

▼ FridgeEye — Entrenamiento Oficial v1

Pipeline: pseudo-labeling con modelo propio → split 80/20 → entrenamiento YOLOv8n → TFLite

```
1 !pip install -q ultralytics
2
```

1.3/1.3 MB 21.8 MB/s eta 0:00:00

```
1 from google.colab import drive
2 drive.mount('/content/drive')
3
```

Mounted at /content/drive

```
1 import os, json, shutil, random, yaml, warnings, collections
2 from pathlib import Path
3 from datetime import datetime
4 import numpy as np
5 import torch
6 from ultralytics import YOLO
7 warnings.filterwarnings('ignore')
8
9 device = '0' if torch.cuda.is_available() else 'cpu'
10 gpu_name = torch.cuda.get_device_name(0) if device == '0' else 'CPU'
11 print(f'Device: {gpu_name}')
12
```

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>

Device: Tesla T4

```
1 # — Rutas —————
2 DRIVE_BASE = '/content/drive/MyDrive/EAN/Proyecto_de_grado'
3 FRESH_SRC = f'{DRIVE_BASE}/dataZips/DataSetFull/Fresh'
4 ROTTEN_SRC = f'{DRIVE_BASE}/dataZips/DataSetFull/Rotten'
5 OUTPUT_BASE = f'{DRIVE_BASE}/Modelos'
6
7 # Mejor modelo del benchmarking (pseudo-labeler)
8 PSEUDO_LABELER_PT = f'{OUTPUT_BASE}/pt/fridgeeye_oficial_20260422_1316/weights/best.pt'
9
10 # Todo en Drive — persistente entre sesiones
11 WORK_DIR = Path(f'{DRIVE_BASE}/TrainingData')
12 IMGS_RAW = WORK_DIR / 'images_raw'
13 LBLS_RAW = WORK_DIR / 'labels_raw'
14 DATASET_DIR = WORK_DIR / 'dataset'
15 CHECKPOINT_F = WORK_DIR / 'pseudo_progress.json'
16
17 MODELS_PT = f'{OUTPUT_BASE}/pt'
18 MODELS_TFLITE = f'{OUTPUT_BASE}/tflite'
19 REPORTS_DIR = f'{OUTPUT_BASE}/reports'
20
21 # — Pseudo-labeling —————
22 NC = 2
23 NAMES = ['Fresco', 'Deteriorado']
24 IMGSZ = 640
25 CONF_THRESH = 0.40 # confianza mínima para aceptar detección
26 MAX_IMAGES_PER_CLASS = 99_000
27 VAL_RATIO = 0.20
28 SEED = 42
29
30 # — Entrenamiento (mejor config del benchmarking) —————
31 LR0 = 0.001
32 BATCH = 32
33 EPOCHS = 120
34 PATIENCE = 25
35
36 RUN_ID = datetime.now().strftime('%Y%m%d_%H%M')
37 # RUN_NAME = f'fridgeeye_oficial_{RUN_ID}'
38 RUN_NAME = 'fridgeeye_oficial_20260429_1412'
39
40 for d in [WORK_DIR, IMGS_RAW, LBLS_RAW]:
41     d.mkdir(parents=True, exist_ok=True)
42 for d in [MODELS_PT, MODELS_TFLITE, REPORTS_DIR]:
43     Path(d).mkdir(parents=True, exist_ok=True)
44
```

```

45 print(f'Run: {RUN_NAME}')
46 print(f'Pseudo-labeler: {PSEUDO_LABELER_PT}')
47 print(f'Datos en: {WORK_DIR}')
48

```

Run: fridgeeye_oficial_20260429_1412

Pseudo-labeler: /content/drive/MyDrive/EAN/Proyecto_de_grado/Modelos/pt/fridgeeye_oficial_20260422_1316/weights/best.pt

Datos en: /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData

✓ 1. Pseudo-labeling con modelo propio

Usa el mejor modelo del benchmarking (F1=0.679) para generar bounding boxes. Solo se aceptan detecciones con confianza $\geq$ CONF_THRESH (0.40 por defecto). La celda es **reanudable** si el runtime se reinicia.

```

1 TMP_DIR = Path('/content/tmp_batch') # temporal local, se vacía después de cada batch
2
3 def pseudo_label_folder(model, src_folder, class_id, imgs_dir, lbls_dir,
4                           conf_thresh=0.40, max_images=15_000,
5                           done_stems=None, batch_size=500, save_fn=None):
6     imgs_dir = Path(imgs_dir)
7     lbls_dir = Path(lbls_dir)
8     exts = ['.jpg', '.jpeg', '.png', '.webp', '.bmp']
9
10    # os.walk — fiable en Drive montado
11    all_imgs = []
12    for root, _, files in os.walk(str(src_folder)):
13        for fname in files:
14            if Path(fname).suffix.lower() in exts:
15                all_imgs.append(Path(root) / fname)
16
17    random.seed(SEED)
18    random.shuffle(all_imgs)
19    all_imgs = all_imgs[:max_images]
20
21    done_set = set(done_stems or [])
22    accepted = list(done_stems or [])
23    skipped = 0
24
25    # Solo las imágenes que aún no están en el checkpoint
26    pending = []
27    for i, img_path in enumerate(all_imgs):
28        stem = f'c{class_id}_{i:06d}_{img_path.stem[:28]}'
29        if stem not in done_set:
30            pending.append((stem, img_path))
31
32    print(f' Total encontradas : {len(all_imgs)}')
33    print(f' Pendientes      : {len(pending)} | Ya procesadas: {len(done_set)}')
34
35    for b_start in range(0, len(pending), batch_size):
36        batch = pending[b_start:b_start + batch_size]
37
38        # 1. Copiar batch a disco local (rápido para GPU)
39        TMP_DIR.mkdir(parents=True, exist_ok=True)
40        tmp_paths = []
41        for stem, img_path in batch:
42            tmp_dst = TMP_DIR / f'{stem}{img_path.suffix.lower()}'
43            shutil.copy2(img_path, tmp_dst)
44            tmp_paths.append(str(tmp_dst))
45
46        # 2. Inferencia en batch (todas las imágenes de una sola pasada por GPU)
47        try:
48            results = model.predict(tmp_paths, conf=conf_thresh,
49                                   verbose=False, device=device, stream=False)
50        except Exception as e:
51            print(f' Error en batch {b_start//batch_size + 1}: {e}')
52            shutil.rmtree(TMP_DIR, ignore_errors=True)
53            skipped += len(batch)
54            continue
55
56        # 3. Procesar resultados (mismo orden que tmp_paths)
57        batch_acc = 0
58        for (stem, img_path), res in zip(batch, results):
59            boxes = res.bboxes
60            if boxes is None or len(boxes) == 0:
61                skipped += 1
62                continue
63
64            best_idx = int(boxes.conf.argmax())
65            if boxes.conf[best_idx].item() < conf_thresh:

