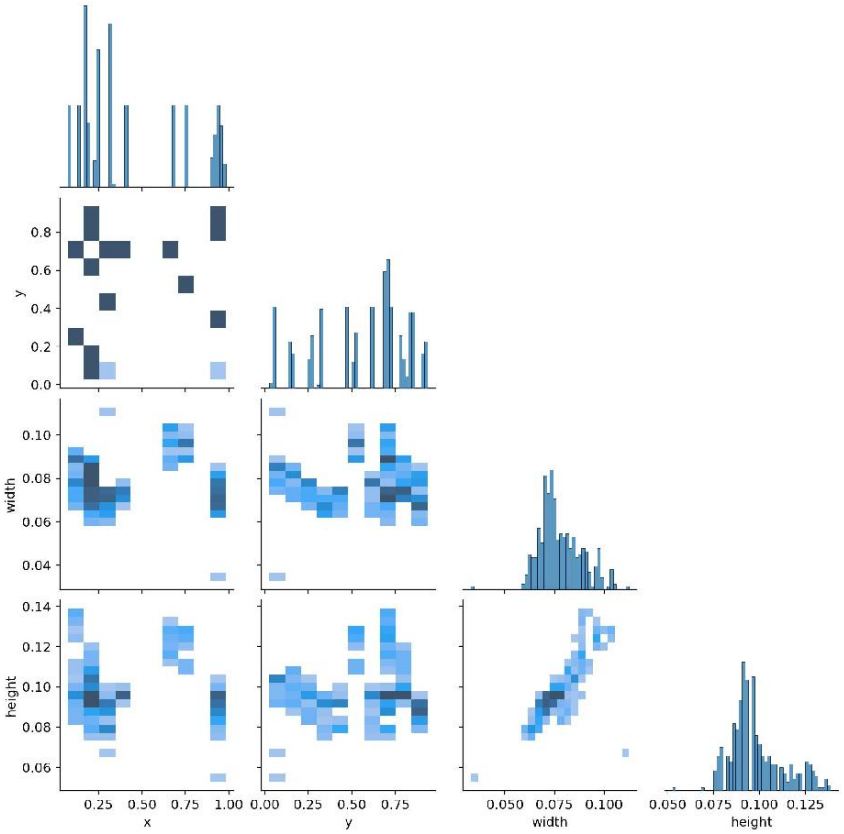
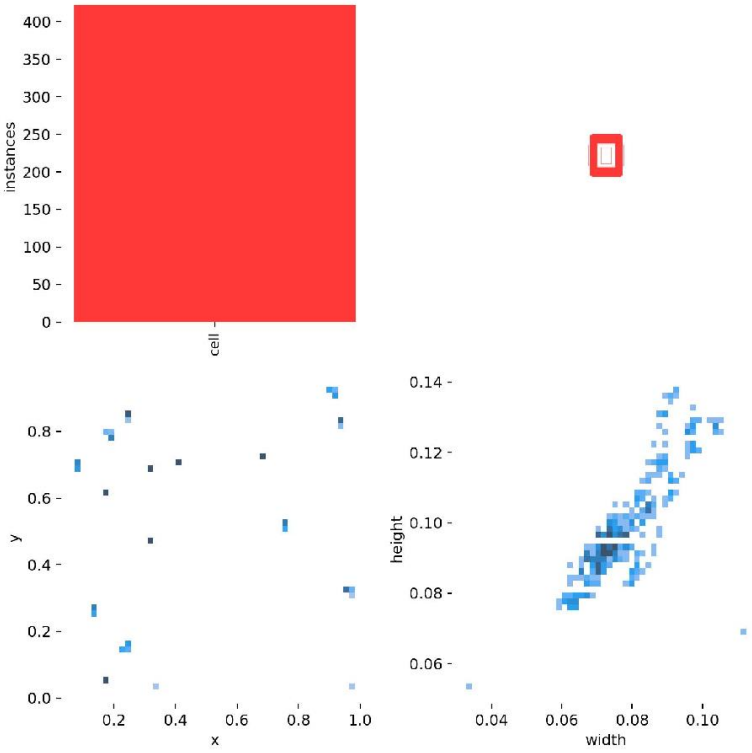
ДОДАТОК А

