Automatic Target Detection (ATD)& Recognition (ATR)

Centre for Airborne System – IES Project

Joe Wilson Thamiyan C
RANGSONS AEROSPACE PVT LTD | BENGALORE

Automatic Target Detection (ATD) & Recognition (ATR) System

ATD/ATR is a computer vision-based system that detects, extracts, and recognizes defense targets.

(ATD: Ammunition Dump, Blastpen, Bridge, Cooling Tower, Destroyer, Frigate, Merchant Ship, Submarine, Switch Gear Yard, Tank, Hangar, Helipad, Military Aircraft, Passenger Aircraft, Runway Mark, Storage Tank, Tarmac, Transport Aircraft, Vehicle).

(ATR: C17, C130, F16, Mirage, Training Aircraft, Single Rotor Helicopter, Double Rotor Helicopter) from images.

1. High-Level Architecture

The (ATD/ATR) system consists of multiple components working together in real time. The key components are:

- Data Acquisition (Open-Source platform)
- Environment Setup through offline mode
- Choosing the annotation tools
- Data Preprocessing
- Docker container included (DGX A100)
- Computer Vision-based Object-Detection
- Visualization & Inference

2. Technical Architecture

This section details the technology stack and workflow of the (ATR/ATR) object-detection system.

Technology Stack:

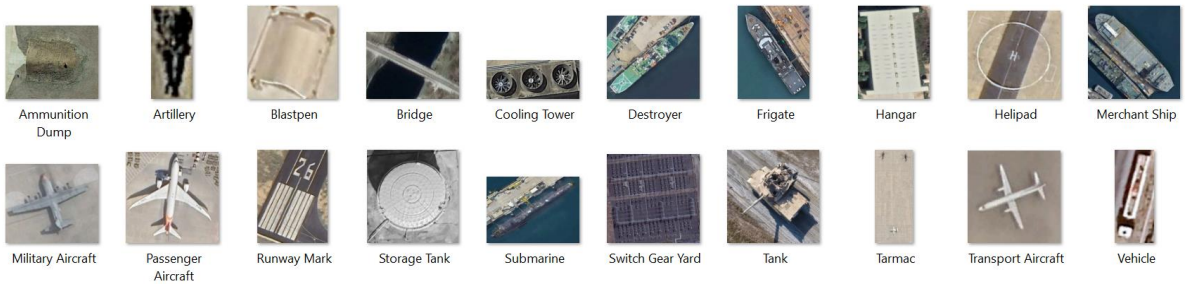
- Data Acquisition: Google-Earth / UMBRA (SAR) Images.
- Data Open source: DOTA / Roboflow
- Environment Setup: PYPI package / Docker Image / Standalone mode.
- Annotation Tools: Ro-labeling / X-any labeling.
- Deep learning Model: REDET (Supporting) / STD+HVIT (provided) / Mask-RCNN (POC) Camouflaged.
- Language: Python
- Framework: Pytorch
- Docker container name: redet-experiment

End-to-End Workflow Components:

1. Data Acquisition Module:

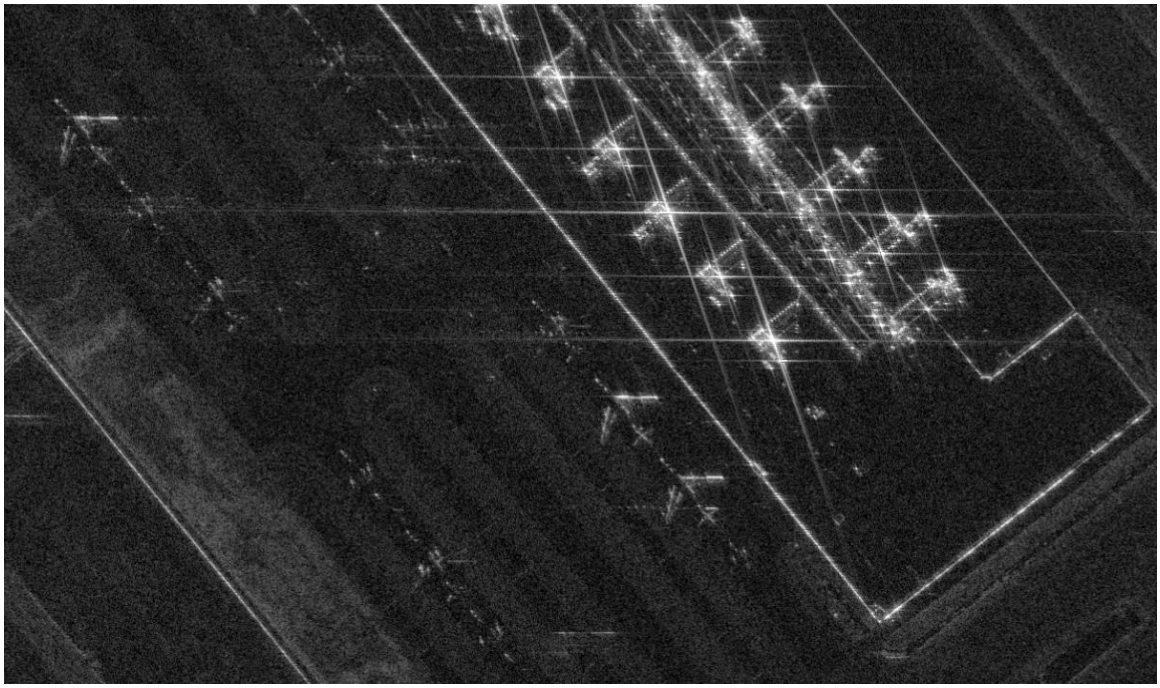
- Description: Acquires data from various sources, including Google-Earth, DOTA, RoboFlow, and UMBRA (SAR) images.
- Functionality: Data ingestion, preprocessing, and format conversion.
- Resolution: Max 8000 ~ 4000 dimensional

- Angle of Capture: Must ensure that the required targets are visible from different angle



Preview of ATD targets

Preview of ATR targets



Preview of SAR targets (Only Aircraft)

Preview of IR targets

2. **Environment Setup:**
    - Approximately 350+ libraries to bring through offline mode.

3. **Annotation Tool:**
    - Ro-labeling tool: Use for annotating the images with OBB direction.
    - X-AnyLabeling tool: To verify, export the DOTA Format, and Visualization crop images.

4. **Data Preprocessing Module:**
    - Description: Preprocesses the acquired data to enhance image quality and prepare it for object detection.
    - Functionality: Image resizing, noise reduction, contrast enhancement.

Code snippet:

Script-1: Common separate function

```
import os
path = r"D:\Dataset\JOE\DOTA\labels"
for i in os.listdir(path):
    try:
        with open(path+ "/" + i) as f:
            data = f.read()
        data = data.replace(","," ")
        with open(path+ "/" + i,"w") as f:
            f.write(data)
    except:
        print(i)
print("Successfully completed")
```

Script-2: Count Instance for target classes

```
import os

import pandas as pd

path1 = r"C:\Users\dev-2\Joe\dataset\LIVE\train_general_classes\reworks\merged-overall\labels"

unique_labels = []

labels_count = {}

file_count = 0

for i in os.listdir(path1): # filename: [I000001, I000002, I000003]

    try:

        file_count += 1

        with open(path1 + "/" + i) as f:

            data = f.readlines() # annotation values: (x1,y1), (x2,y2), (x3,y3), (x4,y4),
class_label, diffculty

            count = 0

            for j in data:

                 split_data = j.split(" ")  # space separated values or comma separated
values.

                 unique_labels.append(split_data[8]) # dict: {target_classes} filter the
unique_classes

    except:

        pass


disc=list(unique_labels)


for k in disc:

    if k in labels_count:

        labels_count[k]+=1

    else:

        labels_count[k]=1

print("target_classes:", labels_count)


# Convert into DataFrame

df=pd.DataFrame(list(labels_count.items()),columns=['value','count'])

print(df)
```

Script-3: Choosing the selected classes in the txt file and moving into designation path

---

```
import os

import shutil

lst_=['Runway_Mark']

path = r"\\90.0.100.238\isr\Joe\dataset-DGX-master\val-20240906\labels"
new_path = r"\\90.0.100.238\isr\Joe\dataset-DGX-master\ac"
for i in os.listdir(path):
    try:
        if i.endswith(".txt"):
            with open(os.path.join(path,i)) as f:
                labels = f.readlines()

                for row_number,label in enumerate(labels,start = 1):
                    parts = label.strip().split()
                    coordinates = list(map(float,parts[:8]))
                    class_name = parts[8]
                    if class_name in lst_:
                        mm = path+"/"+i[:-4]
                        shutil.copy(path+"/"+i,new_path)
                        shutil.copy(mm +".jpg",new_path)
                        break
    except:
        print(i)
```

Script-4: Moving label its corresponding all images into new directory

---

```
import os
import shutil
from tqdm import tqdm

img_src_path = r"./val_images_rgb_20250228/6/IMAGES/"
txt_src_path = r"./val_labels_rgb_20250228/6/labelTxt/"

des_txt_src_path = "./val_labels_rgb_20250228/6/test/"

split_data = [i.split('.txt')[0] for i in os.listdir(txt_src_path) if i.endswith('.txt')]


for i in tqdm(split_data):
    try:
        if os.path.exists(os.path.join(txt_src_path, i + '.txt')):
            shutil.copy(os.path.join(img_src_path, i + '.jpg'), des_txt_src_path + i +'.jpg')
    except:
        print(i)
```

Script-5: Removing the unwanted labels in the txt file

```
path = r"D:\Dataset\JOE\ATR\USA-Dataset\verification\txt"

lst_=['Helipad', 'Hangar', 'Military_Aircraft', 'Storage', 'Blastpen', 'Doomed_Fuel_Tank']

for i in os.listdir(path):
    try:
        if i.endswith(".txt"):
            with open(os.path.join(path,i)) as f:
                labels = f.readlines()
                line = []
                for row_number,label in enumerate(labels,start = 1):
                    parts = label.split(' ')
                    coordinates = list(map(float,parts[:8]))
                    class_name = parts[8]
                    if class_name not in lst_:
                        line.append(label)
                with open(os.path.join(path,i),"w") as f:
                    f.writelines(line)
    except:
        print(i)
```

Script-6: Rename the filename

```
import os
path = "./classification_dataset/vehicle/"
var = 0
for i in sorted(os.listdir(path)):
    if i.endswith('jpg'):
        strt = i.split(".")
        strts = str(strt[0]) + ".jpg"
        if os.path.isfile(path + strts):
            os.rename(path+i, path+'vehicle_'+str(var)+".jpg")
            var += 1
path = "./classification_dataset/vehicle/"
var = 1
```

