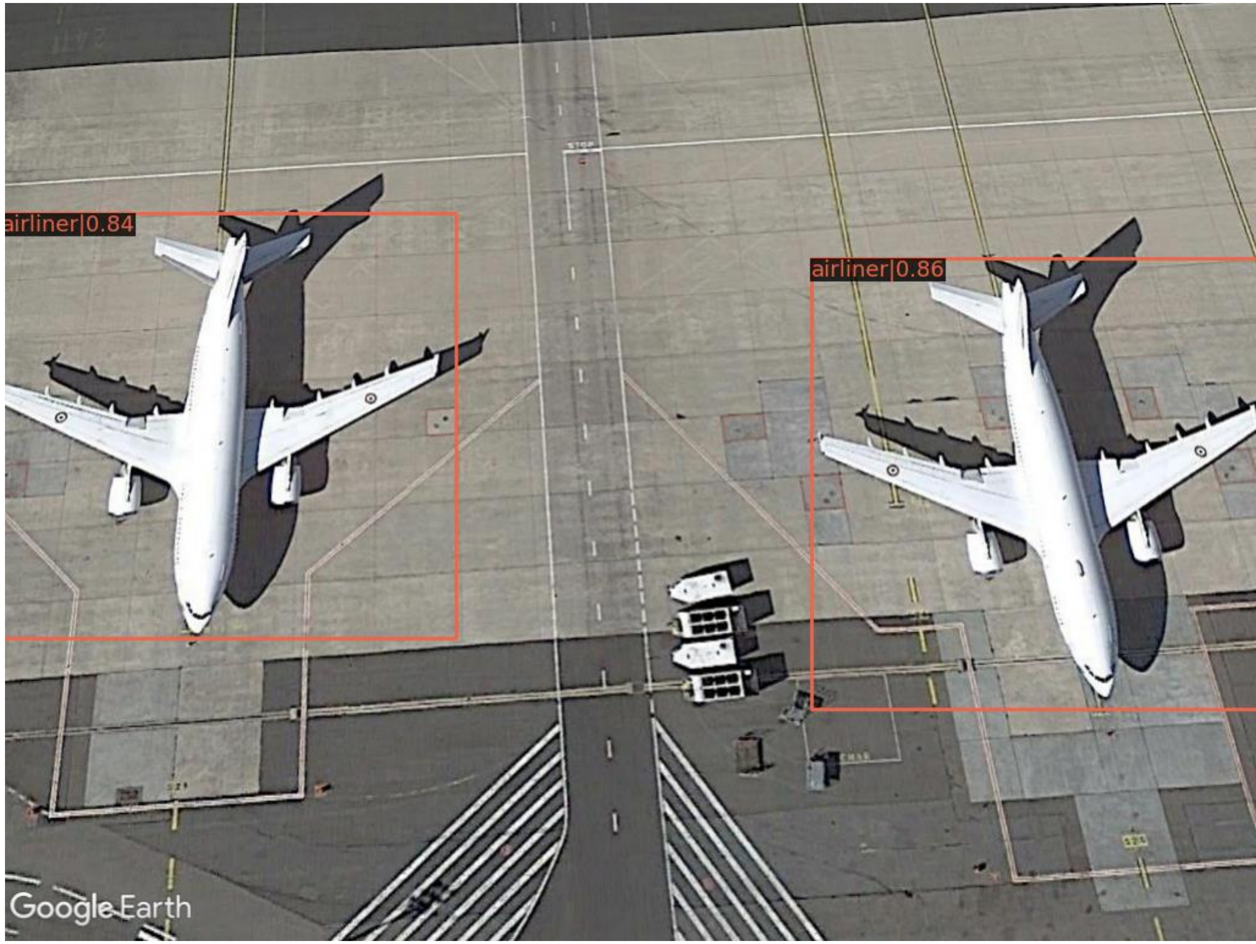
mAP score comparision
0.75
0.74
0.73
0.72
0.71
0.7
0.69
0.68
mAP
Faster RCNN
Yolo V3
Retina Net
Center Net
DETR
Architecture

Results

Faster RCNN



Center Net



Bottle Necks

- Slow inference time for two-stage object detectors.
- Too many anchor boxes.
- Easy to miss oddly shaped objects, and partially occluded objects.
- Instance mask generation is difficult for Center Net
- Quadratic attention mechanism for DETR.