Графіки результатів навчання моделі YOLO

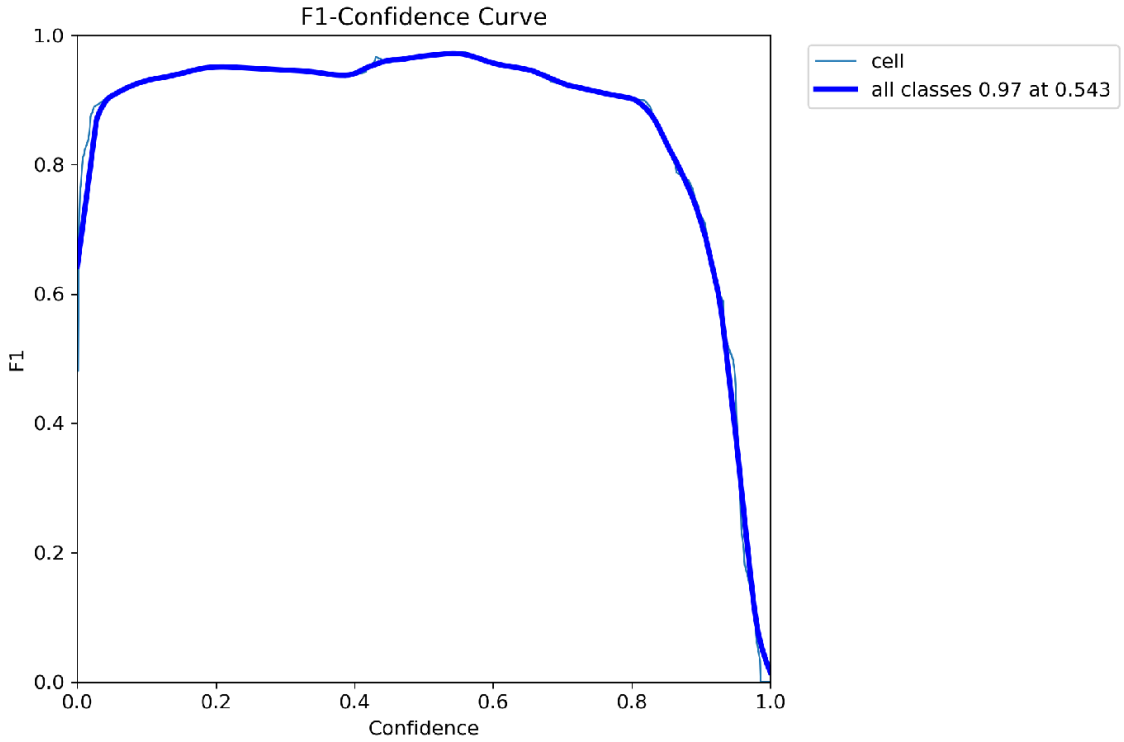
Labels correlogram



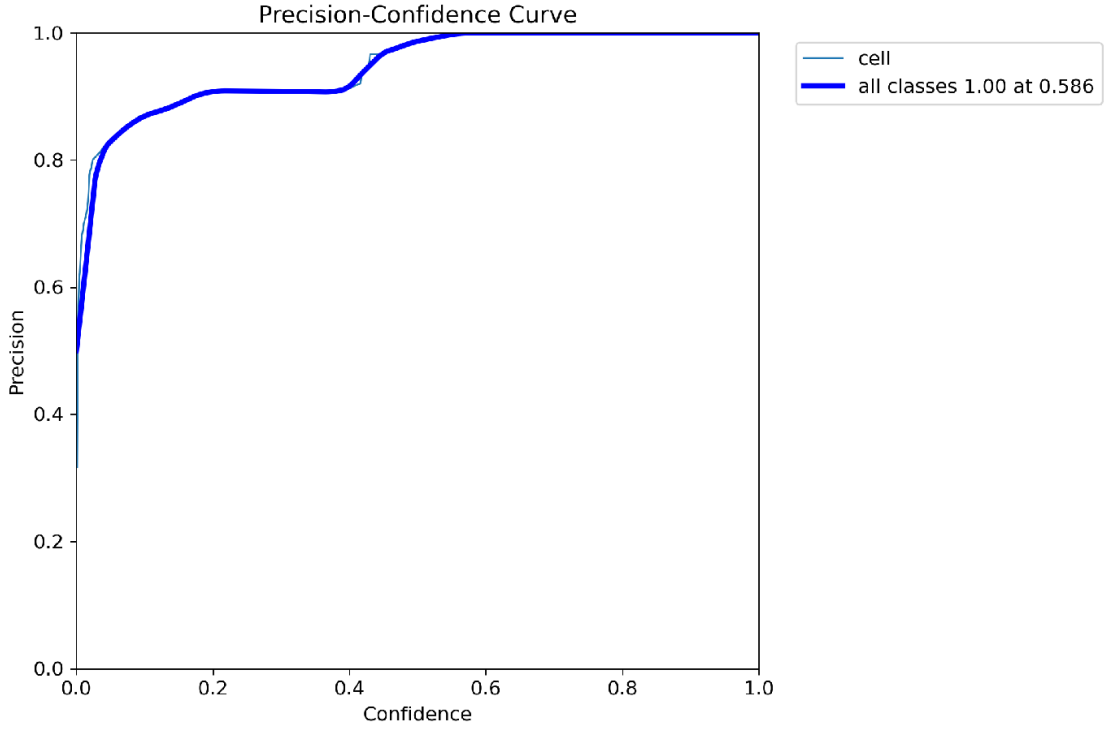
Labels