```

```

66         skipped += 1
67         continue
68
69         xywhn      = boxes.xywhn[best_idx].tolist()
70         label_line = f'{class_id} {xywhn[0]:.6f} {xywhn[1]:.6f} {xywhn[2]:.6f} {xywhn[3]:.6f}'
71
72         shutil.copy2(img_path, imgs_dir / f'{stem}{img_path.suffix.lower()}')
73         (lbls_dir / f'{stem}.txt').write_text(label_line)
74
75         accepted.append(stem)
76         done_set.add(stem)
77         batch_acc += 1
78
79         # 4. Limpiar tmp y guardar checkpoint en Drive
80         shutil.rmtree(TMP_DIR, ignore_errors=True)
81         if save_fn:
82             save_fn(accepted)
83
84         processed = b_start + len(batch)
85         pct = len(accepted) / (processed + 1e-9) * 100
86         print(f' Batch {b_start//batch_size + 1:>3} [{processed:>5}/{len(pending)}] '
87               f'+{batch_acc} aceptadas | total={len(accepted)} ({pct:.0f}%) | '
88               f'descartadas={skipped}')
89
90         shutil.rmtree(TMP_DIR, ignore_errors=True)
91         print(f'Clase {class_id} ({NAMES[class_id]}): {len(accepted)} aceptadas, '
92               f'{skipped} descartadas ({skipped/(len(all_imgs)+1e-9)*100:.1f}%)')
93         return accepted
94

```

```

1 # Carga checkpoint si el runtime se reinició
2 progress = {}
3 if CHECKPOINT_F.exists():
4     progress = json.loads(CHECKPOINT_F.read_text())
5     print(f'Reanudando – Fresh: {len(progress.get("fresh", []))}, '
6           f'Rotten: {len(progress.get("rotten", []))}')
7 else:
8     print('Iniciando pseudo-labeling desde cero')
9
10 pseudo_model = YOLO(PSEUDO_LABELER_PT)
11
12 # save_fn guarda checkpoint en Drive después de cada batch
13 def make_save_fn(key):
14     def _save(stems):
15         progress[key] = stems
16         CHECKPOINT_F.write_text(json.dumps(progress, indent=2))
17     return _save
18
19 print('\nPseudo-labeling Fresco (clase 0)...')
20 fresh_stems = pseudo_label_folder(
21     pseudo_model, FRESH_SRC, class_id=0,
22     imgs_dir=IMGS_RAW, lbls_dir=LBLS_RAW,
23     conf_thresh=CONF_THRESH, max_images=MAX_IMAGES_PER_CLASS,
24     done_stems=progress.get('fresh', []),
25     batch_size=500,
26     save_fn=make_save_fn('fresh'),
27 )
28
29 print('\nPseudo-labeling Deteriorado (clase 1)...')
30 rotten_stems = pseudo_label_folder(
31     pseudo_model, ROTTEN_SRC, class_id=1,
32     imgs_dir=IMGS_RAW, lbls_dir=LBLS_RAW,
33     conf_thresh=CONF_THRESH, max_images=MAX_IMAGES_PER_CLASS,
34     done_stems=progress.get('rotten', []),
35     batch_size=500,
36     save_fn=make_save_fn('rotten'),
37 )
38
39 print(f'\nTotal – Fresco: {len(fresh_stems)}, Deteriorado: {len(rotten_stems)}')
40 print(f'Checkpoint final guardado en {CHECKPOINT_F}')
41

```

Reanudando – Fresh: 37040, Rotten: 8344

Pseudo-labeling Fresco (clase 0)...

Total encontradas : 0

Pendientes : 0 | Ya procesadas: 37040

Clase 0 (Fresco): 37040 aceptadas, 0 descartadas (0.0%)

Pseudo-labeling Deteriorado (clase 1)...

Total encontradas : 23977

Pendientes : 15633 | Ya procesadas: 8344

Batch 1 [500/15633] +412 aceptadas | total=8756 (1751%) | descartadas=88

Batch	2	[1000/15633]	+429 aceptadas	total=9185 (918%)	descartadas=159
Batch	3	[1500/15633]	+426 aceptadas	total=9611 (641%)	descartadas=233
Batch	4	[2000/15633]	+423 aceptadas	total=10034 (502%)	descartadas=310
Batch	5	[2500/15633]	+428 aceptadas	total=10462 (418%)	descartadas=382
Batch	6	[3000/15633]	+419 aceptadas	total=10881 (363%)	descartadas=463
Batch	7	[3500/15633]	+426 aceptadas	total=11307 (323%)	descartadas=537
Batch	8	[4000/15633]	+417 aceptadas	total=11724 (293%)	descartadas=620
Batch	9	[4500/15633]	+413 aceptadas	total=12137 (270%)	descartadas=707
Batch	10	[5000/15633]	+433 aceptadas	total=12570 (251%)	descartadas=774
Batch	11	[5500/15633]	+422 aceptadas	total=12992 (236%)	descartadas=852
Batch	12	[6000/15633]	+422 aceptadas	total=13414 (224%)	descartadas=930
Batch	13	[6500/15633]	+406 aceptadas	total=13820 (213%)	descartadas=1024
Batch	14	[7000/15633]	+447 aceptadas	total=14267 (204%)	descartadas=1077
Batch	15	[7500/15633]	+472 aceptadas	total=14739 (197%)	descartadas=1105
Batch	16	[8000/15633]	+454 aceptadas	total=15193 (190%)	descartadas=1151
Batch	17	[8500/15633]	+462 aceptadas	total=15655 (184%)	descartadas=1189
Batch	18	[9000/15633]	+460 aceptadas	total=16115 (179%)	descartadas=1229
Batch	19	[9500/15633]	+457 aceptadas	total=16572 (174%)	descartadas=1272
Batch	20	[10000/15633]	+461 aceptadas	total=17033 (170%)	descartadas=1311
Batch	21	[10500/15633]	+456 aceptadas	total=17489 (167%)	descartadas=1355
Batch	22	[11000/15633]	+461 aceptadas	total=17950 (163%)	descartadas=1394
Batch	23	[11500/15633]	+467 aceptadas	total=18417 (160%)	descartadas=1427
Batch	24	[12000/15633]	+457 aceptadas	total=18874 (157%)	descartadas=1470
Batch	25	[12500/15633]	+455 aceptadas	total=19329 (155%)	descartadas=1515
Batch	26	[13000/15633]	+459 aceptadas	total=19788 (152%)	descartadas=1556
Batch	27	[13500/15633]	+459 aceptadas	total=20247 (150%)	descartadas=1597
Batch	28	[14000/15633]	+458 aceptadas	total=20705 (148%)	descartadas=1639
Batch	29	[14500/15633]	+454 aceptadas	total=21159 (146%)	descartadas=1685
Batch	30	[15000/15633]	+462 aceptadas	total=21621 (144%)	descartadas=1723
Batch	31	[15500/15633]	+466 aceptadas	total=22087 (142%)	descartadas=1757
Batch	32	[15633/15633]	+127 aceptadas	total=22214 (142%)	descartadas=1763
Clase 1 (Deteriorado): 22214 aceptadas, 1763 descartadas (7.4%)					

Total – Fresco: 37040, Deteriorado: 22214
Checkpoint final guardado en /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/pseudo_progress.json

Resultados pseudo labeling fresh

Reanudando – Fresh: 7570, Rotten: 8344

Pseudo-labeling Fresco (clase 0)...

