

## Model Development Phase Template

Date	8 November 2024
Team ID	team - 739994
Project Title	Virtual Eye - Life Guard for Swimming Pools to Detect Active Drowning
Maximum Marks	10 Marks

### Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for the model, presented through respective screenshots.

### Initial Model Training Code (5 marks):

```

[8] yolov5n.pt          100%[=====] 3.77M --.-KB/s  in 0.06s
2024-11-05 12:31:52 (64.5 MB/s) - 'yolov5n.pt' saved [3952441/3952441]

from ultralytics import YOLO

# Load the model using the correct filename
model = YOLO("yolov5n.pt") # Ensure this file is in the current directory

# Step 3: Verify successful loading
print("Model loaded successfully.")
# Train the model
results = model.train(data="/content/swimming-and-drowning-dataset/data.yaml", epochs=70, imgsz=640)

Class      Images  Instances  Box(P   R   mAP50  mAP50-95): 100%|██████████| 62/62 [00:17<00:00, 3.51it/s]

Epoch      GPU_mem  box_loss  cls_loss  dfl_loss  Instances  Size
63/70      2.22G      1.284    0.8229    1.397      25         640: 100%|██████████| 376/376 [01:50<00:00, 3.42it/s]
Class      Images  Instances  Box(P   R   mAP50  mAP50-95): 100%|██████████| 62/62 [00:17<00:00, 3.52it/s]

```

### Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics

## YOLO (v5)

YOLOv5 is a fast and accurate object detection model that identifies and locates objects in images in real-time. It predicts both bounding boxes and class labels in a single pass, making it efficient for tasks like surveillance and drowning detection.

```
[8] yolo5n.pt 100%[=====] 3.77M --KB/s in 0.06s
2024-11-05 12:31:52 (64.5 MB/s) - 'yolo5n.pt' saved [3952441/3952441]
```

```
from ultralytics import YOLO

# Load the model using the correct filename
model = YOLO('yolo5n.pt') # Ensure this file is in the current directory

# Step 3: Verify successful loading
print('Model loaded successfully.')
# Train the model
results = model.train(data='content/swimming-and-drowning-dataset/data.yaml', epochs=70, imgsz=640)
```

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
Drowning	1857	2130	0.856	0.847	0.901	0.547
Swimming	555	1277	0.806	0.838	0.85	0.471
out of water	88	117	0.778	0.658	0.762	0.448

```
70 epochs completed in 2.600 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 5.3MB
Optimizer stripped from runs/detect/train/weights/best.pt, 5.3MB

Validating runs/detect/train/weights/best.pt...
Ultralytics 8.3.17 'Python-3.10.12 torch-2.5.0+cu121 CUDA:0 (Tesla T4, 15102MB)'
YOLOv5n summary (Fused): 193 layers, 2,503,520 parameters, 0 gradients, 7.1 GFLOPs
Class Images Instances Box(P) R mAP50 mAP50-95: 100% 62/62 [00:20:00:00, 2.981it/s]
Drowning 1857 2130 0.856 0.847 0.901 0.547
Swimming 555 1277 0.806 0.838 0.85 0.471
out of water 88 117 0.778 0.658 0.762 0.448
Speed: 0.3ms preprocess, 2.2ms inference, 0.0ms loss, 2.1ms postprocess per image
Results saved to runs/detect/train
```

```
[12] model.val(data = 'content/swimming-and-drowning-dataset/data.yaml')
0.64565, 0.64665, 0.64765, 0.64865, 0.64965, 0.65065, 0.65165, 0.65265, 0.65365, 0.65465, 0.65566, 0.65666, 0.65766,
0.64865, 0.64965, 0.65065, 0.65165, 0.65265, 0.65365, 0.65465, 0.65566, 0.65666, 0.65766,
```

```
names: {'0': 'Drowning', '1': 'Swimming', '2': 'out of water'}
[12] plot: True
results_dict: {'metrics/precision@0': 0.8123269156632354, 'metrics/recall@0': 0.783598454581999, 'metrics/mAP50@0': 0.8368440733851998,
'metrics/mAP50-95@0': 0.4881379161239458, 'fitness': 0.5230885318508712}
save_dir: PosixPath('runs/detect/val2')
speed: {'preprocess': 0.3893097831398333, 'inference': 4.432785597981215, 'loss': 0.0013969857462379889, 'postprocess': 1.486679088649827}
task: 'detect'
```

```
from ultralytics import YOLO

# Load the model using the correct filename
model = YOLO('content/runs/detect/train/weights/best.pt') # Ensure this file is in the current directory

model.predict('content/test/images/-Clipchamp_mp4-11.jpg.rf.0c2b48fc08dc1741bade48d49546814.jpg', save=True, imgsz=320, conf=0.5)
```

```
keypoints: None
masks: None
names: {'0': 'Drowning', '1': 'Swimming', '2': 'out of water'}
obb: None
orig_img: array([[[[106, 105, 107],
[105, 104, 106],
[103, 102, 104],
...,
[157, 149, 112],
[155, 147, 110],
[169, 161, 94],
[169, 161, 94],
[169, 161, 94]],
[[166, 159, 90],
[166, 159, 90],
[166, 159, 90],
...,
[169, 161, 94],
[169, 161, 94],
[169, 161, 94],
[169, 161, 94]],
[[166, 159, 90],
[166, 159, 90],
[166, 159, 90],
...,
[169, 161, 94],
[169, 161, 94],
[169, 161, 94],
[169, 161, 94]]], dtype=uint8)
orig_shape: (640, 640)
path: 'content/test/images/-Clipchamp_mp4-11.jpg.rf.0c2b48fc08dc1741bade48d49546814.jpg'
probs: None
save_dir: 'runs/detect/predict'
speed: {'preprocess': 1.3511188877685547, 'inference': 17.465114593598586, 'postprocess': 4.098176956176758}
```