Multi-View Object Detection.

- Same image taken from different viewpoints.
- Model performing sub optimally for different viewpoint images than what it was trained on.
- As the vehicle moves, the position of the drone changes as well, changing the viewpoint.
- Transformer based CNN could be used to solve this.

Dataset and Data Loader

- Created a custom dataset class to pre-process and load the dataset.
- Transforms images using different augmentations, and stores images and bounding boxes as tensors.
- Custom dataloader class extracts batches indefinitely from the dataset. (OpenMMLab Computer Vision Foundation – used as back bone)

```
infinite_sampler = InfiniteBatchSampler(len(dataset), shuffle=shuffle)
DataLoader( dataset, batch_size=batch_size, sampler=infinite_sampler, num_workers=num_workers, pin_memory=True,
                collate_fn=partial(collate, samples_per_gpu=batch_size))
```

GitHub

Data set

GitHub

Data Loader

Data Loader

```python
class InfiniteBatchSampler(Sampler):
    def __init__(self,
                 dataset_size,
                 world_size=1,
                 rank=0,
                 seed=42,
                 shuffle=True):
        assert dataset_size > 0
        self.rank = rank
        self.world_size = world_size
        self.seed = seed if seed is not None else 42
        self.shuffle = shuffle
        self.size = dataset_size

    def __iter__(self):
        start = self.rank
        yield from itertools.islice(self._infinite_indices(), start, None, self.world_size)

    def _infinite_indices(self):
        g = torch.Generator()
        g.manual_seed(self.seed)
        while True:
            if self.shuffle:
                yield from torch.randperm(self.size, generator=g)
            else:
                yield from torch.arange(self.size)
```