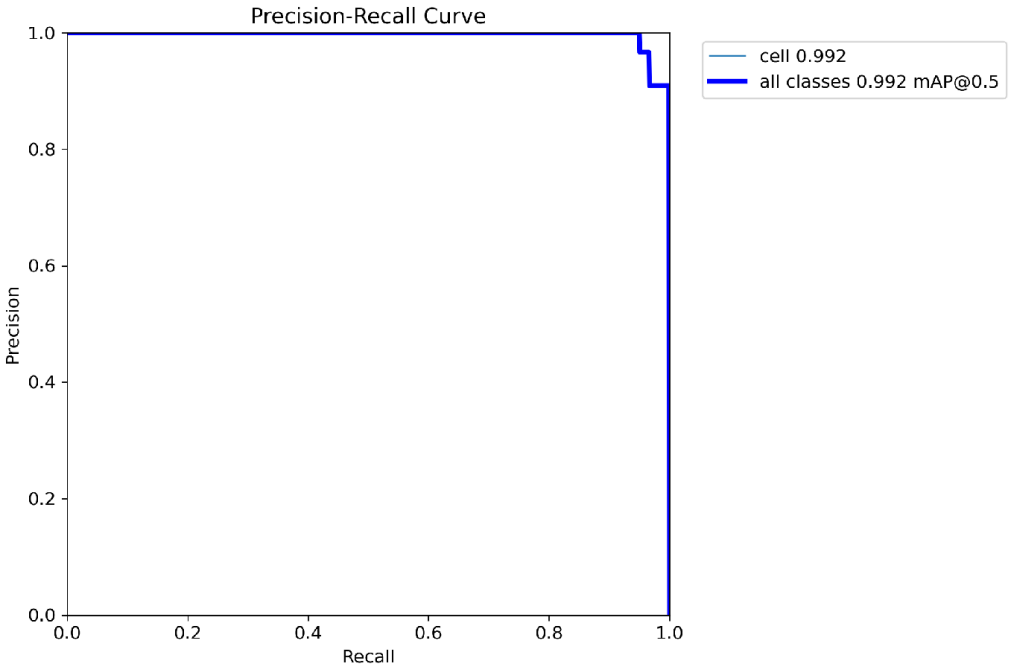
Mask F1



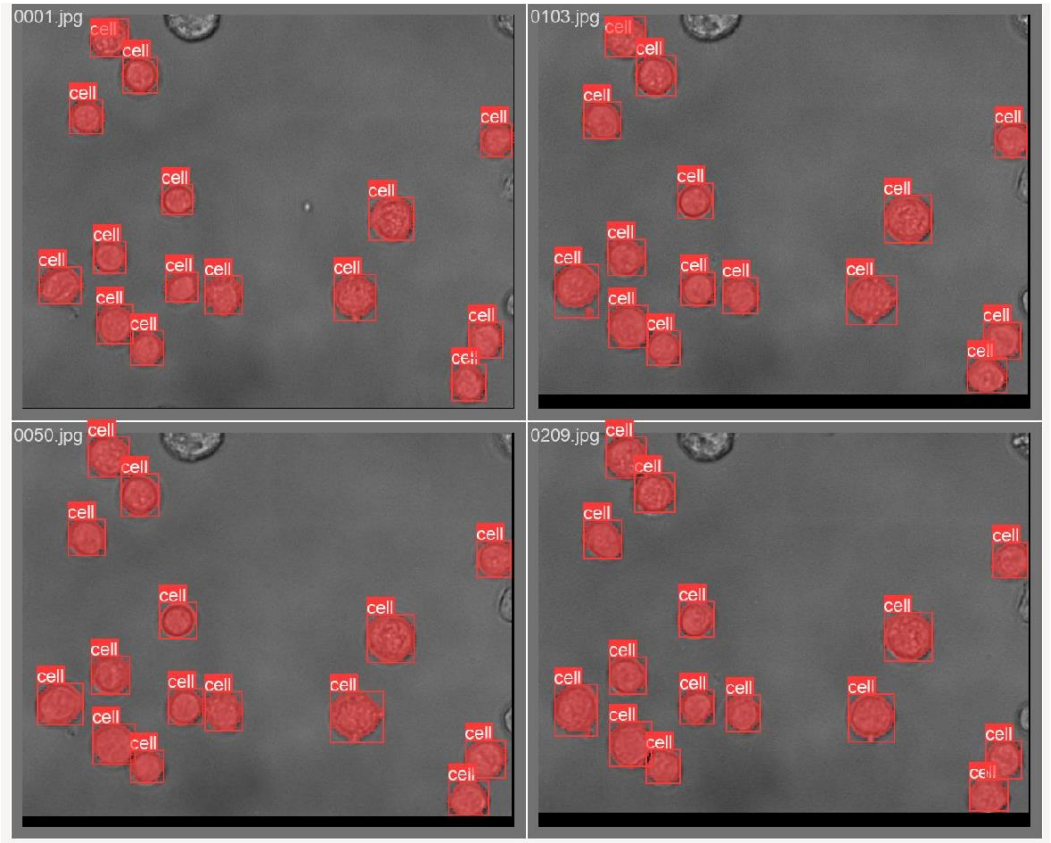
Mask P



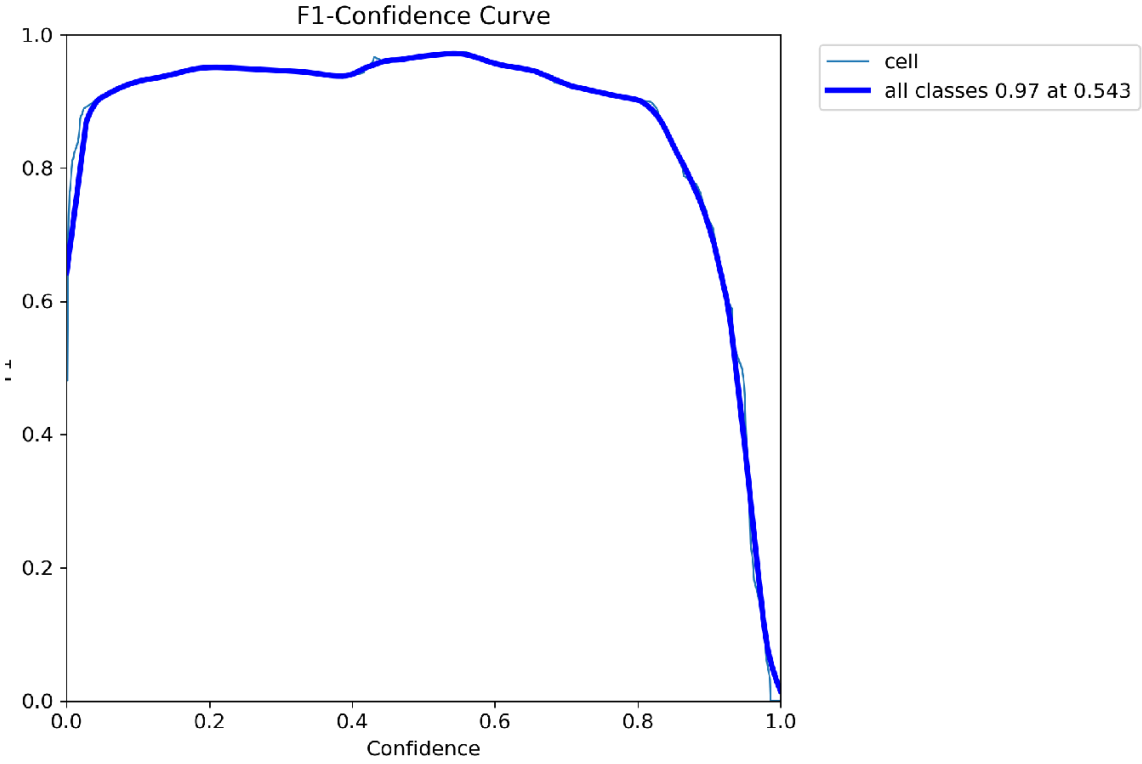
Maskr PR



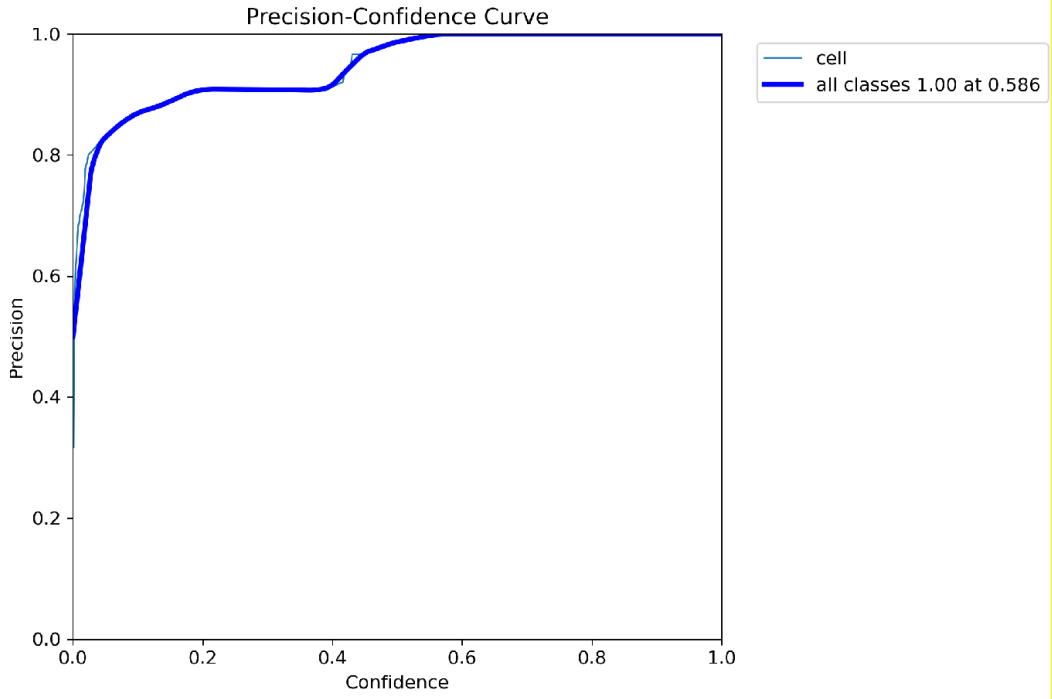
