COLLANTES
TRINIDAD

Midterm Project:

Implementing Object Detection on a Dataset

PERCEPTION AND COMPUTER VISION

1 SELECTED DATASET & OBJECT DETECTION ALGORITHM

Sample image from the dataset

Selected Algorithm:

- **YOLO (You Only Look Once)**
  is a fast object detection model that detects multiple objects in a single pass by dividing images into grids and predicting bounding boxes and class probabilities simultaneously.

- **SSD (Single Shot Multibox Detector)**
  is an efficient object detection model that predicts bounding boxes and class scores in a single pass, balancing speed and accuracy for real-time applications.

MAIN STEPS IN THE PROJECT

01 Data preparation
02 Model implementation
03 Training
04 Evaluation
05 Submission

2 DATA PREPARATION

Each image contains various objects, including playing cards, player chips, and game information boxes, which are essential for detecting game states and outcomes. This data could be useful for building a poker analytics tool or AI that can analyze and provide insights during poker games.

Source: https://universe.roboflow.com/card-fazsf/c-f6bfq/dataset/14/images?split=train

3 Model Implementation

WHY YOLO?

- Real-time object detection
- High accuracy
- Single-shot detection
- Good performance on small objects
- Efficient use of GPUs
- Ability to handle multiple scales

WHY SSD?

- Real-Time Performance
- Single-Stage Detection
- Multi-Scale Feature Maps
- High Accuracy with Speed Trade-Off
- Flexibility and Scalability
- Efficient for Mobile Applications

3 Model Implementation

YOLO (You Only Look Once)

```python
from ultralytics import YOLO

# Load a COCO-pretrained YOLOv8n model
model = YOLO("yolov8n.pt")

# Display model information (optional)
model.info()
```

```
Creating new Ultralytics Settings v0.0.6 file ✅
View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralytics/settings.json'
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see https://docs.ultralytics.com/quickstart/#ultralytics-settings.
Downloading https://github.com/ultralytics/assets/releases/download/v8.3.0/yolov8n.pt to 'yolov8n.pt'...
100%|██████████| 6.25M/6.25M [00:00<00:00, 123MB/s]
YOLOv8n summary: 225 layers, 3,157,200 parameters, 0 gradients, 8.9 GFLOPs
(225, 3157200, 0, 8.8575488)
```