Script-6: Crop image based on are of interest (AOI)

```
%%time

import os

import cv2
from PIL import Image

def create_directory(dir_name):
    if not os.path.exists(dir_name):
        os.makedirs(dir_name)

def crop_and_save_image(image_path, annotations, output_dir):
    image = Image.open(image_path)

    for annotation in annotations:
        parts = annotation.strip().split(",")
        if len(parts) != 10:
            continue

        coords = list(map(float, parts[:8]))
        label = parts[8].strip()

        left = min(coords[0::2])
        top = min(coords[1::2])
        right = max(coords[0::2])
        bottom = max(coords[1::2])

        cropped_image = image.crop((left, top, right, bottom))
        label_dir = os.path.join(output_dir, label)
        create_directory(label_dir)
        cropped_image_name = f"{label}_{int(left)}_{int(top)}.jpg"
        cropped_image_path = os.path.join(label_dir, cropped_image_name)
        cropped_image.save(cropped_image_path)

unique_names = []
path_1 = "./aoi/"
for i in os.listdir(path_1):
    unique_names.append(i.split(".")[0])
unique_names = list(set(unique_names))

output_dir = "testing_with_google_earth"

for i in unique_names:
    image_path = path_1 + "/" + i +".jpg"
    annotations_file = path_1 + "/" + i+".txt"
    print("****:",annotations_file)
#     annotations_file = path_1 + "/" + i + ".txt"


    annotations = []
    with open(annotations_file, "r") as file:
        for line in file:
            annotations.append(line)
#     print('list:', annotations)
    try:
        crop_and_save_image(image_path, annotations, output_dir)
    except:
        print("incorrect", image_path)
print("successfully completed")
```

Script-7: Replacing old label into new label

```
import os

label_path = r"C:\Users\dev-2\Joe\dataset\LIVE\train_general_classes\reworks\merged-
overall\labels"
for i in os.listdir(label_path):
    try:
        if i.endswith(".txt"):
            with open(os.path.join(label_path,i),"r") as f:
                data = f.readlines()

            new_class_name = "Hangar"
            for row_n,v in enumerate(data,start = 1):
                parts = data[row_n-1].strip().split()
                if parts[8] == "Hanger":
                    parts[8] = new_class_name
                    data[row_n-1] = " ".join(parts) + "\n"

                    with open(os.path.join(label_path,i),"w") as f:
                        f.writelines(data)
    except:
        print(i)
```

Script-8: Specific filtered text file corresponding respective images copy to new directory

---

```
import os
import shutil
from tqdm import tqdm

img_src_path = r"./val_images_rgb_20250228/6/IMAGES/"
txt_src_path = r"./val_labels_rgb_20250228/6/labelTxt/"

des_txt_src_path = "./val_labels_rgb_20250228/6/test/"

split_data = [i.split('.txt')[0] for i in os.listdir(txt_src_path) if i.endswith('.txt')]


for i in tqdm(split_data):
    try:
        if os.path.exists(os.path.join(txt_src_path, i + '.txt')):
            shutil.copy(os.path.join(img_src_path, i + '.jpg'), des_txt_src_path + i +'.jpg')
    except:
        print(i)
```

Script-9: Specific filtered Images corresponding respective text file copy to new directory

---

```
import os
import shutil
from tqdm import tqdm

img_src_path = r"E:\DEMO\val_subClasses\images\additional"
txt_src_path = r"E:\DEMO\train_subClasses\labels"

des_txt_src_path = "E:\\DEMO\\val_subClasses\\images\\addition_txt\\"

split_data = [i.split('.txt')[0] for i in os.listdir(txt_src_path) if i.endswith('.txt')]


for i in tqdm(split_data):
    try:
        if os.path.exists(os.path.join(img_src_path, i + '.jpg')):
            shutil.copy(os.path.join(txt_src_path, i + '.txt'), des_txt_src_path + i +'.txt')
    except:
        print(i)
```

---

5. Object Detection Module:

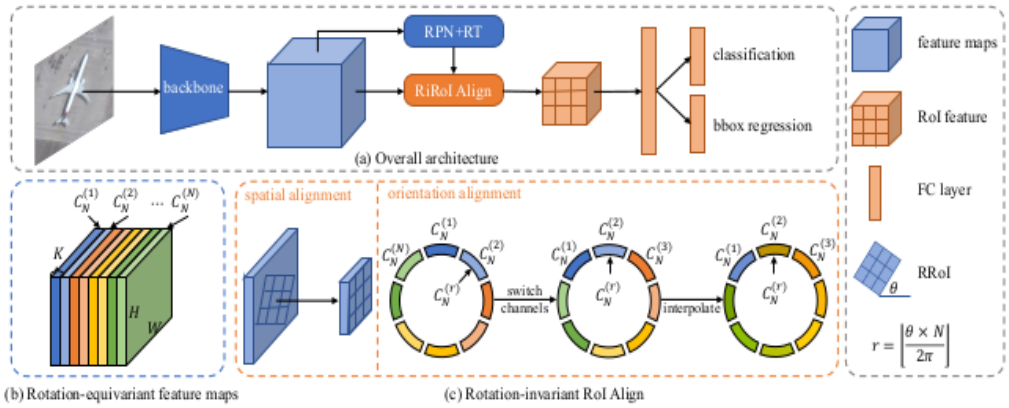
- Description: Detects and extracts potential targets from the preprocessed images.
- Functionality: Object detection, feature extraction, and classification.
- Model: REDET, STD+HVIT

About Model: Rotation-Equivariant Detector (REDET)

- Overall architecture of the proposed rotation-equivariant detector. We first adopt the rotation-equivariant backbone to extract rotation-equivariant features, followed by an RPN and ROI Transformer (RT) to generate RRoIs. Then we use a novel Rotation-invariant RoI Align.

- (RiROI Align) to produce rotation-invariant features for RoI-wise classification and bounding box (BBox) regression. Rotation-equivariant feature maps. Under the cyclic group CN, the rotation-equivariant feature maps with the size (K, N, H, W) have N orientation channels, and each orientation channel is corresponding to an element in CN.

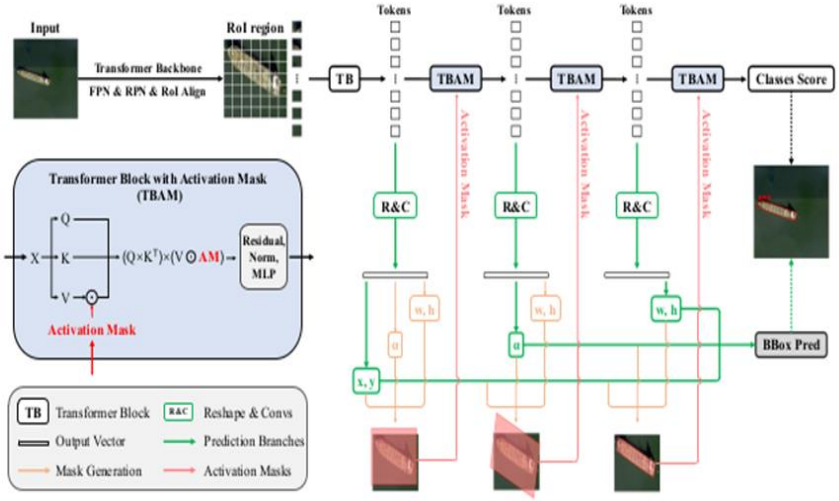
- The proposed RiRoI Align consists of two parts: spatial alignment and orientation alignment. For and RRoI (x, y, w, h, O), spatial alignment warps the RRoI from the spatial dimension, while orientation alignment circularly switches orientation channels and interpolates features to produce completely rotation-invariant features.



- We worked on 3 backbones for check the performance of the model (RESNET-50, 101, 152).
- Changes in the parameters like Optimizer, Learning rate, Batch size, epochs.

About Model: Spatial Transform Decoupling (STD+HViT)

- Spatial Transform Decoupling (STD), providing a simple-yet-effective solution for oriented object detection with ViTs.
- Built upon stacked ViT blocks, STD utilizes separate network branches to predict the position, size, and angle of bounding boxes, effectively harnessing the spatial transform potential of ViTs in a divide-and-conquer fashion.
- Check this model on DOTA benchmark of object detection in Aerial image on DOTA, this model is the 2nd rank on DOTA dataset with mAP (82.24%)



As per model changes which required to our use case:

- We worked on 3-backbone for check the performance of the model (Orentient FRCNN, ViT, HiViT).
- Changes in the parameters like Optimizer, Learning rate, Batch-size, and epochs etc.

6. Infrastructure:

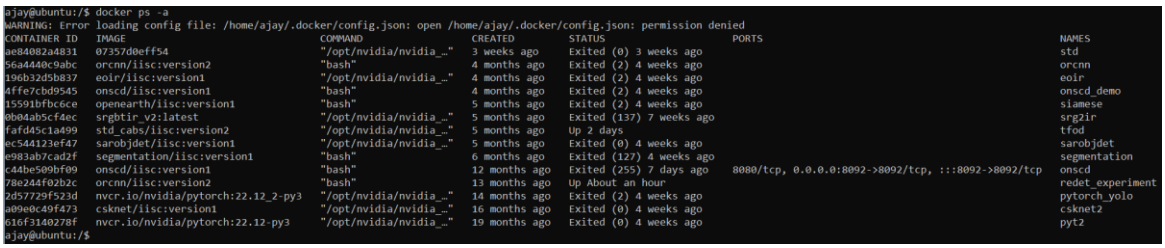
- Containerization: Docker
- Container Name: redet_experiment
- Image Name: orcnn/iisc
- Container ID: d56dbc843ede
- GPU: NVIDIA DGX A100

Docker Module Workflow:

Step-1: Login into DGX environment passed through the credential.

```
C:\Users\dev-2>ssh ajay@192.168.0.129
ajay@192.168.0.129's password:
Welcome to NVIDIA DGX Server Version 5.4.2 (GNU/Linux 5.4.0-137-generic x86_64)

  System information as of Monday 26 May 2025 10:53:09 AM IST

  System load:  0.8               Processes:                2507
  Usage of /:   73.7% of 1.72TB   Users logged in:          1
  Memory usage: 0%                IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                IPv4 address for enp226s0: 192.168.0.129

The system has 0 critical alerts and 12 warnings. Use 'sudo nvsm show alerts' for more details.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Mon May 26 09:49:58 2025 from 192.168.0.144
Could not chdir to home directory /home/ajay: Permission denied
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

-bash: /home/ajay/.bash_profile: Permission denied
ajay@ubuntu:/$
ajay@ubuntu:/$
ajay@ubuntu:/$
```