Validation batch



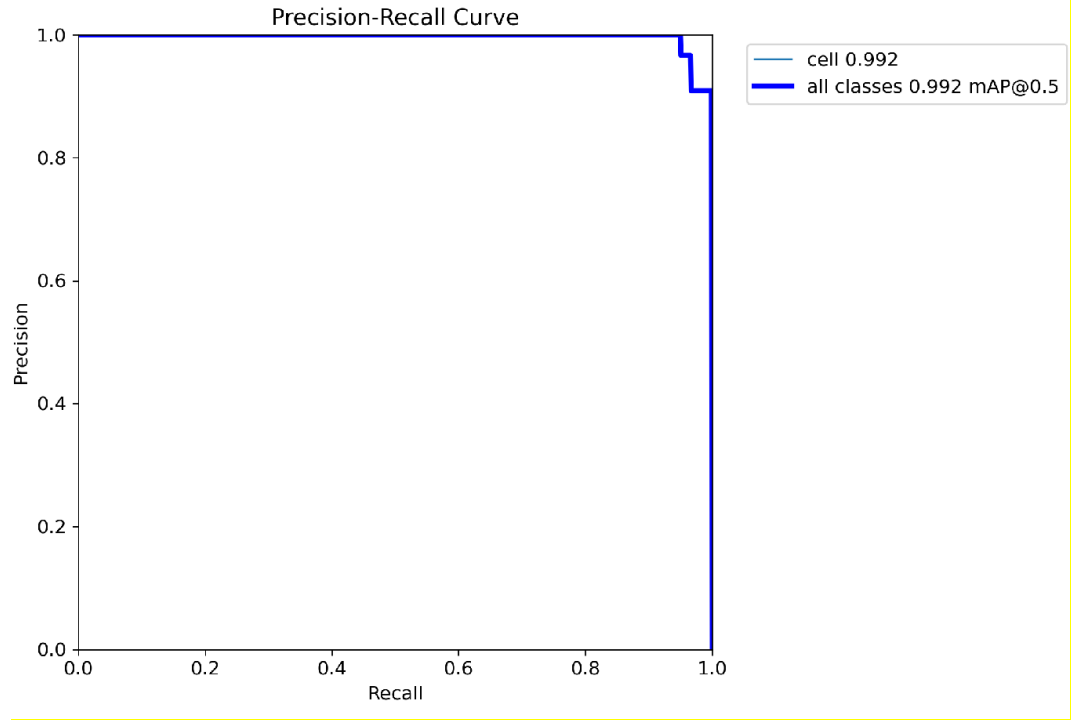
Box F1


F1-Confidence Curve
cell
all classes 0.97 at 0.543
F1
Confidence
0.0
0.2
0.4
