# YOLO DATASET- MEDICAL IMAGE + TEXT

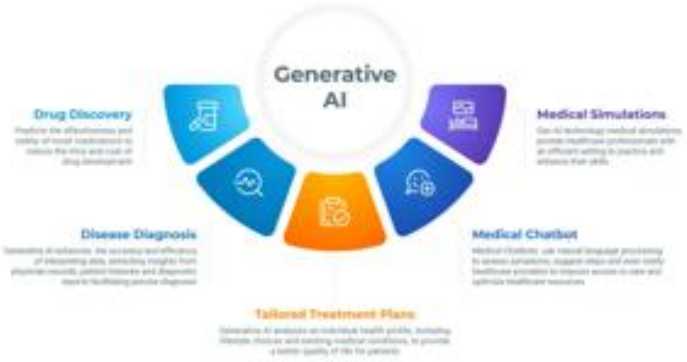
M SUMANA
ASSISTANT PROFESSOR,CSE DEPT
VELS UNIVERSITY
VISTAS

# WORKING CONCEPTS IN YOLO DATASET PROCESS

**How To Work In Medical Image Processing Using Genai For Documenation Purpose Generated**

Generative AI combined with image processing can analyze photos of medical lab equipment (e.g., microscopes, centrifuges, analyzers) to automatically generate documentation like usage logs, maintenance reports, calibration certificates, or inventory records. This streamlines lab workflows by detecting equipment state, defects, or usage via computer vision, then using GenAI for natural language summaries.pmc.ncbi.nlm.nih+3

**Key Applications**




- Equipment identification and inventory: Detects tools via object detection for asset tracking.
- Quality inspection: Spots defects or assembly issues in manufacturing/inspection.
- Predictive maintenance: Analyzes wear from images to generate repair guides or schedules.
- Usage documentation: Logs procedures from captured images, creating step-by-step reports.

**Core Technologies**

Object detection models like YOLO (Ultralytics) identify equipment in images. GenAI (e.g., LLMs via RAG) then generates text from detection outputs, such as "Centrifuge model XYZ shows no visible wear; recommend calibration."ultralytics+3

**Workflow Steps**

- **Capture and Preprocess**: Use cameras/IoT to image equipment; apply OpenCV/MONAI for enhancement.
- **Detection**: Train/fine-tune YOLO on lab equipment datasets for bounding boxes and labels.
- **Feature Extraction**: Analyze for defects (e.g., scratches) using segmentation or anomaly detection.
- **GenAI Generation**: Feed results to LLM (e.g., Llama via Hugging Face) with prompts for reports; use RAG for manuals/context.databricks+1

- **Output and Validate**: Produce PDFs/logs; clinician/lab tech reviews for accuracy.

**Tools and Resources**

- **Vision**: YOLOv8 for detection (ultralytics.com); Landing AI for device inspection.
- **GenAI**: Hugging Face Transformers, Databricks Mosaic AI for RAG-based docs.
- **Frameworks**: PyTorch, OpenCV; GitHub repos for surgical/lab tool detection adaptable to labs.

What specific lab equipment (e.g., microscope, analyzer) or focus (e.g., maintenance, inventory) do you have in mind?

**Is It Medical Lab Equipments** will be used as **YOLO**

Yes, YOLO (You Only Look Once) models are widely used and effective for detecting medical lab equipment through image processing. Versions like YOLOv8 achieve high accuracy (e.g., 94% precision, 92% recall) in identifying hospital assets including lab devices such as dialysis machines and infusion poles.umpir.ump+1

**Proven Applications**

YOLO excels in real-time object detection for lab settings, including chemistry lab apparatuses (25 classes, mAP@50 >0.98 with YOLOv11) and protective gear in labs. It's adapted for hospital equipment tracking, outperforming other models in speed and precision.

**Implementation Benefits**

Pre-trained YOLO models (e.g., YOLOv8 via Ultralytics) fine-tune easily on custom datasets like Roboflow's lab equipment images, enabling documentation generation from detections. Datasets for chemistry labs support direct training for tools like pipettes or centrifuges.