Total encontradas : 41160

Pendientes : 33590 | Ya procesadas: 7570

Batch	1	[500/33590]	+401 aceptadas	total=7971 (1594%)	descartadas=99
Batch	2	[1000/33590]	+413 aceptadas	total=8384 (838%)	descartadas=186
Batch	3	[1500/33590]	+401 aceptadas	total=8785 (586%)	descartadas=285
Batch	4	[2000/33590]	+402 aceptadas	total=9187 (459%)	descartadas=383
Batch	5	[2500/33590]	+402 aceptadas	total=9589 (384%)	descartadas=481
Batch	6	[3000/33590]	+392 aceptadas	total=9981 (333%)	descartadas=589
Batch	7	[3500/33590]	+406 aceptadas	total=10387 (297%)	descartadas=683
Batch	8	[4000/33590]	+402 aceptadas	total=10789 (270%)	descartadas=781
Batch	9	[4500/33590]	+403 aceptadas	total=11192 (249%)	descartadas=878
Batch	10	[5000/33590]	+424 aceptadas	total=11616 (232%)	descartadas=954
Batch	11	[5500/33590]	+416 aceptadas	total=12032 (219%)	descartadas=1038
Batch	12	[6000/33590]	+411 aceptadas	total=12443 (207%)	descartadas=1127
Batch	13	[6500/33590]	+399 aceptadas	total=12842 (198%)	descartadas=1228
Batch	14	[7000/33590]	+406 aceptadas	total=13248 (189%)	descartadas=1322
Batch	15	[7500/33590]	+413 aceptadas	total=13661 (182%)	descartadas=1409
Batch	16	[8000/33590]	+439 aceptadas	total=14100 (176%)	descartadas=1470
Batch	17	[8500/33590]	+448 aceptadas	total=14548 (171%)	descartadas=1522
Batch	18	[9000/33590]	+440 aceptadas	total=14988 (167%)	descartadas=1582
Batch	19	[9500/33590]	+446 aceptadas	total=15434 (162%)	descartadas=1636
Batch	20	[10000/33590]	+441 aceptadas	total=15875 (159%)	descartadas=1695
Batch	21	[10500/33590]	+455 aceptadas	total=16330 (156%)	descartadas=1740
Batch	22	[11000/33590]	+445 aceptadas	total=16775 (152%)	descartadas=1795
Batch	23	[11500/33590]	+446 aceptadas	total=17221 (150%)	descartadas=1849
Batch	24	[12000/33590]	+440 aceptadas	total=17661 (147%)	descartadas=1909
Batch	25	[12500/33590]	+453 aceptadas	total=18114 (145%)	descartadas=1956
Batch	26	[13000/33590]	+444 aceptadas	total=18558 (143%)	descartadas=2012
Batch	27	[13500/33590]	+457 aceptadas	total=19015 (141%)	descartadas=2055
Batch	28	[14000/33590]	+445 aceptadas	total=19460 (139%)	descartadas=2110
Batch	29	[14500/33590]	+435 aceptadas	total=19895 (137%)	descartadas=2175
Batch	30	[15000/33590]	+439 aceptadas	total=20334 (136%)	descartadas=2236
Batch	31	[15500/33590]	+437 aceptadas	total=20771 (134%)	descartadas=2299
Batch	32	[16000/33590]	+440 aceptadas	total=21211 (133%)	descartadas=2359
Batch	33	[16500/33590]	+440 aceptadas	total=21651 (131%)	descartadas=2419

Batch	34	[17000/33590]	+447 aceptadas		total=22098 (130%)		descartadas=2472
Batch	35	[17500/33590]	+449 aceptadas		total=22547 (129%)		descartadas=2523
Batch	36	[18000/33590]	+454 aceptadas		total=23001 (128%)		descartadas=2569
Batch	37	[18500/33590]	+459 aceptadas		total=23460 (127%)		descartadas=2610
Batch	38	[19000/33590]	+438 aceptadas		total=23898 (126%)		descartadas=2672
Batch	39	[19500/33590]	+465 aceptadas		total=24363 (125%)		descartadas=2707
Batch	40	[20000/33590]	+453 aceptadas		total=24816 (124%)		descartadas=2754
Batch	41	[20500/33590]	+451 aceptadas		total=25267 (123%)		descartadas=2803
Batch	42	[21000/33590]	+452 aceptadas		total=25719 (122%)		descartadas=2851
Batch	43	[21500/33590]	+445 aceptadas		total=26164 (122%)		descartadas=2906
Batch	44	[22000/33590]	+447 aceptadas		total=26611 (121%)		descartadas=2959
Batch	45	[22500/33590]	+456 aceptadas		total=27067 (120%)		descartadas=3003
Batch	46	[23000/33590]	+443 aceptadas		total=27510 (120%)		descartadas=3060
Batch	47	[23500/33590]	+449 aceptadas		total=27959 (119%)		descartadas=3111
Batch	48	[24000/33590]	+452 aceptadas		total=28411 (118%)		descartadas=3159
Batch	49	[24500/33590]	+462 aceptadas		total=28873 (118%)		descartadas=3197
Batch	50	[25000/33590]	+448 aceptadas		total=29321 (117%)		descartadas=3249
Batch	51	[25500/33590]	+455 aceptadas		total=29776 (117%)		descartadas=3294
Batch	52	[26000/33590]	+434 aceptadas		total=30210 (116%)		descartadas=3360
Batch	53	[26500/33590]	+444 aceptadas		total=30654 (116%)		descartadas=3416
Batch	54	[27000/33590]	+446 aceptadas		total=31100 (115%)		descartadas=3470
Batch	55	[27500/33590]	+446 aceptadas		total=31546 (115%)		descartadas=3524
Batch	56	[28000/33590]	+449 aceptadas		total=31995 (114%)		descartadas=3575
Batch	57	[28500/33590]	+451 aceptadas		total=32446 (114%)		descartadas=3624
Batch	58	[29000/33590]	+454 aceptadas		total=32900 (113%)		descartadas=3670
Batch	59	[29500/33590]	+458 aceptadas		total=33358 (113%)		descartadas=3712
Batch	60	[30000/33590]	+450 aceptadas		total=33808 (113%)		descartadas=3762
Batch	61	[30500/33590]	+447 aceptadas		total=34255 (112%)		descartadas=3815
Batch	62	[31000/33590]	+447 aceptadas		total=34702 (112%)		descartadas=3868
Batch	63	[31500/33590]	+453 aceptadas		total=35155 (112%)		descartadas=3915
Batch	64	[32000/33590]	+440 aceptadas		total=35595 (111%)		descartadas=3975
Batch	65	[32500/33590]	+448 aceptadas		total=36043 (111%)		descartadas=4027
Batch	66	[33000/33590]	+458 aceptadas		total=36501 (111%)		descartadas=4069
Batch	67	[33500/33590]	+458 aceptadas		total=36959 (110%)		descartadas=4111
Batch	68	[33590/33590]	+81 aceptadas		total=37040 (110%)		descartadas=4120
Clase 0 (Fresco): 37040 aceptadas, 4120 descartadas (10.0%)							