Step-2: List out the all Docker container



```
ajay@ubuntu:/$ docker ps -a
WARNING: Error loading config file: /home/ajay/.docker/config.json: open /home/ajay/.docker/config.json: permission denied
CONTAINER ID   IMAGE                                COMMAND                  CREATED         STATUS                     PORTS                                                NAMES
ae84082a4831   07357d0eff54                         "/opt/nvidia/nvidia_…"   3 weeks ago     Exited (0) 3 weeks ago                                                  std
56a4440c9abc   orcnn/iisc:version2                  "bash"                   4 months ago    Exited (2) 4 weeks ago                                                  orcnn
196b32d5b837   eoir/iisc:version1                   "/opt/nvidia/nvidia_…"   4 months ago    Exited (2) 4 weeks ago                                                  eoir
4ffe7cbd9545   onscd/iisc:version1                  "bash"                   4 months ago    Exited (2) 4 weeks ago                                                  onscd_demo
15591bfbc6ce   openearth/iisc:version1              "bash"                   5 months ago    Exited (2) 4 weeks ago                                                  siamese
0b04ab5cf4ec   srgbtir_v2:latest                    "/opt/nvidia/nvidia_…"   5 months ago    Exited (137) 7 weeks ago                                                srg2ir
fafd45c1a499   std_cabs/iisc:version2               "/opt/nvidia/nvidia_…"   5 months ago    Up 2 days                                                               tfod
ec544123ef47   sarobjdet/iisc:version1              "/opt/nvidia/nvidia_…"   5 months ago    Exited (0) 4 weeks ago                                                  sarobjdet
e983ab7cad2f   segmentation/iisc:version1           "bash"                   6 months ago    Exited (127) 4 weeks ago                                                segmentation
c44be509bf09   onscd/iisc:version1                  "bash"                   12 months ago   Exited (255) 7 days ago    8080/tcp, 0.0.0.0:8092->8092/tcp, :::8092->8092/tcp   onscd
78e244f02b2c   orcnn/iisc:version2                  "bash"                   13 months ago   Up About an hour                                                        redet_experiment
2d57729f523d   nvcr.io/nvidia/pytorch:22.12_2-py3   "/opt/nvidia/nvidia_…"   14 months ago   Exited (2) 4 weeks ago                                                  pytorch_yolo
a09e0c49f473   csknet/iisc:version1                 "/opt/nvidia/nvidia_…"   16 months ago   Exited (0) 4 weeks ago                                                  csknet2
616f3140278f   nvcr.io/nvidia/pytorch:22.12-py3     "/opt/nvidia/nvidia_…"   19 months ago   Exited (0) 4 weeks ago                                                  pyt2
ajay@ubuntu:/$
```

Start the container cmd: docker start redet_experiment

Attach the container cmd: docker attach redet_experiment

Step-3: Launched the jupyter-lab in the container

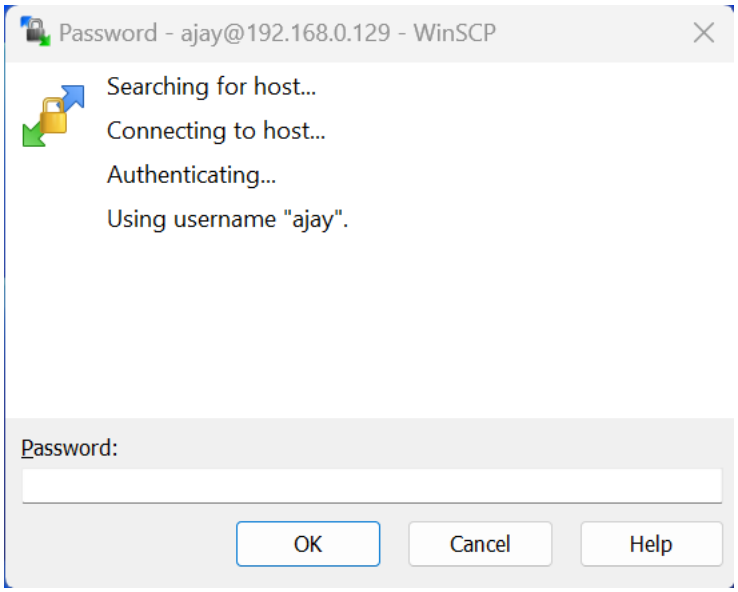
```
Cmd: jupyter-lab –ip=0.0.0.0 –port=8888 –no-browser –NotebookApp.token='123'
–allow-root
```

Step-4: Hit the IP-address in any browser

```
192.168.0.129: port no
```

Step-5: Data Transferring local into DGX

Tool: Used the tool on Win SCP




```
Docker cmd: docker cp ./local_path redet_experiment:/raid/home/joe/folderName
```

(Local to Docker)

```
Docker cmd: docker cp redet_experiment:/raid/home/joe/folderName
./local_path_folderName
```

(Docker to local)

```
Docker cmd: ls –l | wc –l # To check the total no.of count in the folder
```

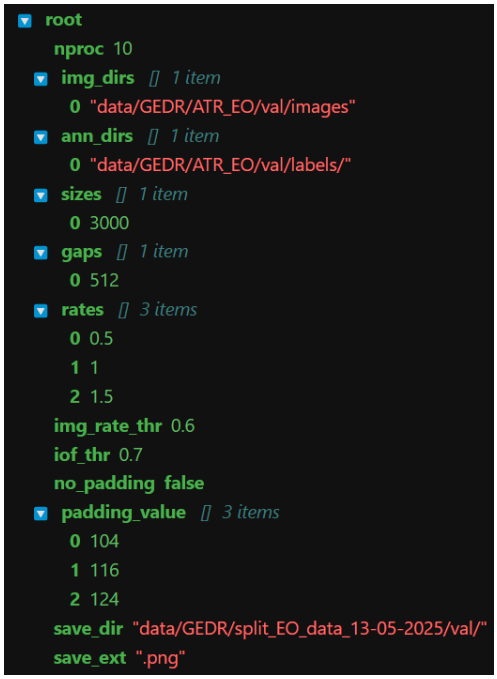
Step-6: Split for train, Val, and test using img_split.py in the tools folder

Path: cd tools/data/dota/split/split_configs

Open: vi ms_trainval.json

Path: cd tools/data/dota/split/split_configs

Open: vi ms_val.json



Cmd: python tools/data/dota/split/img_split.py –base-json tools/data/dota/split/split_configs/ms_trainval.json

Cmd: python tools/data/dota/split/img_split.py –base-json tools/data/dota/split/split_configs/ms_val.json

Step-7: Modify the config.py files

```
_base_ = [
    '../_base_/datasets/hrsc.py', '../_base_/schedules/schedule_3x.py',
    '../_base_/default_runtime.py'
]


angle_version = 'le90'
model = dict(
    type='ReDet',
    backbone=dict(
        type='ReResNet',
        depth=50,
        num_stages=4,
        out_indices=(0, 1, 2, 3),
        frozen_stages=1,
        style='pytorch',
        pretrained='./work_dirs/re_resnet50_c8_batch256-25b16846.pth'),
    neck=dict(
        type='ReFPN',
        in_channels=[256, 512, 1024, 2048],
        out_channels=256,
        num_outs=5),
    rpn_head=dict(
        type='RotatedRPNHead',
        in_channels=256,
        feat_channels=256,
        version=angle_version,
        anchor_generator=dict(
            type='AnchorGenerator',
            scales=[8],
            ratios=[0.5, 1.0, 2.0],
            strides=[4, 8, 16, 32, 64]),
        bbox_coder=dict(
            type='DeltaXYWHBBoxCoder',
            target_means=[.0, .0, .0, .0],
            target_stds=[1.0, 1.0, 1.0, 1.0]),
```

```
    loss_cls=dict(
        type='CrossEntropyLoss', use_sigmoid=True, loss_weight=1.0),
    loss_bbox=dict(type='SmoothL1Loss', beta=1.0 / 9.0, loss_weight=1.0)),
roi_head=dict(
    type='RoITransRoIHead',
    version=angle_version,
    num_stages=2,
    stage_loss_weights=[1, 1],
    bbox_roi_extractor=[
        dict(
            type='SingleRoIExtractor',
            roi_layer=dict(
                type='RoIAlign', output_size=7, sampling_ratio=0),
            out_channels=256,
            featmap_strides=[4, 8, 16, 32]),
        dict(
            type='RotatedSingleRoIExtractor',
            roi_layer=dict(
                type='RiRoIAlignRotated',
                out_size=7,
                num_samples=2,
                num_orientations=8,
                clockwise=True),
            out_channels=256,
            featmap_strides=[4, 8, 16, 32]),
    ],
    bbox_head=[
        dict(
            type='RotatedShared2FCBBoxHead',
            in_channels=256,
            fc_out_channels=1024,
            roi_feat_size=7,
            num_classes=1,
            bbox_coder=dict(
                type='DeltaXYWHAHBBoxCoder',
                angle_range=angle_version,
```

```
                    norm_factor=2,
                    edge_swap=True,
                    target_means=[0., 0., 0., 0., 0.],
                    target_stds=[0.1, 0.1, 0.2, 0.2, 0.1]),
                reg_class_agnostic=True,
                loss_cls=dict(
                    type='CrossEntropyLoss',
                    use_sigmoid=False,
                    loss_weight=1.0),
                loss_bbox=dict(type='SmoothL1Loss', beta=1.0,
                               loss_weight=1.0)),
            dict(
                type='RotatedShared2FCBBoxHead',
                in_channels=256,
                fc_out_channels=1024,
                roi_feat_size=7,
                num_classes=1,
                bbox_coder=dict(
                    type='DeltaXYWHAOBBoxCoder',
                    angle_range=angle_version,
                    norm_factor=None,
                    edge_swap=True,
                    proj_xy=True,
                    target_means=[0., 0., 0., 0., 0.],
                    target_stds=[0.05, 0.05, 0.1, 0.1, 0.05]),
                reg_class_agnostic=False,
                loss_cls=dict(
                    type='CrossEntropyLoss',
                    use_sigmoid=False,
                    loss_weight=1.0),
                loss_bbox=dict(type='SmoothL1Loss', beta=1.0, loss_weight=1.0))
        ]),
    train_cfg=dict(
        rpn=dict(
            assigner=dict(
                type='MaxIoUAssigner',
```

```
            pos_iou_thr=0.7,
            neg_iou_thr=0.3,
            min_pos_iou=0.3,
            match_low_quality=True,
            ignore_iof_thr=-1),
        sampler=dict(
            type='RandomSampler',
            num=256,
            pos_fraction=0.5,
            neg_pos_ub=-1,
            add_gt_as_proposals=False),
        allowed_border=0,
        pos_weight=-1,
        debug=False),
    rpn_proposal=dict(
        nms_pre=2000,
        max_per_img=2000,
        nms=dict(type='nms', iou_threshold=0.7),
        min_bbox_size=0),
    rcnn=[
        dict(
            assigner=dict(
                type='MaxIoUAssigner',
                pos_iou_thr=0.5,
                neg_iou_thr=0.5,
                min_pos_iou=0.5,
                match_low_quality=False,
                ignore_iof_thr=-1,
                iou_calculator=dict(type='BboxOverlaps2D')),
            sampler=dict(
                type='RandomSampler',
                num=512,
                pos_fraction=0.25,
                neg_pos_ub=-1,
                add_gt_as_proposals=True),
            pos_weight=-1,
```

```
                debug=False),
            dict(
                assigner=dict(
                    type='MaxIoUAssigner',
                    pos_iou_thr=0.5,
                    neg_iou_thr=0.5,
                    min_pos_iou=0.5,
                    match_low_quality=False,
                    ignore_iof_thr=-1,
                    iou_calculator=dict(type='RBboxOverlaps2D')),
                sampler=dict(
                    type='RRandomSampler',
                    num=512,
                    pos_fraction=0.25,
                    neg_pos_ub=-1,
                    add_gt_as_proposals=True),
                pos_weight=-1,
                debug=False)
        ]),
    test_cfg=dict(
        rpn=dict(
            nms_pre=2000,
            max_per_img=2000,
            nms=dict(type='nms', iou_threshold=0.7),
            min_bbox_size=0),
        rcnn=dict(
            nms_pre=2000,
            min_bbox_size=0,
            score_thr=0.05,
            nms=dict(iou_thr=0.1),
            max_per_img=2000)))

img_norm_cfg = dict(
    mean=[123.675, 116.28, 103.53], std=[58.395, 57.12, 57.375], to_rgb=True)
train_pipeline = [
    dict(type='LoadImageFromFile'),
```

```
    dict(type='LoadAnnotations', with_bbox=True),
    dict(type='RResize', img_scale=(800, 512)),
    dict(type='RRandomFlip', flip_ratio=0.5),
    dict(type='Normalize', **img_norm_cfg),
    dict(type='Pad', size_divisor=32),
    dict(type='DefaultFormatBundle'),
    dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels'])
]
test_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(
        type='MultiScaleFlipAug',
        img_scale=(800, 512),
        flip=False,
        transforms=[
            dict(type='RResize'),
            dict(type='Normalize', **img_norm_cfg),
            dict(type='Pad', size_divisor=32),
            dict(type='DefaultFormatBundle'),
            dict(type='Collect', keys=['img'])
        ])
]

data = dict(
    train=dict(pipeline=train_pipeline),
    val=dict(pipeline=test_pipeline),
    test=dict(pipeline=test_pipeline))

evaluation = dict(interval=12, metric='mAP')
optimizer = dict(lr=0.01)
```