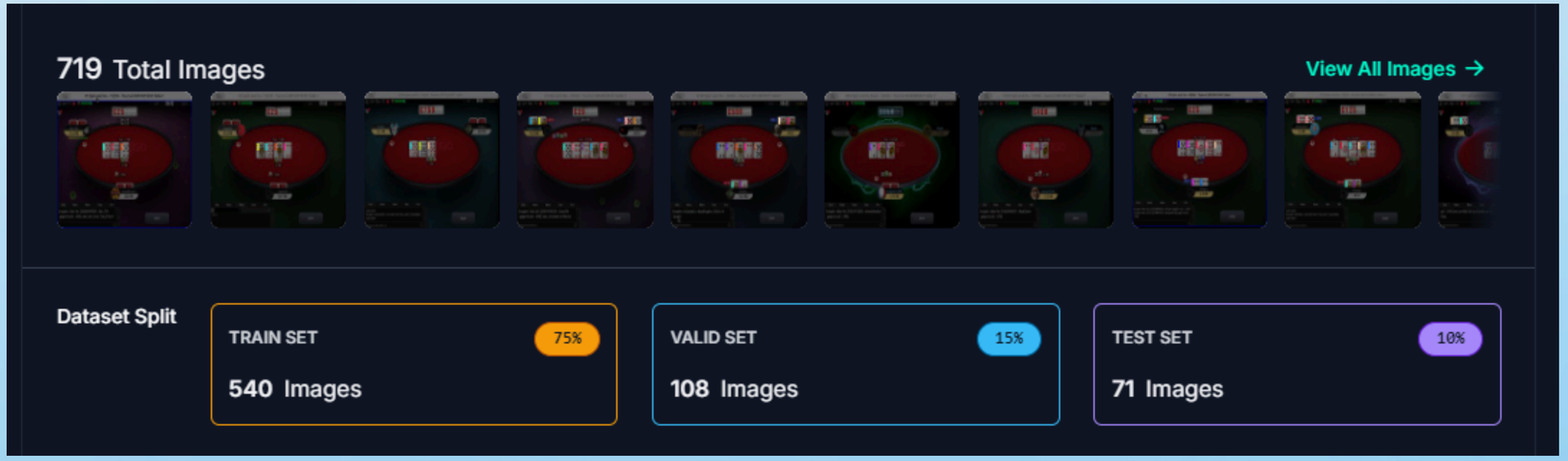
3 Model Implementation

SSD (Single Shot Multibox Detector)

```
# Create "mymodel" folder for holding pre-trained weights and configuration files
%mkdir /content/models/mymodel/
%cd /content/models/mymodel/

# Download pre-trained model weights
import tarfile
download_tar = 'http://download.tensorflow.org/models/object_detection/tf2/20200711/' + pretrained_checkpoint
!wget {download_tar}
tar = tarfile.open(pretrained_checkpoint)
tar.extractall()
tar.close()

# Download training configuration file for model
download_config = 'https://raw.githubusercontent.com/tensorflow/models/master/research/object_detection/configs/tf2/' + base_pipeline_file
!wget {download_config}
```

```
/content/models/mymodel
--2024-10-28 01:12:17--  http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz
Resolving download.tensorflow.org (download.tensorflow.org)... 74.125.137.207, 142.250.101.207, 142.251.2.207, ...
Connecting to download.tensorflow.org (download.tensorflow.org)|74.125.137.207|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 20515344 (20M) [application/x-tar]
Saving to: ‘ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz’

ssd_mobilenet_v2_fp 100%[===================>]  19.56M  --.-KB/s    in 0.1s

2024-10-28 01:12:17 (178 MB/s) - ‘ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz’ saved [20515344/20515344]

--2024-10-28 01:12:18--  https://raw.githubusercontent.com/tensorflow/models/master/research/object_detection/configs/tf2/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.config
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4684 (4.6K) [text/plain]
Saving to: ‘ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.config’

ssd_mobilenet_v2_fp 100%[===================>]   4.57K  --.-KB/s    in 0s

2024-10-28 01:12:18 (70.3 MB/s) - ‘ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.config’ saved [4684/4684]
```

4 Model Training



Source: https://universe.roboflow.com/card-fazsf/c-f6bfq/dataset/14

4 Model Training

YOLO (You Only Look Once)

```
[ ] results = model.train(data="/content/data.yaml", epochs=100, imgsz=640)

Ultralytics 8.3.23 🚀 Python-3.10.12 torch-2.5.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=/content/data.yaml, epochs=100, time=None, patience=100, batch=16, imgsz=640, save=True, save_period=-1, cache=False, device=None, workers
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultralytics/Arial.ttf'...
100%|██████████| 755k/755k [00:00<00:00, 20.9MB/s]
Overriding model.yaml nc=80 with nc=52

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                  -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                  -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 11             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 12                  -1  1    148224  ultralytics.nn.modules.block.C2f             [384, 128, 1]
 13                  -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
 14             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 15                  -1  1     37248  ultralytics.nn.modules.block.C2f             [192, 64, 1]
 16                  -1  1     36992  ultralytics.nn.modules.conv.Conv             [64, 64, 3, 2]
 17            [-1, 12]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 18                  -1  1    123648  ultralytics.nn.modules.block.C2f             [192, 128, 1]
 19                  -1  1    147712  ultralytics.nn.modules.conv.Conv             [128, 128, 3, 2]
 20             [-1, 9]  1         0  ultralytics.nn.modules.conv.Concat           [1]
 21                  -1  1    493056  ultralytics.nn.modules.block.C2f             [384, 256, 1]
 22        [15, 18, 21]  1    761452  ultralytics.nn.modules.head.Detect           [52, [64, 128, 256]]
Model summary: 225 layers, 3,020,988 parameters, 3,020,972 gradients, 8.2 GFLOPs
```