Resultados Pseudo labeling rotten

Reanudando – Fresh: 37040, Rotten: 8344							
Pseudo-labeling Fresco (clase 0)...							
Total encontradas : 0							
Pendientes : 0 Ya procesadas: 37040							
Clase 0 (Fresco): 37040 aceptadas, 0 descartadas (0.0%)							
Pseudo-labeling Deteriorado (clase 1)...							
Total encontradas : 23977							
Pendientes : 15633 Ya procesadas: 8344							
Batch	1	[500/15633]	+412 aceptadas		total=8756 (1751%)		descartadas=88
Batch	2	[1000/15633]	+429 aceptadas		total=9185 (918%)		descartadas=159
Batch	3	[1500/15633]	+426 aceptadas		total=9611 (641%)		descartadas=233
Batch	4	[2000/15633]	+423 aceptadas		total=10034 (502%)		descartadas=310
Batch	5	[2500/15633]	+428 aceptadas		total=10462 (418%)		descartadas=382
Batch	6	[3000/15633]	+419 aceptadas		total=10881 (363%)		descartadas=463
Batch	7	[3500/15633]	+426 aceptadas		total=11307 (323%)		descartadas=537
Batch	8	[4000/15633]	+417 aceptadas		total=11724 (293%)		descartadas=620
Batch	9	[4500/15633]	+413 aceptadas		total=12137 (270%)		descartadas=707
Batch	10	[5000/15633]	+433 aceptadas		total=12570 (251%)		descartadas=774
Batch	11	[5500/15633]	+422 aceptadas		total=12992 (236%)		descartadas=852
Batch	12	[6000/15633]	+422 aceptadas		total=13414 (224%)		descartadas=930
Batch	13	[6500/15633]	+406 aceptadas		total=13820 (213%)		descartadas=1024
Batch	14	[7000/15633]	+447 aceptadas		total=14267 (204%)		descartadas=1077
Batch	15	[7500/15633]	+472 aceptadas		total=14739 (197%)		descartadas=1105
Batch	16	[8000/15633]	+454 aceptadas		total=15193 (190%)		descartadas=1151
Batch	17	[8500/15633]	+462 aceptadas		total=15655 (184%)		descartadas=1189
Batch	18	[9000/15633]	+460 aceptadas		total=16115 (179%)		descartadas=1229
Batch	19	[9500/15633]	+457 aceptadas		total=16572 (174%)		descartadas=1272
Batch	20	[10000/15633]	+461 aceptadas		total=17033 (170%)		descartadas=1311

```

Batch 21 [10500/15633] +456 aceptadas | total=17489 (167%) | descartadas=1355
Batch 22 [11000/15633] +461 aceptadas | total=17950 (163%) | descartadas=1394
Batch 23 [11500/15633] +467 aceptadas | total=18417 (160%) | descartadas=1427
Batch 24 [12000/15633] +457 aceptadas | total=18874 (157%) | descartadas=1470
Batch 25 [12500/15633] +455 aceptadas | total=19329 (155%) | descartadas=1515
Batch 26 [13000/15633] +459 aceptadas | total=19788 (152%) | descartadas=1556
Batch 27 [13500/15633] +459 aceptadas | total=20247 (150%) | descartadas=1597
Batch 28 [14000/15633] +458 aceptadas | total=20705 (148%) | descartadas=1639
Batch 29 [14500/15633] +454 aceptadas | total=21159 (146%) | descartadas=1685
Batch 30 [15000/15633] +462 aceptadas | total=21621 (144%) | descartadas=1723
Batch 31 [15500/15633] +466 aceptadas | total=22087 (142%) | descartadas=1757
Batch 32 [15633/15633] +127 aceptadas | total=22214 (142%) | descartadas=1763
Clase 1 (Deteriorado): 22214 aceptadas, 1763 descartadas (7.4%)

Total – Fresco: 37040, Deteriorado: 22214
Checkpoint final guardado en /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/pseudo_progress.json

```

2. Split train/val y data.yamll

```

1 # YOLO necesita /images/ en la ruta para encontrar labels automáticamente
2 # (reemplaza /images/ → /labels/ internamente)
3 # Solución: symlinks locales que apuntan a las carpetas de Drive
4 import os
5
6 os.makedirs('/content/fridgeeye', exist_ok=True)
7 for link, target in [('/content/fridgeeye/images', str(IMGs_RAW)),
8                      ('/content/fridgeeye/labels', str(LBLS_RAW))]:
9     if os.path.islink(link):
10         os.unlink(link)
11     os.symlink(target, link)
12
13 LOCAL_IMGs = Path('/content/fridgeeye/images')
14 LOCAL_LBLS = Path('/content/fridgeeye/labels')
15 print('Symlinks listos:')
16 print(f' {LOCAL_IMGs} → {IMGs_RAW}')
17 print(f' {LOCAL_LBLS} → {LBLS_RAW}')
18
19 # Eliminar caches incorrectos de ejecuciones previas
20 for cache in WORK_DIR.glob('*.cache'):
21     cache.unlink()
22     print(f' Cache eliminado: {cache.name}')
23

```

Symlinks listos:

```

/content/fridgeeye/images → /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/images_raw
/content/fridgeeye/labels → /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/labels_raw

```

```

1 def split_dataset_txt(imgs_dir, lbls_dir, work_dir, local_imgs, val_ratio=0.20, seed=42):
2     """Genera train.txt y val.txt con rutas via symlink – sin copiar archivos."""
3     imgs_dir = Path(imgs_dir)
4     lbls_dir = Path(lbls_dir)
5     work_dir = Path(work_dir)
6     local_imgs = Path(local_imgs)
7
8     # Un solo glob para labels → set en memoria (evita 15K llamadas .exists() a Drive)
9     label_stems = {p.stem for p in lbls_dir.glob('*.txt')}
10    print(f' Labels en Drive : {len(label_stems)}')
11
12    all_imgs = sorted([
13        p for p in imgs_dir.glob('*.jpg')
14        if p.stem in label_stems
15    ])
16    print(f' Imágenes válidas: {len(all_imgs)}')
17
18    random.seed(seed)
19    random.shuffle(all_imgs)
20
21    n_val = int(len(all_imgs) * val_ratio)
22    val_imgs = all_imgs[:n_val]
23    train_imgs = all_imgs[n_val:]
24
25    # Rutas via symlink: /content/fridgeeye/images/xxx.jpg
26    # YOLO encontrará labels en: /content/fridgeeye/labels/xxx.txt
27    def to_local(p):
28        return str(local_imgs / p.name)
29