GitHub

Data set

GitHub

Data Loader

Data Loader

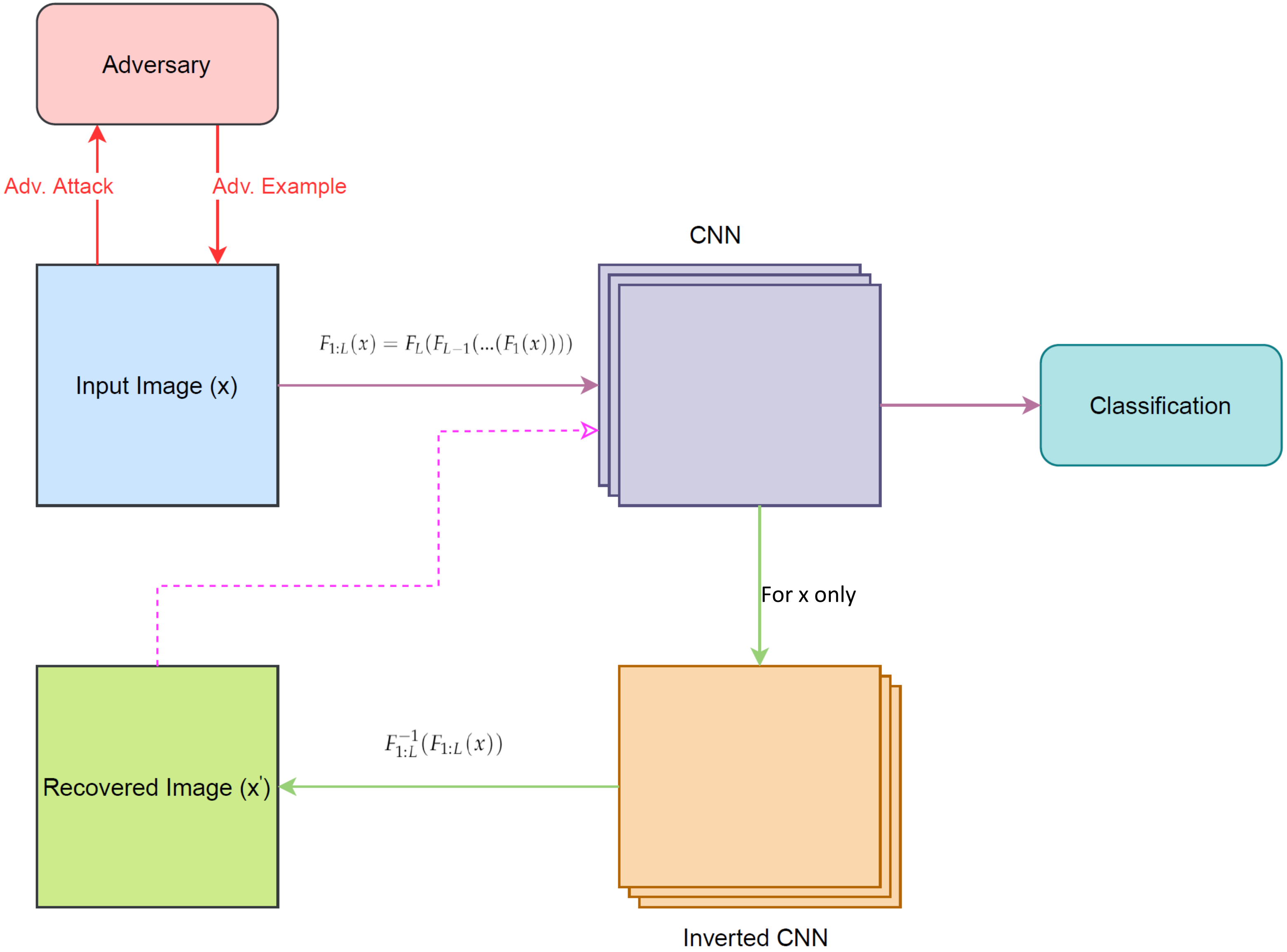
```python
def collate(batch, samples_per_gpu=1):
    stacked=[]
    for i in range(0, len(batch), samples_per_gpu):
        assert isinstance(batch[i]['img'], torch.Tensor)
        ndim = batch[i]['img'].dim()
        assert ndim > 2
        max_shape = [0 for _ in range(2)]
        for dim in range(1, 3):
            max_shape[dim - 1] = batch[i]['img'].size(-dim)
        for sample in batch[i:i + samples_per_gpu]:
            for dim in range(ndim - 2):
                assert batch[i]['img'].size(dim) == sample['img'].size(dim)
            for dim in range(1, 2+ 1):
                max_shape[dim - 1] = max(max_shape[dim - 1],
                                         sample['img'].size(-dim))
        padded_samples = []
        for sample in batch[i:i + samples_per_gpu]:
            pad = [0 for _ in range(4)]
            for dim in range(1, 3):
                pad[2 * dim -
                    1] = max_shape[dim - 1] - sample['img'].size(-dim)
            padded_samples.append(
                F.pad(
                    sample['img'], pad, value=0))
        stacked.append(default_collate(padded_samples))

        stacked.append(default_collate([sample['gt_bboxes'] for sample in batch[i:i + samples_per_gpu]]))
        stacked.append(default_collate([sample['gt_labels'] for sample in batch[i:i + samples_per_gpu]]))
    return stacked
```