0.6
0.8
1.0

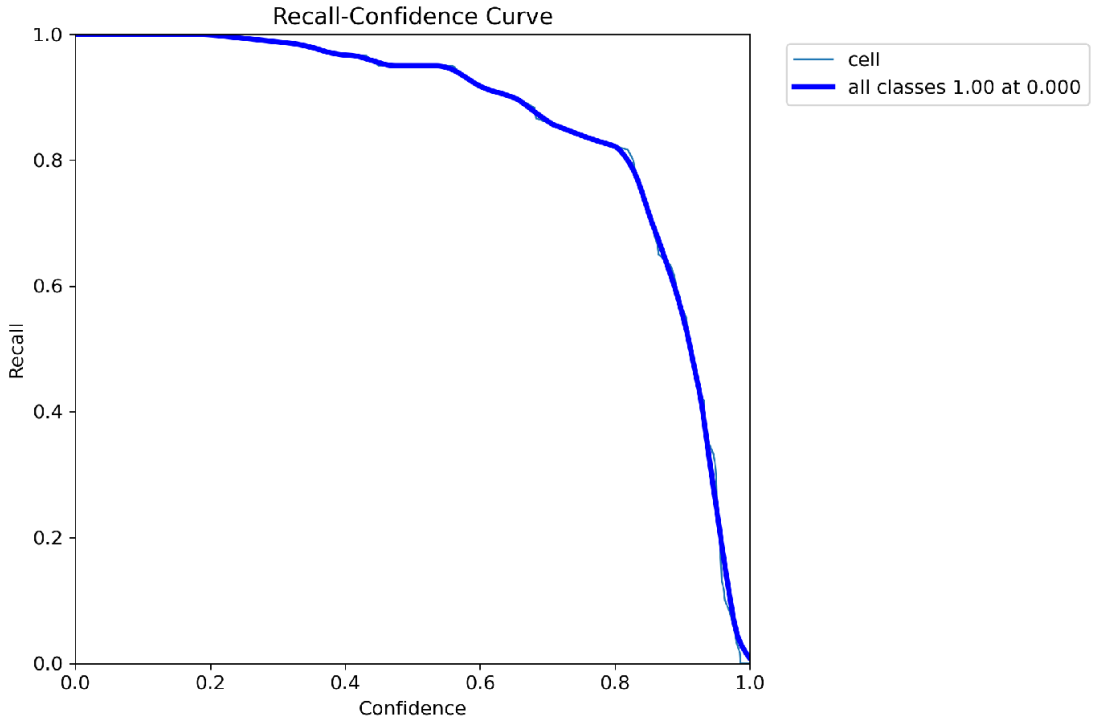
Box P


Precision-Confidence Curve
cell
all classes 1.00 at 0.586
Precision
Confidence
0.0
0.2
0.4
0.6
0.8
1.0