**Example Performance**

| Model | Dataset | mAP@50 | Use Case |
|---|---|---|---|
| YOLOv8 | Hospital assets (16 classes) | 0.90 | Medicaldevices tracking |
| YOLOv11 | Chemistry lab equipment (25 classes) | 0.987 | Apparatus recognition |
| YOLOv7 | Lab protective gear | High accuracy | Safety equipment |

**Will Coco Dataset Will Be Useful For Medical Lab Equipments For Image Processing?**

No, the COCO dataset is not directly useful for medical lab equipment detection in image processing. COCO's 80 classes cover everyday objects like "person," "backpack," or "sports ball" but exclude lab-specific items such as centrifuges, microscopes, or analyzers.

**Indirect Benefits**

Pre-trained YOLO models (e.g., YOLOv8) start from COCO weights, providing strong feature extraction for transfer learning—fine-tuning boosts lab equipment mAP by leveraging general shapes and edges. However, zero-shot performance on lab tools is low without custom data.

**Better Alternatives**

Use specialized datasets like the Real-world Chemistry Lab dataset (25 lab classes incl. centrifuges, mAP>0.98 on YOLOv11) or Roboflow lab/medical sets. Start with COCO-pretrained models, then fine-tune on 200+ lab images.

| Dataset | Lab Equipment Classes | Best for YOLO |
|---|---|---|
| COCO | 0 (none) | Transfer learning base |
| Chemistry Lab | 25 (incl. centrifuge) | Direct training |
| Hospital Assets | Medical devices | Healthcare adaptation |

**Medical Lab Equipments Dataset Should Be Considered For Summary Of Documentaion Geneerated.**

For generating documentation summaries from medical lab equipment detections (e.g., inventory reports, maintenance logs via YOLO + GenAI), prioritize domain-specific object detection datasets. These enable accurate equipment identification (centrifuges, microscopes, analyzers) before feeding bounding boxes/labels to LLMs for text generation.

**Top Datasets**

- **Real-world Chemistry Lab Dataset**: 4,599 images, 25 classes (beaker, pipette, funnel, centrifuge, etc.); real conditions (occlusion, lighting); ideal for YOLO (mAP@50=0.987), supports robust detection for documentation.
- **Hospital Assets Dataset**: Medical equipment (e.g., dialysis, infusion pumps); YOLOv8 achieves 94% precision/92% recall; directly relevant for healthcare labs.
- **LabEquipVis**: Annotated computer lab equipment (2,584 images); adaptable to medical with high-res images; great for general lab tools.
- **Laboratory Apparatus Detection (Roboflow)**: 2,242 images of lab gear; easy YOLO export, community models available.universe.

**Why These?**

Lab-generic like COCO lacks classes; these provide medical/lab relevance for precise detections → GenAI summaries (e.g., "Detected 2 centrifuges, no defects"). Download from Nature, Roboflow Universe, or Kaggle; fine-tune YOLO then pipe to LLM

| Dataset | Images | Classes | Strengths for Docs |
|---|---|---|---|
| Chemistry Lab | 4,599 | 25 lab tools | Real-world variety |
| Hospital Assets | Custom | Medical devices | Healthcare accuracy |
| LabEquipVis | 2,584 | Lab equipment | High-res annotations |

**Why ChemEq25 Dataset?**

- **FAIR Alignment**: Findable (Mendeley DOI), Accessible (free download), Interoperable (YOLO annotations, 640x640 JPG), Reusable (CC-BY, provenance from 3 labs, 70/20/10 splits).
- **Relevance**: Real chemical/medical labs (Bangladesh Atomic Energy Lab); diverse conditions for robust documentation generation.
- Download: data.mendeley.com/datasets/zptphkynt6; GitHub benchmarks.

**Next Actions**

1. Download ChemEq25; setup Colab (Ultralytics pip).
2. Train YOLO: yolo detect train data=chemeq25.yaml.
3. GenAI: Prompt LLM with detections for FAIR reports (JSON-LD metadata).
4. Document FAIR compliance in notebook for thesis/paper.

**YOLO (You Only Look Once) detection with Vision-Language Models (VLMs)**

Combining YOLO (You Only Look Once) detection with Vision-Language Models (VLMs) creates a powerful, hybrid AI pipeline for automated report generation, auditing, and visual analysis. YOLO provides fast, precise localization of objects, while VLMs provide context, reasoning, and textual description. This combination is particularly effective for generating detailed reports in fields such as infrastructure inspection, medical diagnostics, and surveillance.