../_base_/datasets/dotav1.py

```
# dataset settings
dataset_type = 'DOTADataset'
data_root = 'data/GEDR/'
img_norm_cfg = dict(
    mean=[123.675, 116.28, 103.53], std=[58.395, 57.12, 57.375], to_rgb=True)
train_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='LoadAnnotations', with_bbox=True),
    # dict(type='RandomGrayscale', prob=0.1, channel_weights=(0.25,0.5,0.25)),
    dict(type='RResize', img_scale=(512, 512)),
    dict(type='RRandomFlip', flip_ratio=0.5),
    dict(type='Normalize', **img_norm_cfg),
    dict(type='Pad', size_divisor=32),
    dict(type='DefaultFormatBundle'),
    dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels'])
]
test_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(
        type='MultiScaleFlipAug',
        img_scale=(512, 512),
        flip=False,
        transforms=[
            dict(type='RResize'),
            dict(type='Normalize', **img_norm_cfg),
            dict(type='Pad', size_divisor=32),
            dict(type='DefaultFormatBundle'),
            dict(type='Collect', keys=['img'])
        ])
]
data = dict(
    samples_per_gpu=1,
    workers_per_gpu=16,
    train=dict(
        type=dataset_type,
```

```
        # ann_file=data_root + 'train/annfiles/',
        # img_prefix=data_root + 'train/images/',
        ann_file=data_root + 'ATR_EO/train/labels',
        img_prefix=data_root + 'ATR_EO/train/images/',
        pipeline=train_pipeline),
    val=dict(
        type=dataset_type,
        ann_file=data_root + 'ATR_EO/val/labels/',
        img_prefix=data_root + 'ATR_EO/val/images/',
        pipeline=test_pipeline),
    test=dict(
        type=dataset_type,
        ann_file=data_root + 'ATR_EO/val/labels',
        img_prefix=data_root + 'ATR_EO/val/images/',
        pipeline=test_pipeline))
```

../__base__/schedules/schedule_1x.py

```
# dataset settings
dataset_type = 'DOTADataset'
data_root = 'data/GEDR/'
img_norm_cfg = dict(
    mean=[123.675, 116.28, 103.53], std=[58.395, 57.12, 57.375], to_rgb=True)
train_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='LoadAnnotations', with_bbox=True),
    # dict(type='RandomGrayscale', prob=0.1, channel_weights=(0.25,0.5,0.25)),
    dict(type='RResize', img_scale=(512, 512)),
    dict(type='RRandomFlip', flip_ratio=0.5),
    dict(type='Normalize', **img_norm_cfg),
    dict(type='Pad', size_divisor=32),
    dict(type='DefaultFormatBundle'),
    dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels'])
]
test_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(
```

```
        type='MultiScaleFlipAug',
        img_scale=(512, 512),
        flip=False,
        transforms=[
            dict(type='RResize'),
            dict(type='Normalize', **img_norm_cfg),
            dict(type='Pad', size_divisor=32),
            dict(type='DefaultFormatBundle'),
            dict(type='Collect', keys=['img'])
        ])
]
data = dict(
    samples_per_gpu=1,
    workers_per_gpu=16,
    train=dict(
        type=dataset_type,
        # ann_file=data_root + 'train/annfiles/',
        # img_prefix=data_root + 'train/images/',
        ann_file=data_root + 'ATR_EO/train/labels',
        img_prefix=data_root + 'ATR_EO/train/images/',
        pipeline=train_pipeline),
    val=dict(
        type=dataset_type,
        ann_file=data_root + 'ATR_EO/val/labels/',
        img_prefix=data_root + 'ATR_EO/val/images/',
        pipeline=test_pipeline),
    test=dict(
        type=dataset_type,
        ann_file=data_root + 'ATR_EO/val/labels',
        img_prefix=data_root + 'ATR_EO/val/images/',
        pipeline=test_pipeline))
```

tools/dist_train.sh

```
#!/usr/bin/env bash

CONFIG=$1

GPUS=$2

NNODES=${NNODES:-1}

NODE_RANK=${NODE_RANK:-0}

PORT=${PORT:-29502}

MASTER_ADDR=${MASTER_ADDR:-"127.0.0.1"}


PYTHONPATH="$(dirname $0)/..":$PYTHONPATH \

python -m torch.distributed.launch \

    --nnodes=1 \

    --node_rank=$NODE_RANK \

    --master_addr=$MASTER_ADDR \

    --nproc_per_node=$GPUS \

    --master_port=$PORT \

    $(dirname "$0")/train.py \

    $CONFIG \

    --seed 0 \

    --launcher pytorch ${@:3}
```

Steps: Train the Model

Cmd: CUDA_VISIBLE_DEVICES=0,1,2,3 tools/dist_train.sh configs/redet/ redet_re50_refpn_1x_dota_le90.py 4

Experimental:

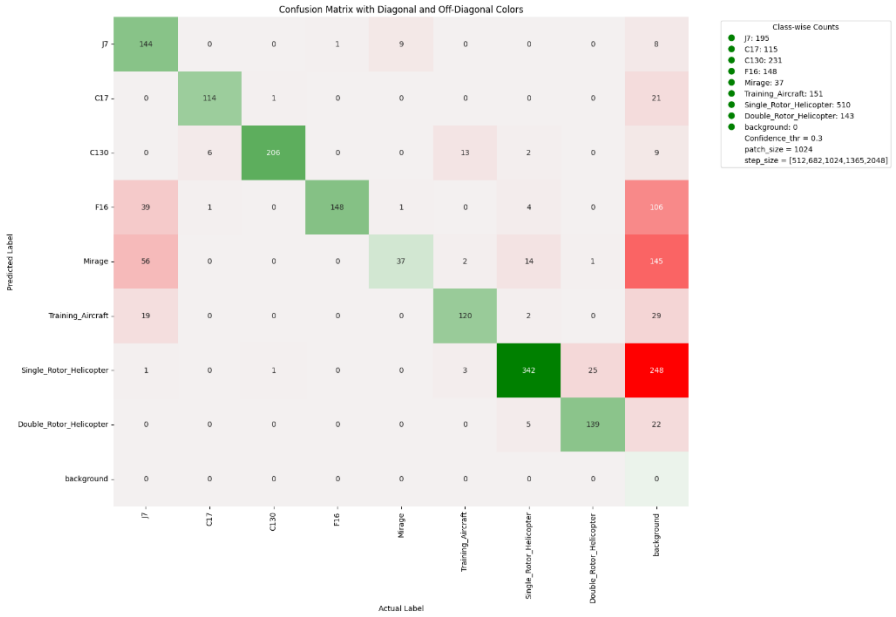
Method: Multi-Scale (MS) - ATD	ATD – (Automatic Target Detection) Hyper-Parameter										
Model	Backbone	Dataset	Img_Count	Categories	Img_Size	Batch_size	Epochs	Experiment	Optimizer	LR	mAP(%)
REDET	RESNET-50	Google Earth	6488	20	512	8	12	Exp0001	SGD	0.01	53.5
	RESNET-101			20	512	8	12	Exp0002	SGD	0.01	56.1
	RESNET-152			20	512	8	12	Exp0003	SGD	0.01	35.9

Method: Multi-Scale (MS) - ATR	ATR - (Automatic Target Recognition) Hyper-Parameter										
Model	Backbone	Dataset	Img_Count	Categories	Img_Size	Batch_size	Epochs	Experiment	Optimizer	LR	mAP(%)
REDET	RESNET-50	Google Earth	4984	8	512	8	12	Exp0001	SGD	0.01	84.0
	RESNET-101			8	512	8	12	Exp0002	SGD	0.01	85.8
	RESNET-152			8	512	8	12	Exp0003	SGD	0.01	76.0

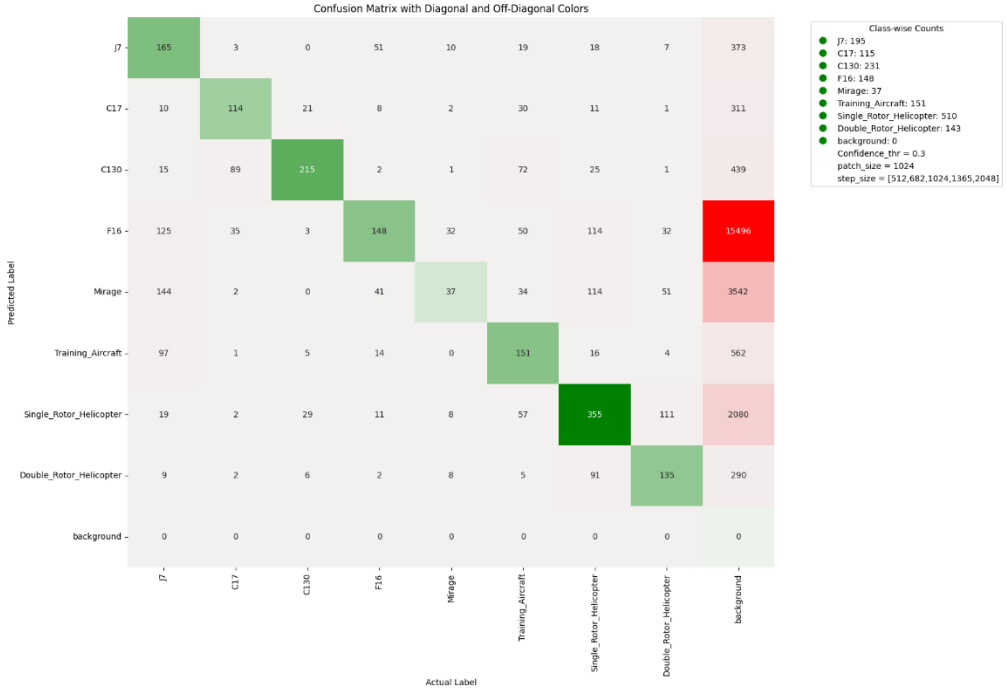
STD+HViT (Automatic Target Recognition)