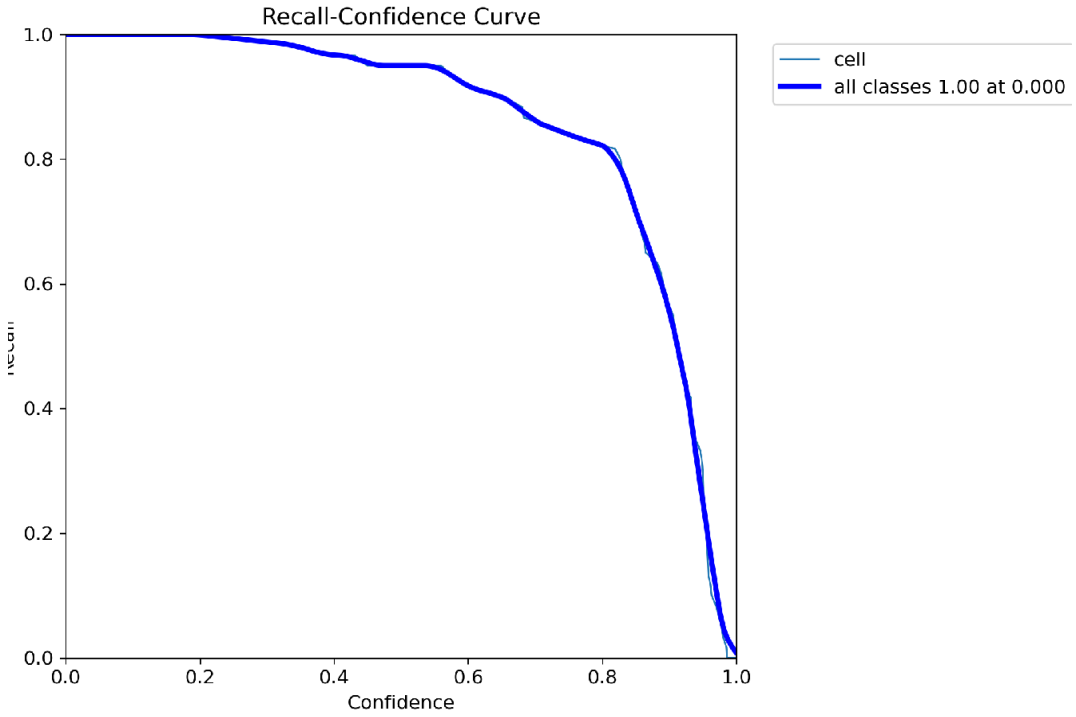
Box PR


Precision-Recall Curve
cell 0.992
all classes 0.992 mAP@0.5
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

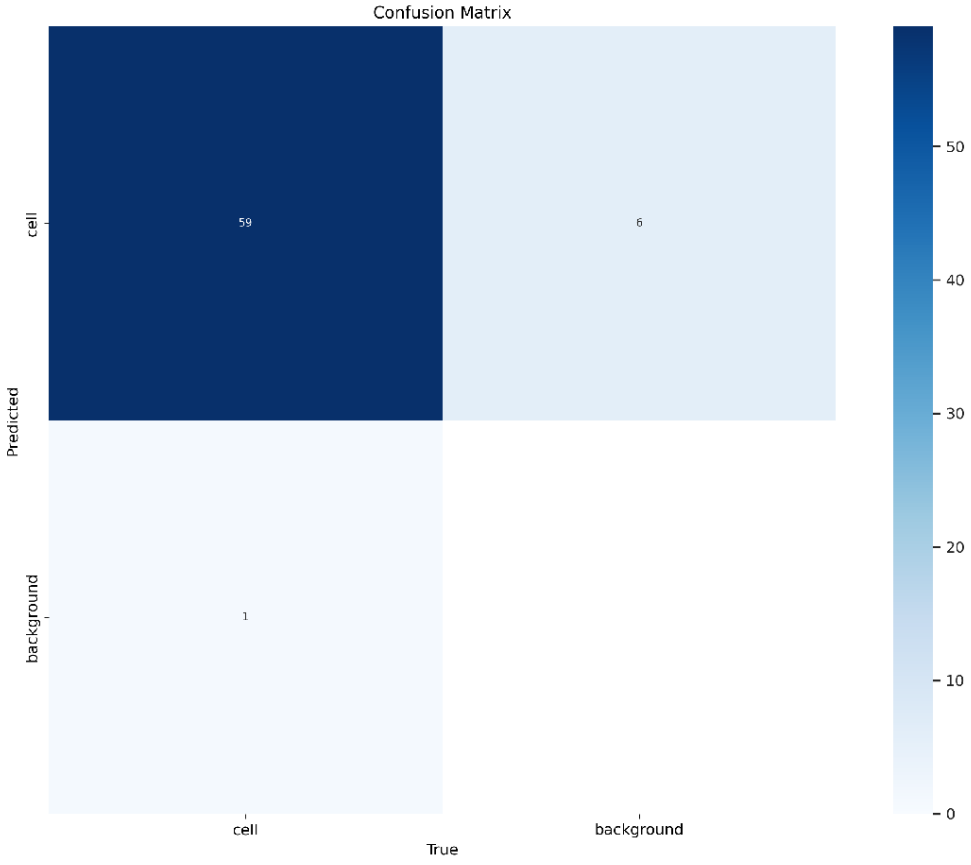
Mask R


Recall-Confidence Curve
cell
all classes 1.00 at 0.000
Recall
Confidence
0.0
0.2
0.4
0.6
0.8
1.0