Data set

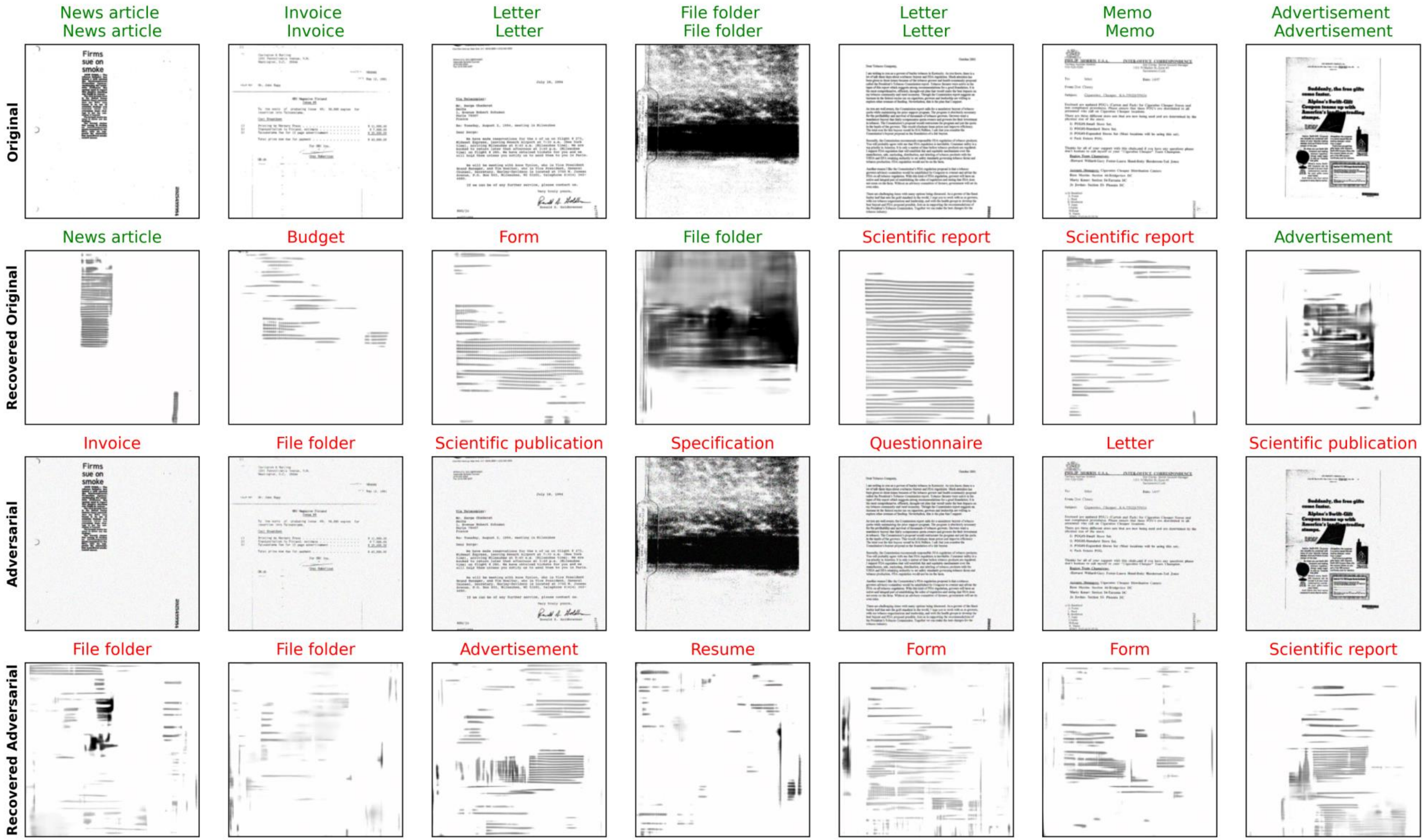
Data Loader

Prior Work

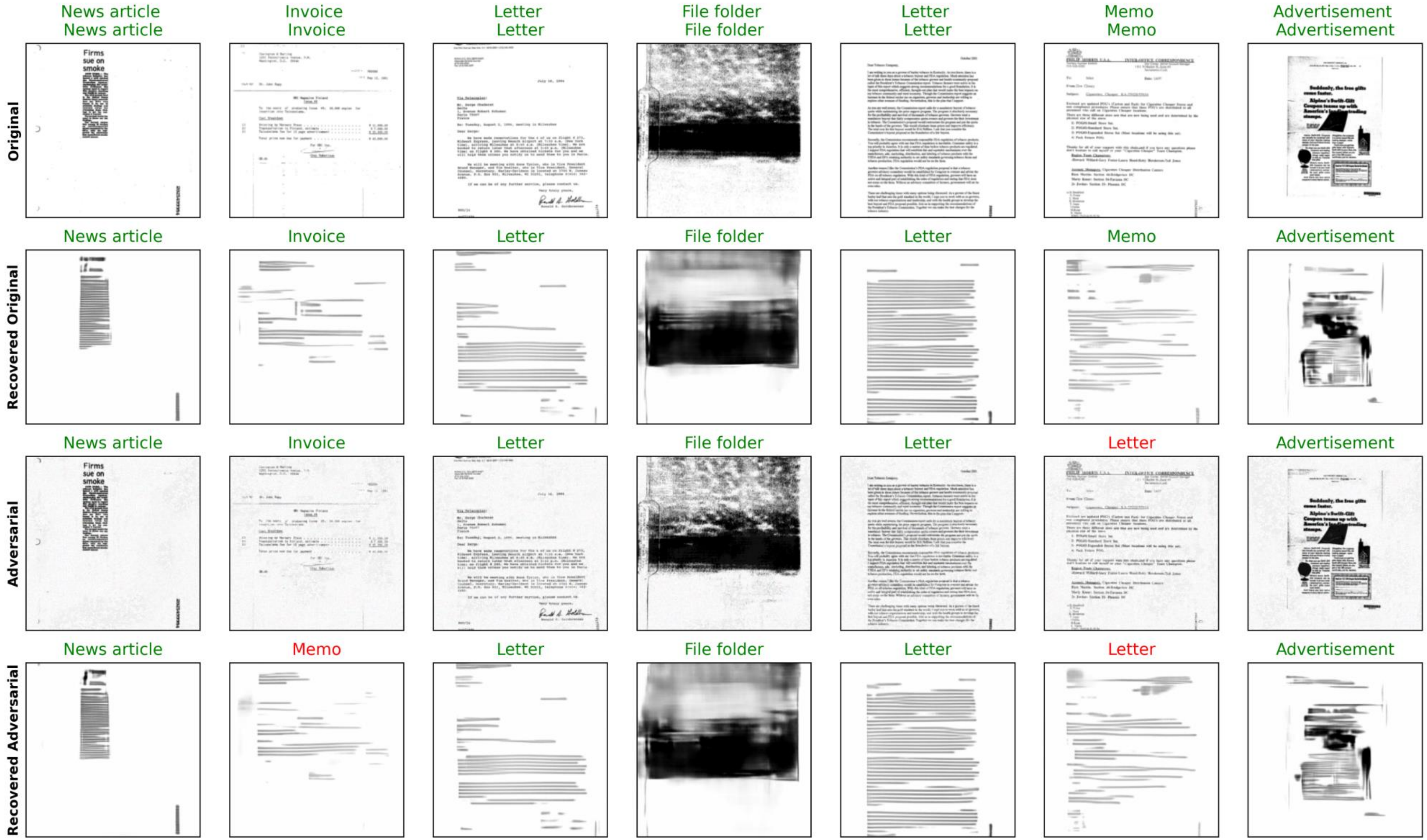


Adversary
Adv. Attack
Adv. Example
CNN
Input Image (x)
$F_{1:L}(x) = F_L(F_{L-1}(...(F_1(x))))$
Classification
For x only
Recovered Image (x')
$F_{1:L}^{-1}(F_{1:L}(x))$
Inverted CNN

Prior Work (Results - Non-Robust Model)



Prior Work (Results - Robust Model)



Thank You

References

[1] Haq, Nazeef Ul, et al. "Orientation Aware Weapons Detection In Visual Data: A Benchmark Dataset." *arXiv preprint arXiv:2112.02221* (2021).

[2] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems* 28 (2015): 91-99.

[3] Kathuria, A., 2022. *What’s new in YOLO v3?*. [online] Medium. Available at: <https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b#:~:text=First%2C%20YOLO%20v3%20uses%20a,v3%20compared%20to%20YOLO%20v2.> [Accessed 19 January 2022].

[4] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.

[5] Zhou, Xingyi, Dequan Wang, and Philipp Krähenbühl. "Objects as points." *arXiv preprint arXiv:1904.07850* (2019).

[6] Carion, Nicolas, et al. "End-to-end object detection with transformers." European Conference on Computer Vision. Springer, Cham, 2020.

Prior Work (Results - Robust Model)