```

```

30 (work_dir / 'train.txt').write_text('\n'.join(to_local(p) for p in train_imgs))
31 (work_dir / 'val.txt').write_text('\n'.join(to_local(p) for p in val_imgs))
32
33 print(f'Split completado - train: {len(train_imgs)}, val: {len(val_imgs)}')
34 return {'train': len(train_imgs), 'val': len(val_imgs)}
35
36 counts = split_dataset_txt(IMGs_RAW, LBLS_RAW, WORK_DIR, LOCAL_IMGs, VAL_RATIO, SEED)
37

```

```

1 # YAML apunta a los .txt con rutas absolutas - YOLO encuentra labels
2 # reemplazando 'images_raw' → 'labels_raw' en cada ruta automáticamente
3 YAML_PATH = WORK_DIR / 'data.yaml'
4 yaml_content = {
5     'path': '/',
6     'train': str(WORK_DIR / 'train.txt'),
7     'val': str(WORK_DIR / 'val.txt'),
8     'nc': NC,
9     'names': NAMES,
10 }
11 YAML_PATH.write_text(yaml.dump(yaml_content, allow_unicode=True))
12 print('data.yaml:')
13 print(yaml.dump(yaml_content, allow_unicode=True))
14

```

```

data.yaml:
names:
- Fresco
- Deteriorado
nc: 2
path: /
train: /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/train.txt
val: /content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/val.txt

```

```

1 train_paths = [p for p in (WORK_DIR / 'train.txt').read_text().strip().split('\n') if p.strip()]
2 val_paths = [p for p in (WORK_DIR / 'val.txt').read_text().strip().split('\n') if p.strip()]
3
4 # El stem codifica la clase (c0_ = Fresco, c1_ = Deteriorado) - sin leer Drive
5 def count_classes(paths):
6     counts = collections.Counter()
7     for p in paths:
8         stem = Path(p).stem
9         if stem.startswith('c0_'): counts[0] += 1
10        elif stem.startswith('c1_'): counts[1] += 1
11    return counts
12
13 tr = count_classes(train_paths)
14 va = count_classes(val_paths)
15
16 print(f'{"Clase":<22} {"Train":>8} {"Val":>8} {"Total":>8}')
17 print('-' * 50)
18 for i, name in enumerate(NAMES):
19     print(f'{name:<22} {tr[i]:>8} {va[i]:>8} {tr[i]+va[i]:>8}')
20 print('-' * 50)
21 print(f'{"TOTAL":<22} {sum(tr.values()):>8} {sum(va.values()):>8} ')
22     f'{sum(tr.values())+sum(va.values()):>8}')
23

```

Clase	Train	Val	Total
Fresco	25189	6260	31449
Deteriorado	18565	4678	23243
TOTAL	43754	10938	54692

▽ prepare training

```

1 # Copia imágenes y labels de Drive a SSD local en paralelo
2 # El entrenamiento leerá desde disco local (~500 MB/s) en vez de Drive (~1 MB/s)
3 from concurrent.futures import ThreadPoolExecutor
4 import shutil, time
5
6 LOCAL_IMGs_DIR = Path('/content/local_imgs')
7 LOCAL_LBLS_DIR = Path('/content/local_lbls')
8 LOCAL_IMGs_DIR.mkdir(exist_ok=True)
9 LOCAL_LBLS_DIR.mkdir(exist_ok=True)
10
11 def _copy(src_dst):
12     shutil.copy2(src_dst[0], src_dst[1])
13

```

```

14 img_files = list(IMGs_RAW.glob('*.*'))
15 lbl_files = list(LBLS_RAW.glob('*.txt'))
16 print(f'Imágenes: {len(img_files)} | Labels: {len(lbl_files)}')
17
18 t0 = time.time()
19 print('Copiando imágenes a local (paralelo 8 workers)...')
20 with ThreadPoolExecutor(max_workers=8) as ex:
21     list(ex.map(_copy, [(f, LOCAL_IMGs_DIR / f.name) for f in img_files]))
22 print(f'  Listo en {time.time()-t0:.0f}s')
23
24 t1 = time.time()
25 print('Copiando labels a local...')
26 with ThreadPoolExecutor(max_workers=8) as ex:
27     list(ex.map(_copy, [(f, LOCAL_LBLS_DIR / f.name) for f in lbl_files]))
28 print(f'  Listo en {time.time()-t1:.0f}s')
29
30 # Redirigir symlinks de Drive → disco local
31 for link, target in [('/content/fridgeeye/images', str(LOCAL_IMGs_DIR)),
32                     ('/content/fridgeeye/labels', str(LOCAL_LBLS_DIR))]:
33     if os.path.islink(link):
34         os.unlink(link)
35     os.symlink(target, link)
36
37 total_gb = sum(f.stat().st_size for f in LOCAL_IMGs_DIR.glob('*')) / 1e9
38 print(f'\nSymlinks → disco local ({total_gb:.2f} GB)')
39 print('Listo para entrenar sin Drive I/O')
40

```

Imágenes: 54692 | Labels: 47572
 Copiando imágenes a local (paralelo 8 workers)...
 Listo en 2111s
 Copiando labels a local...
 Listo en 1871s

Symlinks → disco local (10.91 GB)
 Listo para entrenar sin Drive I/O

```

1 # Redirigir symlinks de Drive → disco local
2 for link, target in [('/content/fridgeeye/images', str(LOCAL_IMGs_DIR)),
3                     ('/content/fridgeeye/labels', str(LOCAL_LBLS_DIR))]:
4     if os.path.islink(link):
5         os.unlink(link)
6     os.symlink(target, link)
7
8 total_gb = sum(f.stat().st_size for f in LOCAL_IMGs_DIR.glob('*')) / 1e9
9 print(f'\nSymlinks → disco local ({total_gb:.2f} GB)')
10 print('Listo para entrenar sin Drive I/O')

```

Symlinks → disco local (10.91 GB)
 Listo para entrenar sin Drive I/O

```

1 # RUN TO RE START TRAINING
2 # fridgeeye_oficial_20260422_1316 -- BEST DEPRECATED
3 # NEW FINAL ROUTE fridgeeye_oficial_20260429_1412
4 RUN_NAME = 'fridgeeye_oficial_20260429_1412'
5 last_pt = f'{MODELS_PT}/{RUN_NAME}/weights/last.pt'
6 model = YOLO(last_pt)
7 results = model.train(resume=True)
8

```

Ultralytics 8.4.46 🚀 Python-3.12.13 torch-2.10.0+cu128 CUDA:0 (Tesla T4, 14913MiB)
engine/trainer: agnostic_nms=False, amp=True, angle=1.0, augment=False, auto_augment=randaugment, batch=32, bgr=0.0, box=7
 Downloading <https://ultralytics.com/assets/Arial.ttf> to '/root/.config/Ultralytics/Arial.ttf': 100% ————— 755.1KB 2