Method : Multi-Scale (MS)	Hyper-Parameter											
Model	Backbone	Dataset	Img_Count	Categories	Img_Size	Batch_Size	Epochs	Experiment	pretrain	Optimizer	LR	mAP(%)
REDET	RESNET-50	GEDR (Google-Earth) DOTAV1.5	6488	8	1024	8	12	Exp01	Resnet-50	SGD	0.01	84.0
STD	Oriented Faster-RCNN					8	12	Exp01	orcnn_std_hivit	AdamW	0.00025	82.9

Confusion Matrix:



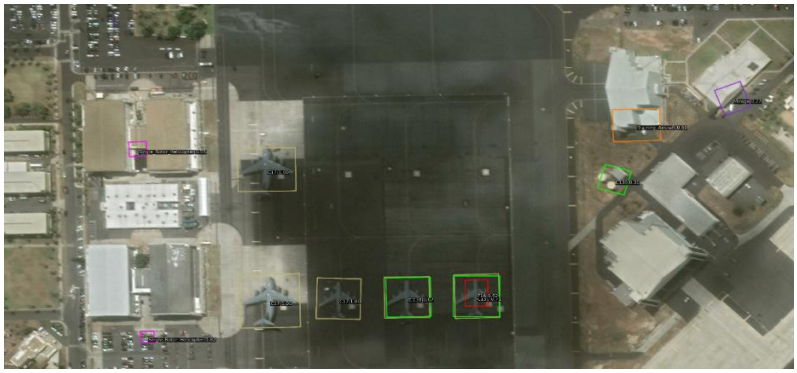
REDET



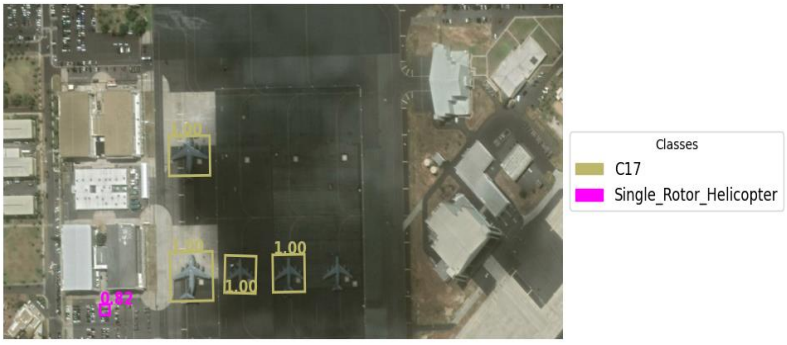
STD+HVIT

7. Inference Module

ORIGINAL IMAGE



REDET



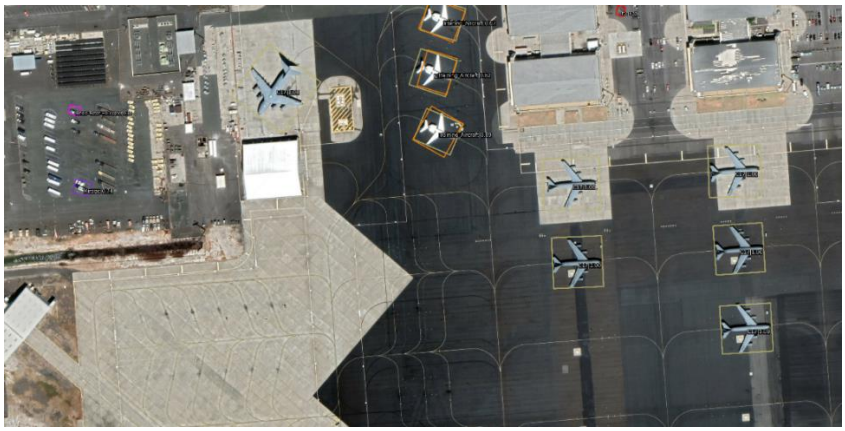
STD

ORIGINAL IMAGE

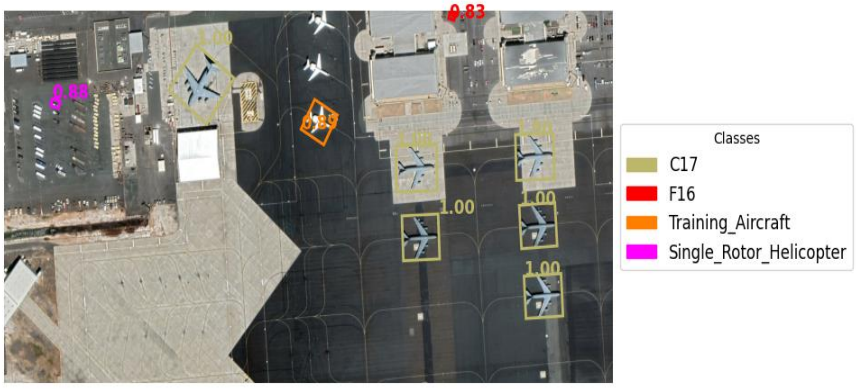
REDET

STD

ORIGINAL IMAGE

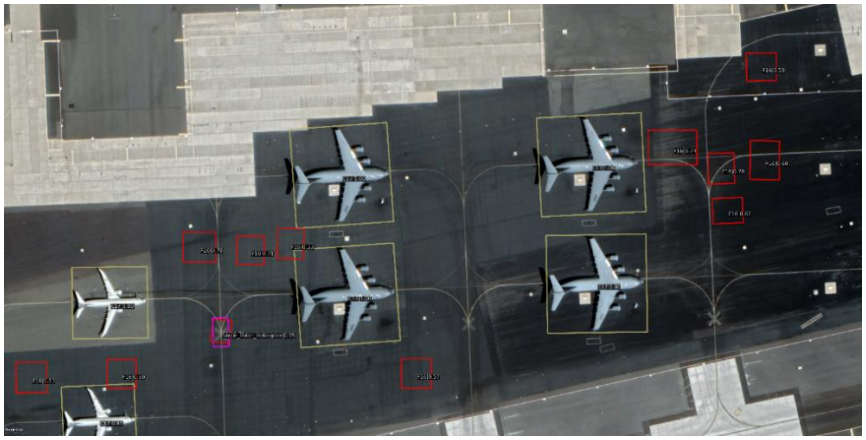


REDET

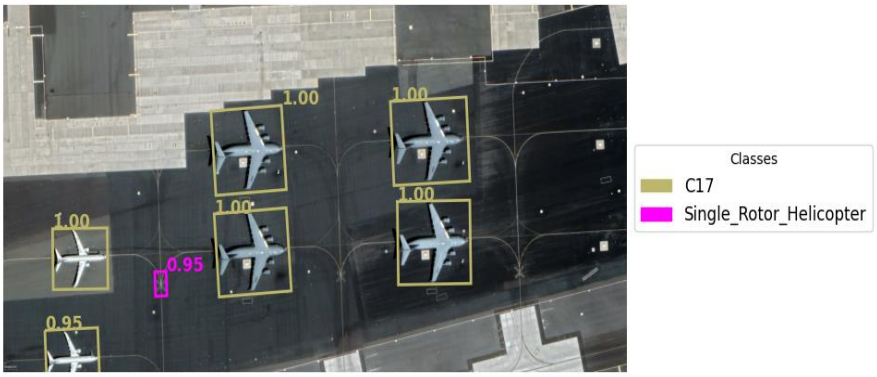


STD

ORIGINAL IMAGE



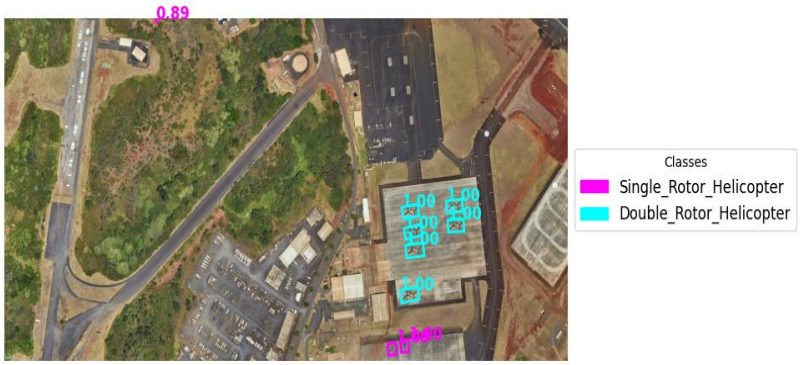
REDET



STD

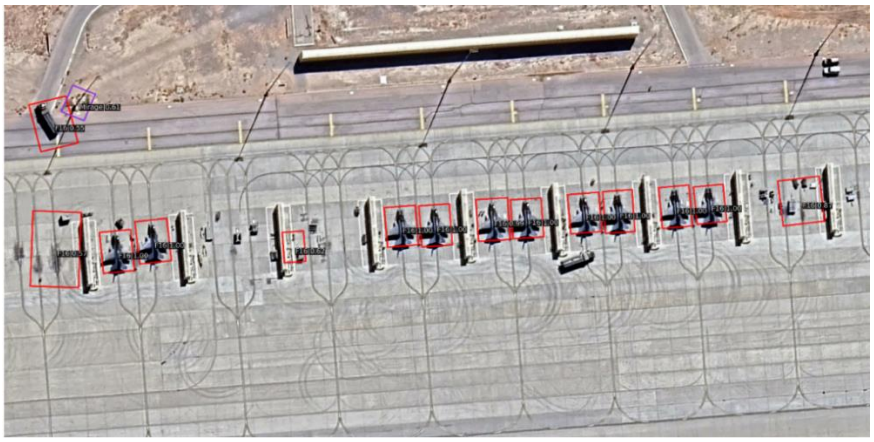
ORIGINAL IMAGE

REDET



STD

ORIGINAL IMAGE



REDET

STD

ORIGINAL IMAGE

REDET

STD

Production

Method: Multi Scale (MS)	ATD/ATR – (Automatic Target Detection & Recognition) (Hyper parameters)										
Model	**TypeOfDetection**	**Backbone**	**Dataset**	**Img_count**	**Categories**	**Img_size**	**Batch_size**	**Epochs**	**Optimizer**	**LR**	**mAP(%)**
REDET	ATD	RESNET-50 + CUTMIX	GEDR		20		16	12	SGD	0.01	85.3
REDET	ATR	RESNET-50	GEDR		8		16	12	SGD	0.01	49
STD+ViT	ATD	ViT	GEDR		20		8	12	AdamW	0.00005	74.5
STD+HViT	ATR	HVIT	GEDR		8		2	12	AdamW	0.00006	89.5

InfraRed (IR) Image

ATD – (Automatic Target Detection)

Description: This model is semi-supervised RGB-to-IR Image to Image Translation. Model Provides IR images at 4 level with different contrast.