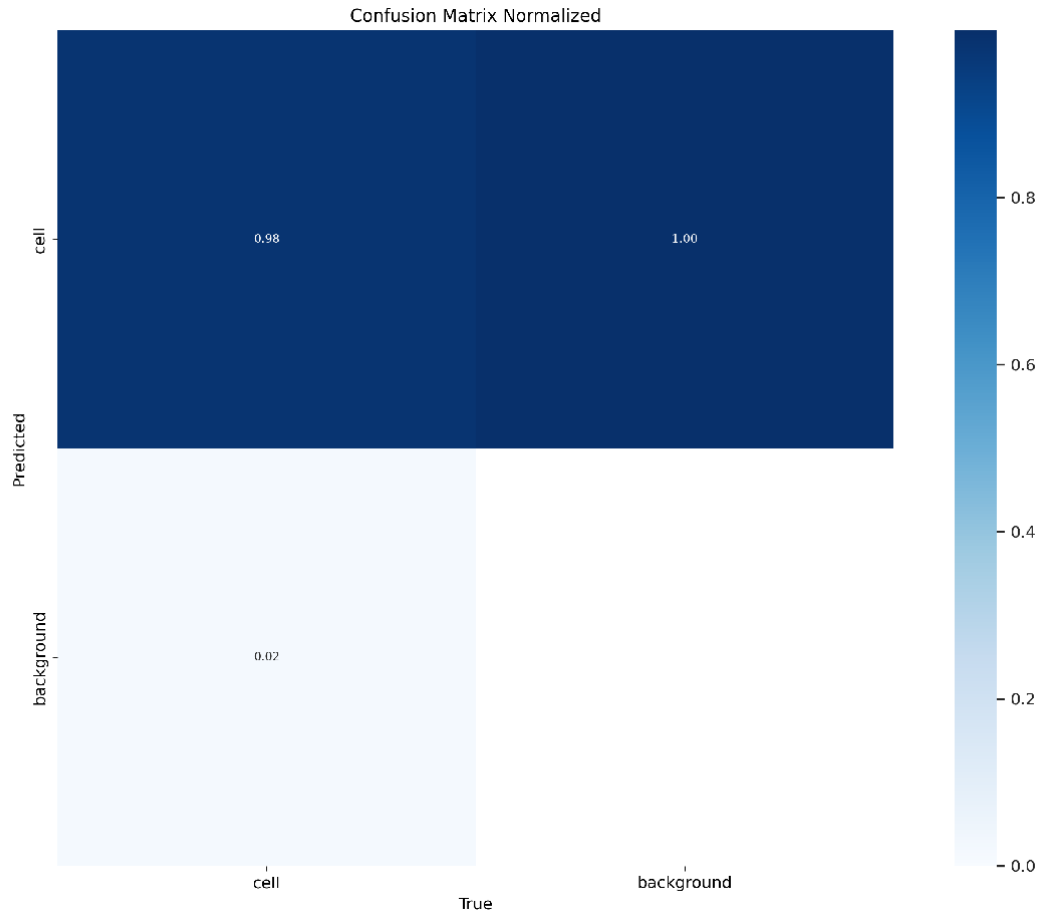
Box R



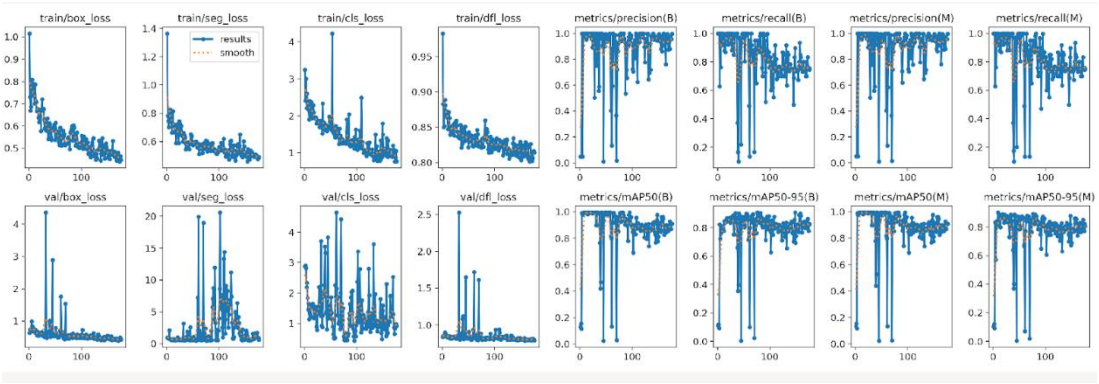
Confusion matrix



Confusion matrix normalized



Results



ДОДАТОК Б

Код програми

Файл train_model.py

```
from ultralytics import YOLO
# Load a model
model = YOLO('yolov8n-seg.pt')  # load a pretrained model (recommended for training)

# Train the model
results = model.train(data='fine-tune.yaml', epochs=10, imgsz=640, name="yolov8n-seg-cells")

print(results)
# Alternatively, it can be run using console:
#   yolo   task=segment   mode=train   model=yolov8n-seg.pt imgsz=640 data=fine-tune.yaml epochs=750 name=yolov8n-seg-cells
```