		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 751702 ultralytics.nn.modules.head.Detect [2, 16, None, [64, 128, 256]]
Model summary: 130 layers, 3,011,238 parameters, 3,011,222 gradients, 8.2 GFLOPs

```

Transferred 355/355 items from pretrained weights

Freezing layer 'model.22.dfl.conv.weight'

AMP: running Automatic Mixed Precision (AMP) checks...

Downloading <https://github.com/ultralytics/assets/releases/download/v8.4.0/yolo26n.pt> to 'yolo26n.pt': 100% ————— 5

AMP: checks passed ☒

train: Fast image access ☒ (ping: 0.0±0.0 ms, read: 666.3±567.6 MB/s, size: 36.5 KB)

train: Scanning /content/fridgeeye/labels... 38043 images, 0 backgrounds, 0 corrupt: 100% ————— 38043/38043 352.7it

train: /content/fridgeeye/images/c0_000326_carrotFresh_7637.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000390_carrotFresh_7584.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000550_carrotFresh_7625.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000878_carrotFresh_7053.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001074_carrotFresh_7330.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001080_carrotFresh_7623.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001265_carrotFresh_7590.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001418_carrotFresh_7688.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001645_carrotFresh_7687.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001697_carrotFresh_9846.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001859_carrotFresh_9857.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001936_carrotFresh_7689.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002160_carrotFresh_7629.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002173_carrotFresh_7566.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002234_carrotFresh_7640.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002264_carrotFresh_7255.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002567_carrotFresh_9860.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002588_carrotFresh_7601.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_003089_carrotFresh_9856.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_003207_carrotFresh_9895.jpg: corrupt JPEG restored and saved

```

1 model = YOLO('yolov8n.pt')
2
3 results = model.train(
4     data          = str(YAML_PATH),
5     epochs        = EPOCHS,
6     patience      = PATIENCE,
7     batch         = BATCH,
8     imgsz         = IMGSZ,
9     lr0           = LR0,
10    lrf            = 0.01,
11    optimizer      = 'AdamW',
12    weight_decay   = 0.0005,
13    warmup_epochs  = 3,
14    cos_lr         = True,
15    label_smoothing = 0.1,
16    # Augmentación agresiva
17    hsv_h          = 0.02,
18    hsv_s          = 0.75,
19    hsv_v          = 0.4,
20    degrees        = 10.0,
21    translate      = 0.15,
22    scale          = 0.5,
23    flipud         = 0.3,
24   fliplr         = 0.5,
25    mosaic         = 1.0,
26    mixup          = 0.1,
27    copy_paste     = 0.1,
28    erasing        = 0.3,
29    # Sistema
30    project        = MODELS_PT,
31    name           = RUN_NAME,
32    exist_ok       = True,
33    verbose        = True,
34    device         = device,
35    cache          = False,
36    amp            = True,
37 )
38
39 best_pt = Path(MODELS_PT) / RUN_NAME / 'weights' / 'best.pt'
40 print(f'\nMejor modelo guardado en: {best_pt}')
41

```

Downloading <https://github.com/ultralytics/assets/releases/download/v8.4.0/yolov8n.pt> to 'yolov8n.pt': 100% ————— 6

New <https://pypi.org/project/ultralytics/8.4.45> available 🤖 Update with 'pip install -U ultralytics'

WARNING ⚠️ 'label_smoothing' is deprecated and will be removed in the future.

Ultralytics 8.4.43 🚀 Python-3.12.13 torch-2.10.0+cu128 CUDA:0 (Tesla T4, 14913MiB)

engine/trainer: agnostic_nms=False, amp=True, angle=1.0, augment=False, auto_augment=randaugument, batch=32, bgr=0.0, box=7

Downloading <https://ultralytics.com/assets/Arial.ttf> to '/root/.config/Ultralytics/Arial.ttf': 100% ————— 755.1KB 1

Overriding model.yaml nc=80 with nc=2

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    751702  ultralytics.nn.modules.head.Detect      [2, 16, None, [64, 128, 256]]
Model summary: 130 layers, 3,011,238 parameters, 3,011,222 gradients, 8.2 GFLOPs

```

Transferred 319/355 items from pretrained weights

Freezing layer 'model.22.dfl.conv.weight'

AMP: running Automatic Mixed Precision (AMP) checks...

Downloading <https://github.com/ultralytics/assets/releases/download/v8.4.0/yolo26n.pt> to 'yolo26n.pt': 100% ————— 5

AMP: checks passed 

train: Fast image access  (ping: 0.0±0.0 ms, read: 270.4±348.8 MB/s, size: 46.1 KB)

train: Scanning /content/fridgeeye/labels... 38043 images, 0 backgrounds, 0 corrupt: 100% ————— 38043/38043 382.3it

train: /content/fridgeeye/images/c0_000326_carrotFresh_7637.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000390_carrotFresh_7584.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000550_carrotFresh_7625.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_000878_carrotFresh_7053.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001074_carrotFresh_7330.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001080_carrotFresh_7623.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001265_carrotFresh_7590.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001418_carrotFresh_7688.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001645_carrotFresh_7687.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001697_carrotFresh_9846.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001859_carrotFresh_9857.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_001936_carrotFresh_7689.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002160_carrotFresh_7629.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002173_carrotFresh_7566.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002234_carrotFresh_7640.jpg: corrupt JPEG restored and saved

train: /content/fridgeeye/images/c0_002264_carrotFresh_7255.jpg: corrupt JPEG restored and saved

```

1 import pandas as pd
2
3 results_csv = Path(MODELS_PT) / RUN_NAME / 'results.csv'
4 df = pd.read_csv(results_csv)
5 df.columns = df.columns.str.strip()
6
7 p = df['metrics/precision(B)'].values
8 r = df['metrics/recall(B)'].values
9 f1 = 2 * p * r / (p + r + 1e-9)
10 map50 = df['metrics/mAP50(B)'].values
11
12 best_f1_ep = int(f1.argmax())
13 best_map_ep = int(map50.argmax())
14
15 print(f'Épocas entrenadas : {len(df)}')
16 print(f'Mejor F1          : {f1[best_f1_ep]:.4f} (época {best_f1_ep+1})')
17 print(f' Precision        : {p[best_f1_ep]:.4f}')
18 print(f' Recall           : {r[best_f1_ep]:.4f}')
19 print(f'Mejor mAP50        : {map50[best_map_ep]:.4f} (época {best_map_ep+1})')
20 print()
21 rnf01 = 'CUMPLE' if f1.max() >= 0.85 else f'FALTA {0.85 - f1.max():.4f} para RNF01'
22 print(f'RNF01 (F1>=85%)   : {rnf01}')
23

```

```

Épocas entrenadas : 120
Mejor F1          : 0.9547 (época 99)
 Precision        : 0.9528
 Recall           : 0.9565
Mejor mAP50        : 0.9853 (época 92)