Training: Co-registered EO and IR images without any annotations.

Dataset: Google Earth

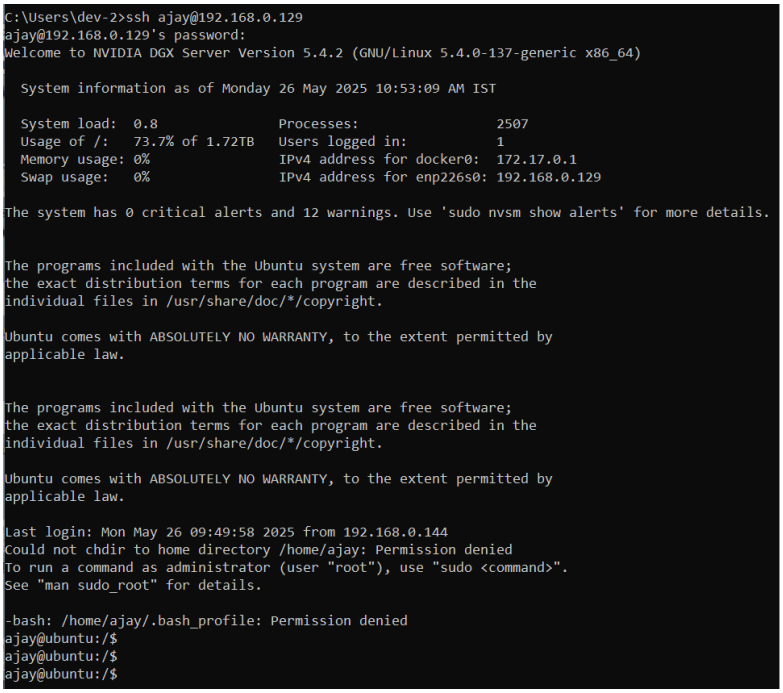
Inference:

- Input: EO images
- Output: It will translate the EO images into IR image at 4 levels

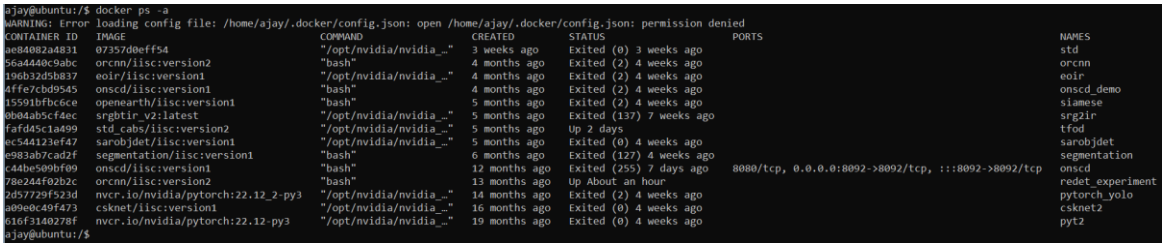
1. Infrastructure:
    - Containerization: Docker
    - Image Name: srgbtir_v2:latest
    - Container Name: srg2ir
    - Container ID: 07357d0eff54
    - Environment Name: UNSB
    - URL: 192.168.0.129:8157
    - Password: Token
    - GPU: NVIDIA DGX A100

    Docker Module Workflow:

    Step-1: Login into DGX environment passed through the credential.



Step-2: List out the all Docker container



```
ajay@ubuntu:/$ docker ps -a
WARNING: Error loading config file: /home/ajay/.docker/config.json: open /home/ajay/.docker/config.json: permission denied
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS                      PORTS                                                                NAMES
ae84082a4831   07357d0eff54                         "/opt/nvidia/nvidia_…"   3 weeks ago    Exited (0) 3 weeks ago                                                                   std
56a4440c9abc   orcnn/iisc:version2                  "bash"                   4 months ago   Exited (2) 4 weeks ago                                                                   orcnn
196b32d5b837   eoir/iisc:version1                   "/opt/nvidia/nvidia_…"   4 months ago   Exited (2) 4 weeks ago                                                                   eoir
4ffe7cbd9545   onscd/iisc:version1                  "bash"                   4 months ago   Exited (2) 4 weeks ago                                                                   onscd_demo
15591bfbc6ce   openearth/iisc:version1              "bash"                   5 months ago   Exited (2) 4 weeks ago                                                                   siamese
0b04ab5cf4ec   srgbtir_v2:latest                    "/opt/nvidia/nvidia_…"   5 months ago   Exited (137) 7 weeks ago                                                                 srg2ir
fafd45c1a499   std_cabs/iisc:version2               "/opt/nvidia/nvidia_…"   5 months ago   Up 2 days                                                                                tfod
ec544123ef47   sarobjdet/iisc:version1              "/opt/nvidia/nvidia_…"   5 months ago   Exited (0) 4 weeks ago                                                                   sarobjdet
e983ab7cad2f   segmentation/iisc:version1           "bash"                   6 months ago   Exited (127) 4 weeks ago                                                                 segmentation
c44be509bf09   onscd/iisc:version1                  "bash"                   12 months ago  Exited (255) 7 days ago     8080/tcp, 0.0.0.0:8092->8092/tcp, :::8092->8092/tcp          onscd
78e244f02b2c   orcnn/iisc:version2                  "bash"                   13 months ago  Up About an hour                                                                         redet_experiment
2d57729f523d   nvcr.io/nvidia/pytorch:22.12_2-py3   "/opt/nvidia/nvidia_…"   14 months ago  Exited (2) 4 weeks ago                                                                   pytorch_yolo
a09e0c49f473   csknet/iisc:version1                 "/opt/nvidia/nvidia_…"   16 months ago  Exited (0) 4 weeks ago                                                                   csknet2
616f3140278f   nvcr.io/nvidia/pytorch:22.12-py3     "/opt/nvidia/nvidia_…"   19 months ago  Exited (0) 4 weeks ago                                                                   pyt2
ajay@ubuntu:/$
```

Start the container cmd: docker start srg2ir

Attach the container cmd: docker attach srg2ir

Step-3: Launched the jupyter-lab in the container

Cmd: jupyter-lab –ip=0.0.0.0 –port=8157 –no-browser –NotebookApp.token='Token' –allow-root

Command for Inference:

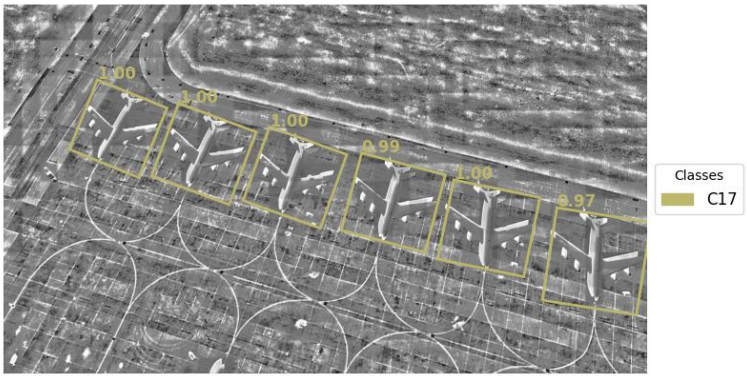
```
python3 inference_batch.py --input_folder ./dummy_input --   output_folder ./dummy_latest --
checkpoint ./translation_weights.pt --a2b 0 --seed 1234 --num_style 5 --synchronized --output_only
--config configs/tir2rgb_folder.yaml
```

Table:

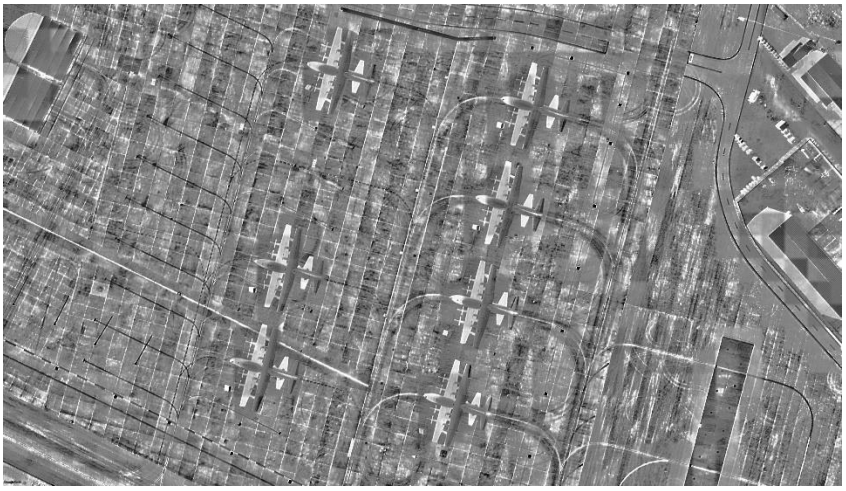
Method: Multi-Scale (MS)	ATD – (Automatic Target Detection) (IR-Image) (Hyper parameters)									
Model	Backbone	Dataset	Img_count	Categories	Img_size	Batch_size	Epochs	Optimizer	LR	mAP(%)
REDET	RESNET-50	Google Earth	2337	8	1024	8	12	SGD	0.01	50.1

Inference:

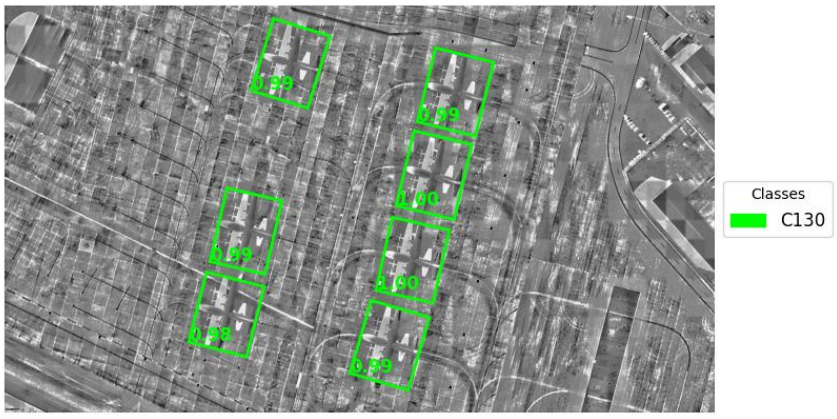
ORIGINAL IMAGE



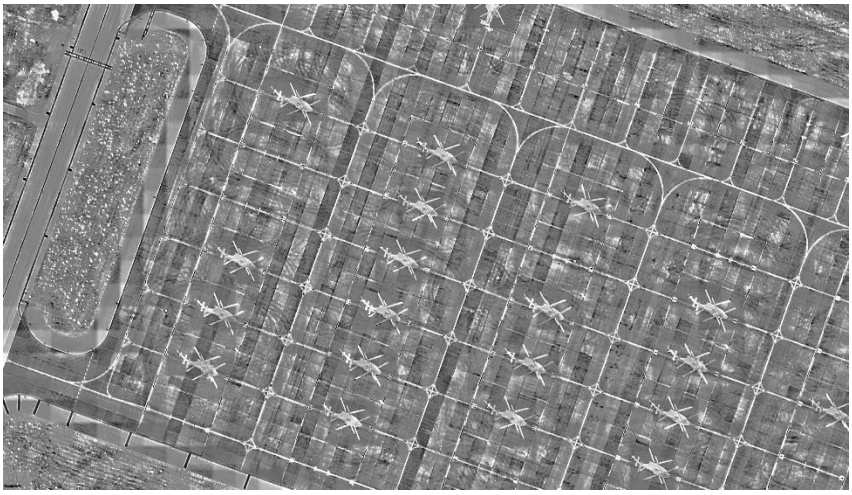
REDET



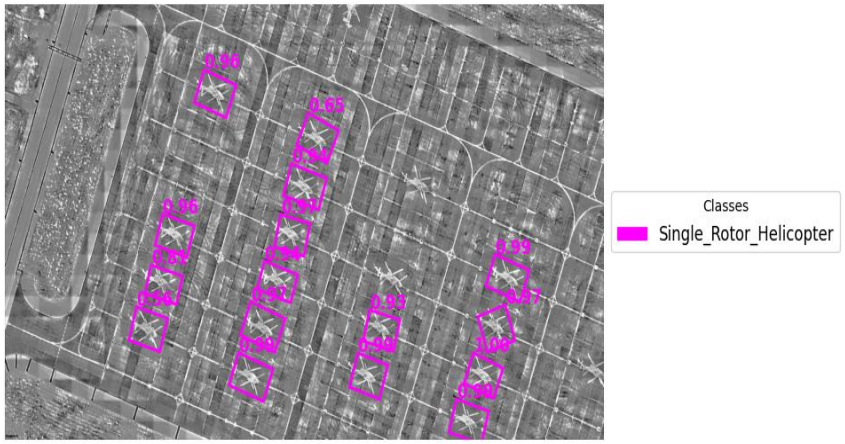
ORIGINAL IMAGE



REDET

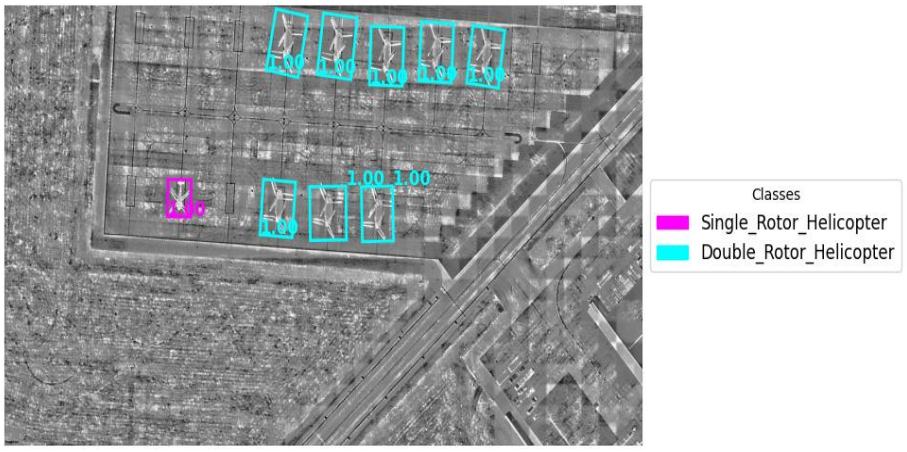


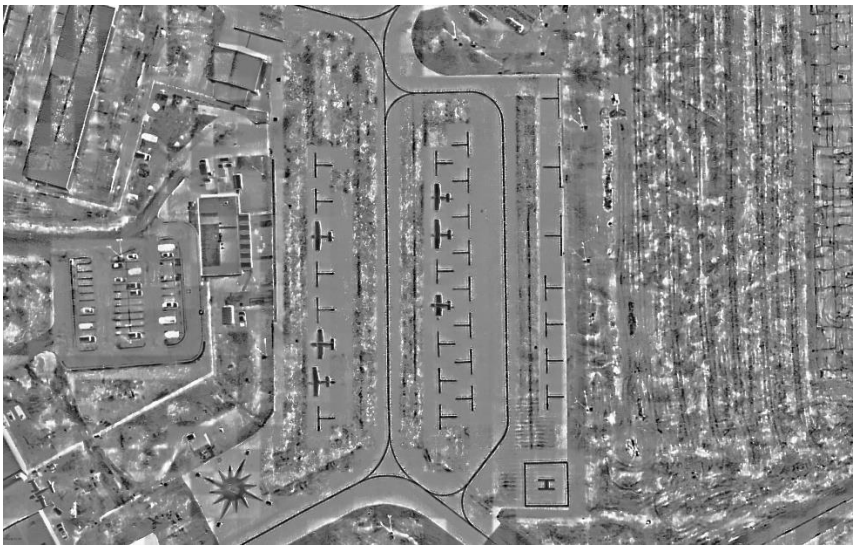
ORIGINAL IMAGE



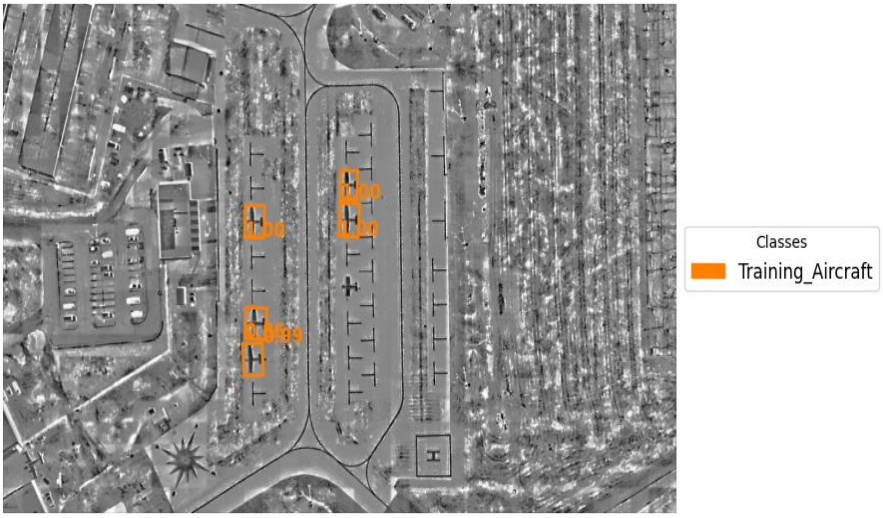
REDET

ORIGINAL IMAGE





ORIGINAL IMAGE



REDET

Synthetic Aperture Radar (SAR – Image)

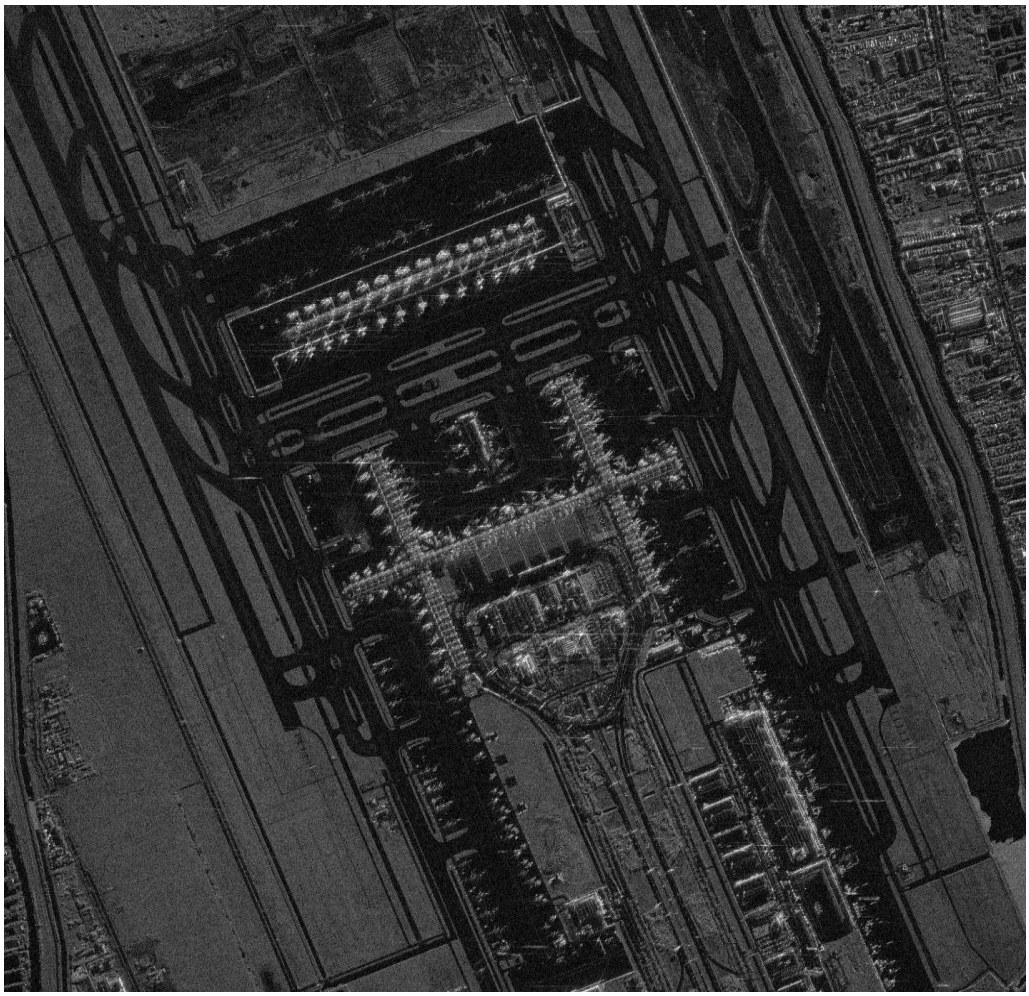
ATD – (Automatic Target Detection)

1. Infrastructure:
    - Containerization: Docker
    - Image Name: sarobjdet/iisc:version1
    - Container Name: sarobjdet
    - Container ID: e8234cc236db
    - Environment Name: MSFA
    - URL: 192.168.0.129:8888
    - Password: 123
    - GPU: NVIDIA DGX A100