Файл main.py

```
import os

import cv2
from ultralytics import YOLO
from ultralytics.utils.plotting import Annotator, colors

from collections import defaultdict

VIDEO_FILE = "cells.mp4"
IMAGES_DIR = "krio"
IMAGE_WIDTH = 640
IMAGE_HEIGHT = 512
FPS = 25

def convert_images_to_video(source_dir, output_file, video_dim, fps):
    video_writer = cv2.VideoWriter(output_file, cv2.VideoWriter_fourcc(*"mp4v"), fps, video_dim)
    images = os.listdir(f"{source_dir}")

    images.sort()
    for img_name in images:
        img = cv2.imread(os.path.join(source_dir, img_name))
        video_writer.write(img)

    video_writer.release()

if not os.path.isfile(VIDEO_FILE):
    convert_images_to_video(IMAGES_DIR, VIDEO_FILE, (IMAGE_WIDTH, IMAGE_HEIGHT), FPS)
```

```
    track_history = defaultdict(lambda: [])

    model = YOLO("yolo8n-seg-cells.pt")
    cap = cv2.VideoCapture(VIDEO_FILE)
    w,    h,    fps    =    (int(cap.get(x))    for    x    in
(cv2.CAP_PROP_FRAME_WIDTH,           cv2.CAP_PROP_FRAME_HEIGHT,
cv2.CAP_PROP_FPS))

    out    =    cv2.VideoWriter('instance-segmentation-object-
tracking.avi', cv2.VideoWriter_fourcc(*'MJPG'), fps, (w, h))

    while True:
        ret, im0 = cap.read()
        if not ret:
            print("Video frame is empty or video processing has
been successfully completed.")
            break

        annotator = Annotator(im0, line_width=2)

        results = model.track(im0, persist=True)

        if results[0].boxes.id is not None and results[0].masks
is not None:
            masks = results[0].masks.xy
            track_ids                                          =
results[0].boxes.id.int().cpu().tolist()

            for mask, track_id in zip(masks, track_ids):
                annotator.seg_bbox(mask=mask,
                                   mask_color=colors(track_id
, True),
                                   track_label=str(track_id))

        out.write(im0)
        cv2.imshow("instance-segmentation-object-tracking",
im0)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    out.release()
    cap.release()
    cv2.destroyAllWindows()

    Файл masks_to_polygons.py
    import os

    import cv2


    input_dir = './masks'
```

```
    output_dir = './labels'

    for j in os.listdir(input_dir):
        image_path = os.path.join(input_dir, j)
        # Load image masks and get their borders
        mask = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
        _,    mask    =    cv2.threshold(mask,    1,    255,
cv2.THRESH_BINARY)

        H, W = mask.shape
        contours,    hierarchy    =    cv2.findContours(mask,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

        # Convert borders to polygons
        polygons = []
        for cnt in contours:
            if cv2.contourArea(cnt) > 200:
                polygon = []
                for point in cnt:
                    x, y = point[0]
                    polygon.append(x / W)
                    polygon.append(y / H)
                polygons.append(polygon)

        # Save to files
        with     open('{}.txt'.format(os.path.join(output_dir,
j)[:-4]), 'w') as f:
            for polygon in polygons:
                for p_, p in enumerate(polygon):
                    if p_ == len(polygon) - 1:
                        f.write('{}\n'.format(p))
                    elif p_ == 0:
                        f.write('0 {} '.format(p))
                    else:
                        f.write('{} '.format(p))

            f.close()
```

Файл resize_img_dir.py

```
    import os

    import cv2

    SOURCE_DIR = "./jpg_images"
    TARGET_DIR = "./jpg_images_resized"
    IMG_WIDTH = 640
    IMG_HEIGHT = 512

    images_file_names = os.listdir(SOURCE_DIR)

    for img_file_name in images_file_names:
        img       =       cv2.imread(os.path.join(SOURCE_DIR,
img_file_name))
```

```
        img = cv2.resize(img, (IMG_WIDTH, IMG_HEIGHT))
        cv2.imwrite(os.path.join(TARGET_DIR,   img_file_name),
img)
```

Файл fine-tune.yaml

```
path: ../datasets/cells-seg
train: images/train
val: images/val

# Classes
names:
  0: cell
```

ДОДАТОК В

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	87 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата			
Розробив		Купріянов С.В.			Інтелектуальний аналіз трекінгу клітин при кріоконсервації	Шифр групи	Код напр./спец.
Перевірив		Аврунін О.Г.				СШІм-22-2	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					