```

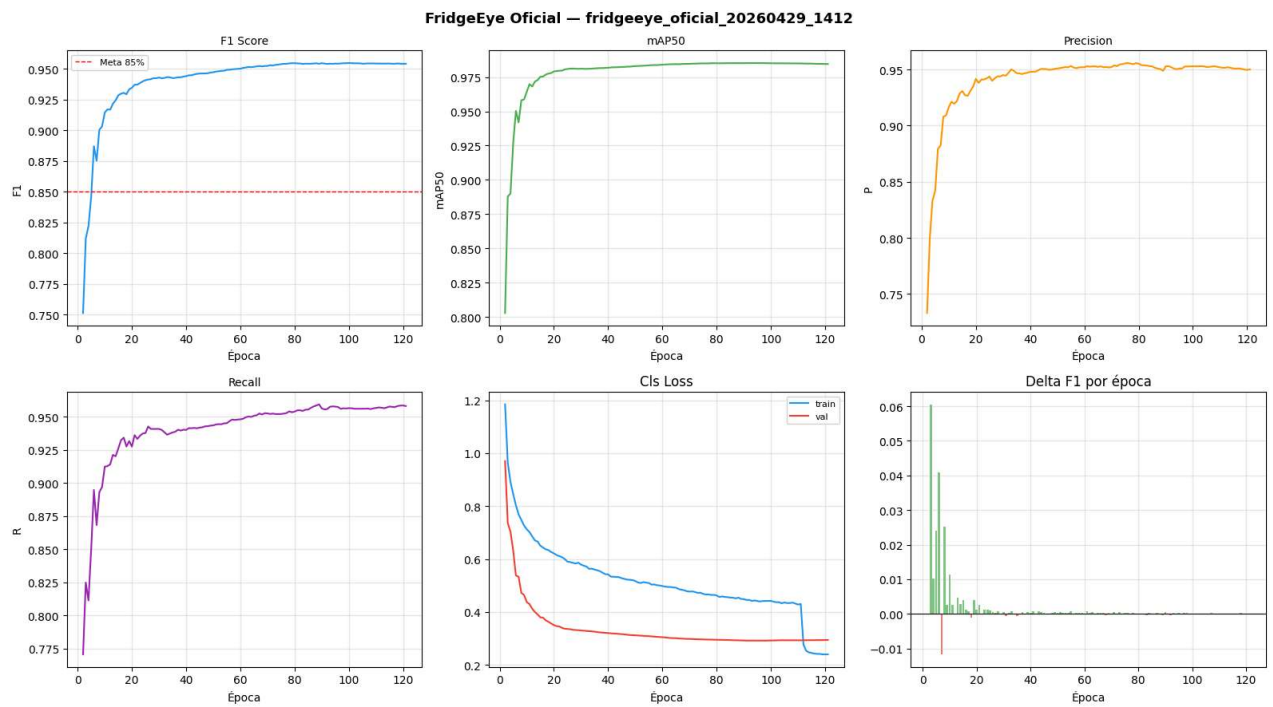
```
RNF01 (F1>=85%) : CUMPLE
```

4. Visualización de entrenamiento

```

1 import matplotlib.pyplot as plt
2
3 df = pd.read_csv(Path(MODELS_PT) / RUN_NAME / 'results.csv')
4 df.columns = df.columns.str.strip()
5
6 p      = df['metrics/precision(B)'].values
7 r      = df['metrics/recall(B)'].values
8 f1     = 2 * p * r / (p + r + 1e-9)
9 map50  = df['metrics/mAP50(B)'].values
10 epochs_x = df['epoch'].values + 1
11
12 fig, axes = plt.subplots(2, 3, figsize=(16, 9))
13 fig.suptitle(f'FridgeEye Oficial - {RUN_NAME}', fontsize=13, fontweight='bold')
14
15 def _plot(ax, y, title, ylabel, color, hline=None):
16     ax.plot(epochs_x, y, color=color, linewidth=1.5)
17     ax.set_title(title, fontsize=10)
18     ax.set_xlabel('Época')
19     ax.set_ylabel(ylabel)
20     ax.grid(True, alpha=0.3)
21     if hline:
22         ax.axhline(hline, color='red', linestyle='--', linewidth=1,
23                    label=f'Meta {hline:.0%}')
24     ax.legend(fontsize=8)
25
26 _plot(axes[0,0], f1, 'F1 Score', 'F1', '#2196F3', hline=0.85)
27 _plot(axes[0,1], map50, 'mAP50', 'mAP50', '#4CAF50')
28 _plot(axes[0,2], p, 'Precision', 'P', '#FF9800')
29 _plot(axes[1,0], r, 'Recall', 'R', '#9C27B0')
30
31 if 'train/cls_loss' in df.columns:
32     axes[1,1].plot(epochs_x, df['train/cls_loss'], label='train', color='#2196F3')
33     axes[1,1].plot(epochs_x, df['val/cls_loss'], label='val', color='#F44336')
34     axes[1,1].set_title('Cls Loss'); axes[1,1].set_xlabel('Época')
35     axes[1,1].legend(fontsize=8); axes[1,1].grid(True, alpha=0.3)
36
37 delta = np.diff(f1, prepend=f1[0])
38 cols = ['#4CAF50' if d >= 0 else '#F44336' for d in delta]
39 axes[1,2].bar(epochs_x, delta, color=cols, alpha=0.7)
40 axes[1,2].axhline(0, color='black', linewidth=0.8)
41 axes[1,2].set_title('Delta F1 por época'); axes[1,2].set_xlabel('Época')
42 axes[1,2].grid(True, alpha=0.3)
43
44 plt.tight_layout()
45 report_path = f'{REPORTS_DIR}/{RUN_NAME}_curves.png'
46 plt.savefig(report_path, dpi=150, bbox_inches='tight')
47 plt.show()
48 print(f'Guardado: {report_path}')
49

```



5. Exportación TFLite

```

1 BEST_PT = str(Path(MODELS_PT) / 'fridgeeye_oficial_20260429_1412' / 'weights' / 'best.pt')
2 exp_model = YOLO(BEST_PT)
3
4 exported_files = {}
5
6 for exp_imgsz in [320, 640]:
7     print(f'\nExportando TFLite imgsz={exp_imgsz} fp16...')
8     exported = exp_model.export(
9         format = 'tflite',
10        imgsz = exp_imgsz,
11        nms = True,
12        simplify = True,
13        half = True,
14    )
15    tflite_src = Path(str(exported))
16    if not tflite_src.exists():
17        candidates = list(Path(BEST_PT).parent.rglob('*.tflite'))
18        tflite_src = max(candidates, key=lambda p: p.stat().st_mtime) if candidates else None
19
20    if tflite_src and tflite_src.exists():
21        dst_name = f'fridgeeye_{exp_imgsz}_fp16.tflite'
22        dst_path = Path(MODELS_TFLITE) / dst_name
23        shutil.copy2(tflite_src, dst_path)
24        size_mb = dst_path.stat().st_size / 1e6
25        exported_files[exp_imgsz] = {'path': str(dst_path), 'size_mb': round(size_mb, 2)}
26        rnf04 = 'OK' if size_mb < 20 else 'FAIL'
27        print(f' -> {dst_name} | {size_mb:.1f} MB | RNF04: {rnf04}')
28    else:

```

Exportando TFLite imsz=320 fp16...