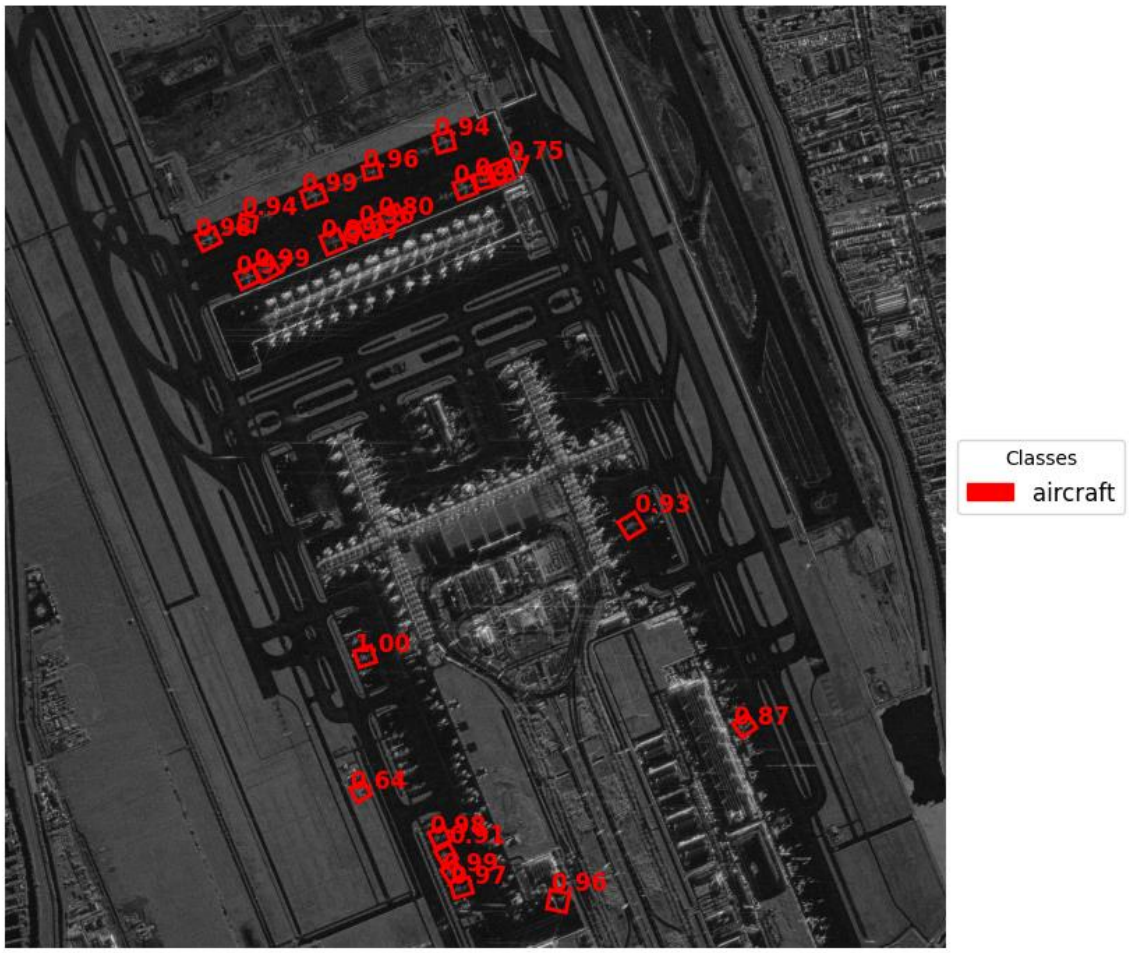
Table:

Method: Multi-Scale (MS)	ATD – (Automatic Target Detection) (SAR-Image) (Hyper parameters)									
Model	Backbone	Dataset	Img_count	Categories	Img_size	Batch_size	Epochs	Optimizer	LR	mAP(%)
REDET	RESNET-50	UMBRA	12	1	1024	8	12	SGD	0.01	51.9

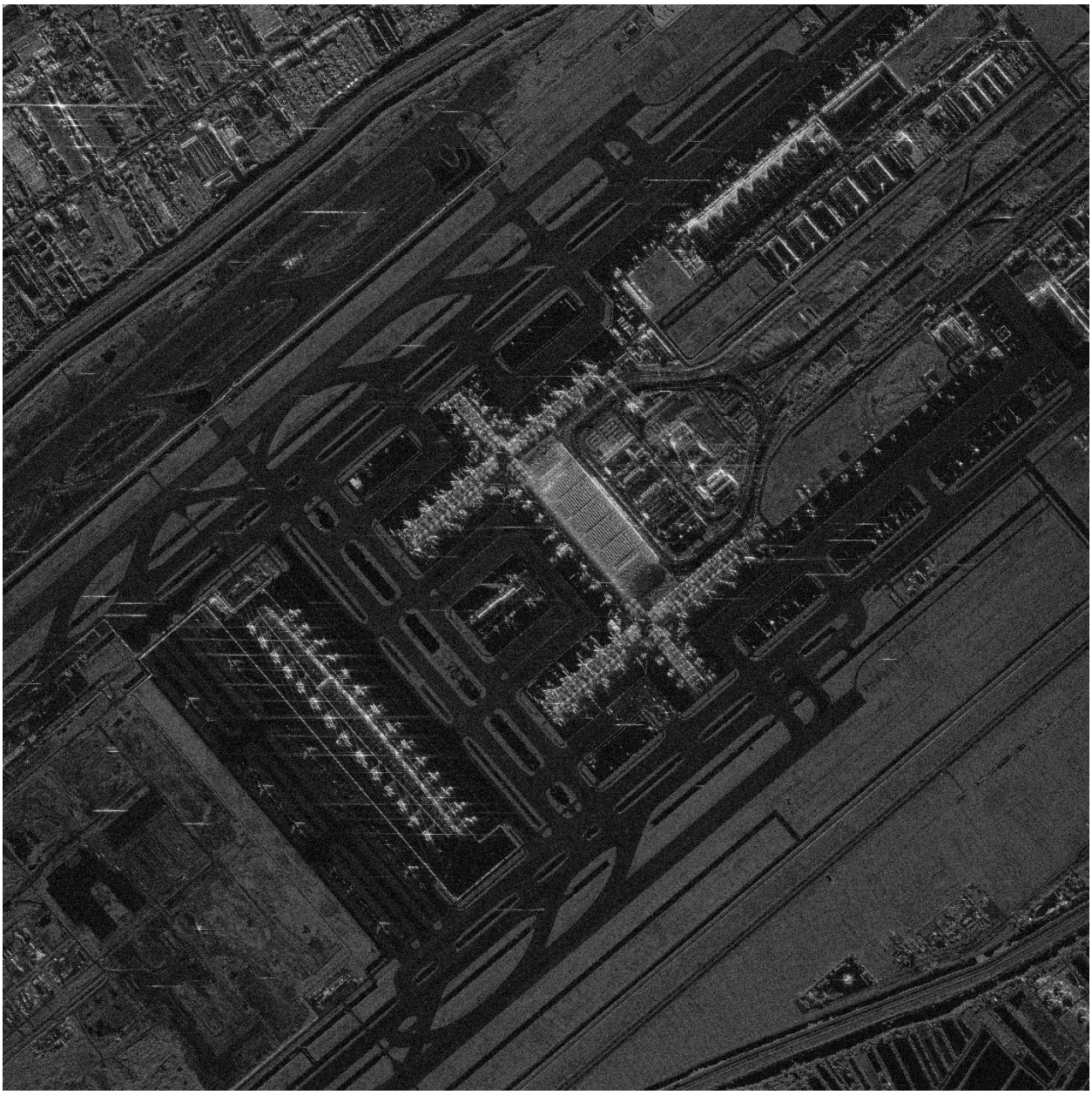
Inference:



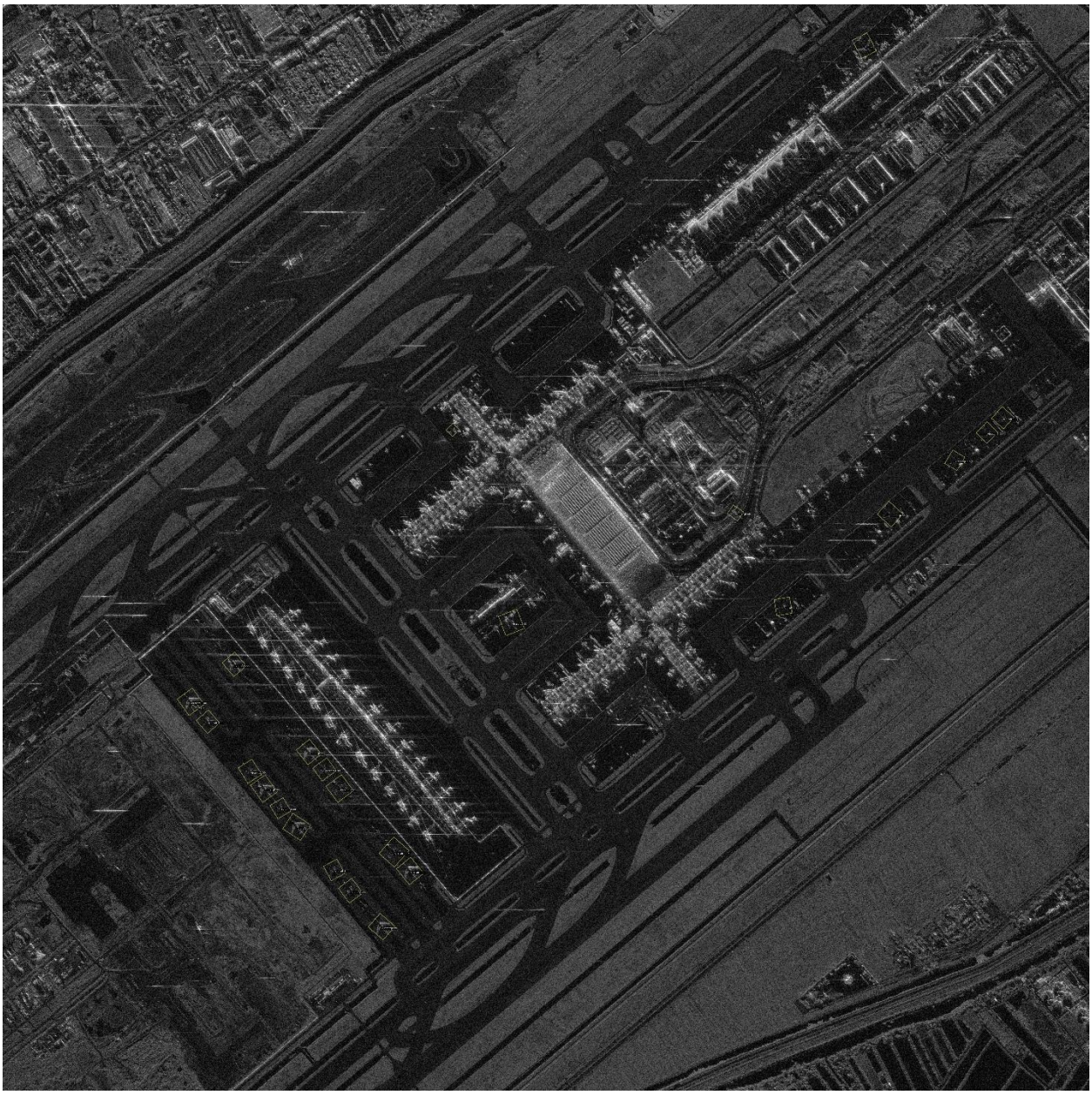
ORIGINAL IMAGE



REDET



ORIGINAL IMAGE



REDET