4 Model Training

SSD (Single Shot Multibox Detector)

```
# Run training!
!python /content/models/research/object_detection/model_main_tf2.py \
    --pipeline_config_path={pipeline_file} \
    --model_dir={model_dir} \
    --alsologtostderr \
    --num_train_steps={num_steps} \
    --sample_1_of_n_eval_examples=1
```

```
 'Loss/localization_loss': 0.041699283,
 'Loss/regularization_loss': 0.102919206,
 'Loss/total_loss': 0.23079742,
 'learning_rate': 0.07407206}
I1028 02:08:31.898431 134995250705024 model_lib_v2.py:708] {'Loss/classification_loss': 0.086178936,
 'Loss/localization_loss': 0.041699283,
 'Loss/regularization_loss': 0.102919206,
 'Loss/total_loss': 0.23079742,
 'learning_rate': 0.07407206}
INFO:tensorflow:Step 9700 per-step time 0.333s
I1028 02:09:05.217246 134995250705024 model_lib_v2.py:705] Step 9700 per-step time 0.333s
INFO:tensorflow:{'Loss/classification_loss': 0.06316814,
 'Loss/localization_loss': 0.022676574,
 'Loss/regularization_loss': 0.10246658,
 'Loss/total_loss': 0.1883113,
 'learning_rate': 0.073937014}
I1028 02:09:05.217570 134995250705024 model_lib_v2.py:708] {'Loss/classification_loss': 0.06316814,
 'Loss/localization_loss': 0.022676574,
 'Loss/regularization_loss': 0.10246658,
 'Loss/total_loss': 0.1883113,
 'learning_rate': 0.073937014}
INFO:tensorflow:Step 9800 per-step time 0.333s
```

5 Testing and Evaluation

YOLO (You Only Look Once)

```python
from ultralytics import YOLO

# Load your model
trained_model = YOLO('/content/runs/detect/train/weights/best.pt')  # Replace with your trained model path

# Evaluate the model on the validation dataset
metrics = trained_model.val()  # This outputs precision, recall, mAP@50, etc.

print(metrics)
```

```
Ultralytics 8.3.23 🚀 Python-3.10.12 torch-2.5.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 168 layers, 3,015,788 parameters, 0 gradients, 8.1 GFLOPs
val: Scanning /content/valid/labels.cache... 108 images, 0 backgrounds, 0 corrupt: 100%|██████████| 108/108 [00:00<?, ?it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 7/7 [00:03<00:00,  2.21it/s]
                   all        108        658      0.967      0.977      0.992      0.851
```

METRIC

mAP50:	0.99
mAP50-95:	0.85
Precision:	0.97
Recall:	0.98

5 Testing and Evaluation

SSD (Single Shot Multibox Detector)

```
%%bash
git clone https://github.com/Cartucho/mAP /content/mAP
cd /content/mAP
rm input/detection-results/*
rm input/ground-truth/*
rm input/images-optional/*
wget https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/util_scripts/calculate_map_cartucho.py
```

```
Cloning into '/content/mAP'...
--2024-10-28 03:13:00--  https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/util_scripts/calculate_map_cartucho.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5397 (5.3K) [text/plain]
Saving to: ‘calculate_map_cartucho.py’

     0K .....                                                 100% 53.7M=0s

2024-10-28 03:13:01 (53.7 MB/s) - ‘calculate_map_cartucho.py’ saved [5397/5397]
```

METRIC

mAP50:	0.82
mAP50-95:	0.30
Precision:	0.39
Recall:	0.30

5 Testing and Evaluation

VISUAL RESULTS

YOLO (You Only Look Once)

SSD (Single Shot Multibox Detector)

6 Discussion of Challenges

CHALLENGES

- Struggle with precise localization for small objects due to its grid-based prediction approach.
- Poor performance in precision and recall can lead to many false positives and missed detections, making it less reliable in critical applications.
- Not effectively manage variations in object scales and aspect ratios, impacting its detection capability.

6 Discussion of Challenges

LEARNINGS

- YOLO is suitable for real-time applications due to its speed, making it ideal for scenarios like autonomous driving and surveillance.
- Effective hyperparameter tuning and training on diverse datasets can enhance YOLO’s accuracy and robustness.
- The limitations observed in SSD highlight the need for improved architectures or methodologies that better handle detection tasks, particularly for small and overlapping objects.
- High-quality, diverse training data is crucial for any detection model to perform well, underscoring the importance of dataset selection and preparation.

7 Conclusion and Next Steps

Conclusion

The evaluation shows that YOLO is far more effective than SSD in the context of object detection. Its high mAP scores, combined with excellent precision and recall, suggest it is better suited for tasks requiring high accuracy and reliability. In contrast, SSD's lower performance metrics indicate potential limitations in its application for similar tasks.

Next Step

Further evaluate YOLO’s performance on various datasets and conditions, optimize it for specific use cases, and begin the implementation process, prioritizing its integration into applications where high accuracy and reliability in object detection are critical.

Thank You for Listening!