Ultralytics 8.4.51 🚀 Python-3.12.13 torch-2.10.0+cu128 CPU (Intel Xeon CPU @ 2.00GHz)

💡 ProTip: Export to OpenVINO format for best performance on Intel hardware. Learn more at <https://docs.ultralytics.com/integrations/openvino>

Model summary (fused): 73 layers, 3,006,038 parameters, 0 gradients, 8.1 GFLOPs

WARNING **requirements:** Restart runtime or rerun command for updates to take effect

WARNING ! requirements: Restart runtime or rerun command for updates to take effect

Captures:

13/16

```

26     print(f'{exp_imgsz:>8} Error: {e}')
27

```

imgsz	Latencia ms	RNF02 <3000ms
320	31.1	OK
640	123.0	OK

6. Guardar artefactos y descargar

```

1 import zipfile
2 from google.colab import files as colab_files
3 import pandas as pd
4
5 # Re-read the results.csv and calculate f1 and map50 to ensure they are defined
6 results_csv = Path(MODELS_PT) / RUN_NAME / 'results.csv'
7 df = pd.read_csv(results_csv)
8 df.columns = df.columns.str.strip()
9
10 p = df['metrics/precision(B)'].values
11 r = df['metrics/recall(B)'].values
12 f1 = 2 * p * r / (p + r + 1e-9)
13 map50 = df['metrics/mAP50(B)'].values
14
15
16 summary = {
17     'run_name': RUN_NAME,
18     'best_pt': BEST_PT,
19     'tflite': exported_files,
20     'metrics': {
21         'best_f1': float(f1.max()),
22         'best_map50': float(map50.max()),
23         'epoch_f1': int(f1.argmax()) + 1,
24     },
25 }
26 summary_path = f'{REPORTS_DIR}/{RUN_NAME}_summary.json'
27 Path(summary_path).write_text(json.dumps(summary, indent=2))
28 print(f'Resumen: {summary_path}')
29
30 # ZIP en Drive (persistente) + descarga local
31 zip_path = f'{REPORTS_DIR}/{RUN_NAME}_artifacts.zip'
32 with zipfile.ZipFile(zip_path, 'w', zipfile.ZIP_DEFLATED) as z:
33     z.write(BEST_PT, arcname='best.pt')
34     z.write(summary_path, arcname='summary.json')
35     for info in exported_files.values():
36         z.write(info['path'], arcname=Path(info['path']).name)
37
38 size_mb = Path(zip_path).stat().st_size / 1e6
39 print(f'ZIP en Drive: {zip_path} ({size_mb:.1f} MB)')
40 colab_files.download(zip_path)

```

```

-----
NameError                                Traceback (most recent call last)
/tmp/ipykernel_2077/4036403974.py in <cell line: 0>()
      4
----> 5 # Re-read the results.csv and calculate f1 and map50 to ensure they are defined
      6 results_csv = Path(MODELS_PT) / RUN_NAME / 'results.csv'
      7 df = pd.read_csv(results_csv)
      8 df.columns = df.columns.str.strip()

NameError: name 'Path' is not defined

```

```

1 ## 7. Prueba con imágenes propias
2 from google.colab import files as colab_files
3 import matplotlib.pyplot as plt
4 import matplotlib.patches as patches
5 from PIL import Image
6
7 COLORS = {'Fresco': '#4CAF50', 'Deteriorado': '#F44336'}
8
9 print('Sube una o más imágenes para probar el modelo...')
10 uploaded = colab_files.upload()
11
12 model_test = YOLO(str(Path(MODELS_PT) / 'fridgeeye_oficial_20260429_1412' / 'weights' / 'best.pt'))
13
14 for fname, data in uploaded.items():
15     img_path = f'/content/{fname}'
16     with open(img_path, 'wb') as f:
17         f.write(data)

```

```

18
19 results = model_test.predict(img_path, conf=0.25, verbose=False)
20 res = results[0]
21
22 img = Image.open(img_path).convert('RGB')
23 fig, ax = plt.subplots(1, 1, figsize=(8, 6))
24 ax.imshow(img)
25 ax.axis('off')
26
27 if res.bboxes and len(res.bboxes):
28     for box in res.bboxes:
29         cls_id      = int(box.cls)
30         label       = NAMES[cls_id]
31         conf        = box.conf.item()
32         x1, y1, x2, y2 = box.xyxy[0].tolist()
33         color = COLORS[label]
34         rect = patches.Rectangle((x1, y1), x2-x1, y2-y1,
35                                 linewidth=2, edgecolor=color, facecolor='none')
36         ax.add_patch(rect)
37         ax.text(x1, y1 - 6, f'{label} {conf:.0%}',
38               color='white', fontsize=11, fontweight='bold',
39               bbox=dict(facecolor=color, edgecolor='none', pad=2))
40     ax.set_title(fname, fontsize=10)
41 else:
42     ax.set_title(f'{fname} - sin detección (conf < 0.25)', fontsize=10)
43
44 plt.tight_layout()
45 plt.show()

```

Sube una o más imágenes para probar el modelo...

No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun

this cell to enable.

Saving una-pera-la-mitad-con-su-núcleo-interior-y-semillas-que-enfatizan-texturas-naturales-sobre-fondo-blanco-un-cierre-de-una-pera-la-mitad-con-su-núcleo-interior-y-semillas-que-enfatizan-texturas-naturales-sobre-fondo-blanco-un-cierre-de-en-reveló-386187407.webp



```

1 import seaborn as sns
2 import matplotlib.pyplot as plt
3 from ultralytics import YOLO
4 from pathlib import Path
5
6 # Cargar el mejor modelo
7 model_eval = YOLO(str(Path(MODELS_PT) / 'fridgeeye_oficial_20260429_1412' / 'weights' / 'best.pt'))
8
9 # Realizar la validación para obtener las métricas y la matriz de confusión
10 print("Evaluando el modelo para generar la matriz de confusión...")
11 # Es importante pasar la ruta del dataset (data.yaml) para evitar errores de archivos no encontrados
12 yaml_path = '/content/drive/MyDrive/EAN/Proyecto_de_grado/TrainingData/data.yaml'
13 metrics = model_eval.val(data=yaml_path, verbose=False)
14
15 # Obtener la matriz de confusión cruda (usualmente incluye la clase "background")
16 raw_matrix = metrics.confusion_matrix.matrix
17

```

```
18 # Extraer solo la submatriz correspondiente a nuestras clases (2x2)
19 nc = len(NAMES) # Debería ser 2 ('Fresco' y 'Deteriorado')
20 cm_core = raw_matrix[:nc, :nc]
21
22 # Graficar la matriz de confusión filtrada
23 plt.figure(figsize=(7, 6))
24 sns.heatmap(cm_core, annot=True, fmt='.0f', cmap='Blues',
25             xticklabels=NAMES, yticklabels=NAMES, cbar=False)
26
27 plt.ylabel('Clase Verdadera', fontsize=12)
28 plt.xlabel('Clase Predicha', fontsize=12)
29 plt.title('Matriz de Confusión: Fresco vs Deteriorado', fontsize=14, pad=15)
30 plt.xticks(fontsize=11)
31 plt.yticks(fontsize=11, rotation=0)
32
33 plt.tight_layout()
34 plt.show()
35
```