**Report Generation Workflow**

YOLO can crop relevant areas for analysis by a VLM, resulting in detailed text. The process includes:

1. **Object Detection (YOLO):** YOLO identifies and locates specific items and crops these regions.

2. **Context Extraction (VLM):** The cropped image and YOLO class label are input into a VLM (e.g., LLaVA, GPT-4o, Gemini).
3. **Prompting & Analysis:** A structured prompt, such as "Describe this damage in detail," is given to the VLM to generate a description of the cropped area.
4. **Reporting:** The VLM output and location data from YOLO produce a final, structured report (e.g., CSV, JSON, or PDF).

**Advantages**

- **High Precision and Context:** YOLO ensures that small or obscured objects are not missed, while VLMs provide context, such as the severity of a crack.
- **Reduced Errors:** Using YOLO crops to focus the VLM reduces the likelihood of the VLM misinterpreting or imagining objects, particularly in complex scenes.
- **Open-Vocabulary Capability:** Integrating VLMs allows the system to detect and describe new, unseen categories using textual queries (e.g., "Find rusted components").
- **Low-Data Training:** Few-shot learning can be used with VLMs, enabling the system to adapt to new report types with few examples.

**Applications**

- **Infrastructure Inspection:** Combining YOLOv11x with VLMs like Qwen2.5-VL can automatically detect, classify, and describe pavement cracks and maintenance needs in inspection reports.
- **Medical Imaging:** YOLO can detect regions of interest (e.g., polyps) and VLMs like GPT-4o can analyze them, allowing for automated report generation.

**ChemEq25** is a specialized object detection dataset with 4,599 annotated images of chemical/biochemical laboratory equipment, captured from real labs in Bangladesh (Bangladesh Atomic Energy Chemical Lab, UIU Biochemical Lab, Dhaka Imperial College).

**Dataset Details**

- **Images**: 4,599 JPGs (640×640 pixels, multi-angle, varying lighting/backgrounds/occlusions)
- **Classes**: 25 common lab tools (beaker, conical flask, funnel, glass rod, pipette, centrifuge, etc.)
- **Splits**: Train 70% (3,219), Val 20% (920), Test 10% (460)
- **Format**: YOLO-compatible bounding boxes; perfect for documentation pipelines
- **Source**: Mendeley Data (DOI: data.mendeley.com/datasets/zptphkynt6); FAIR-aligned with detailed provenance

**Perfect PhD Fit**

This dataset directly supports your "GenAI Medical Lab Documentation" thesis:

text

1. YOLOv8 detection → identifies "centrifuge_model_XYZ"

2. VLM (LLaVA-Med) → generates "Equipment operational, no visible defects"

3. FAIR metadata → JSON-LD for reproducibility

**Download**: data.mendeley.com/datasets/zptphkynt6/4 → unzip → yolo detect train data=chemeq25.yaml

**Why superior**: Real lab conditions > synthetic; 25 classes cover medical/chemistry overlap; published benchmarks (YOLOv11 mAP@50=0.987).

How To Load The Medical Lab Equipments As A Image In Yolov8?

To load and run YOLOv8 inference on medical lab equipment images (e.g., from ChemEq25 dataset), use Ultralytics' predict mode after training or with pretrained/custom weights. This detects equipment like centrifuges in new images for documentation generation.data.mendeley+1

Step 1: Setup and Download

Install: pip install ultralytics opencv-python.
Download ChemEq25 images (JPGs) from data.mendeley.com/datasets/zptphkynt6/4—extract to /path/to/chemeq25/images/.

Step 2: Load Trained Model

Train first (if not pretrained):

text

yolo detect train data=chemeq25.yaml model=yolov8n.pt epochs=50

Model saves to runs/detect/train/weights/best.pt.

Step 3: Predict on Lab Images

Single Image (local/URL):

python

```
from ultralytics import YOLO
import cv2
# Load model
model = YOLO('runs/detect/train/weights/best.pt')  # or 'yolov8n.pt' pretrained
# Load image
img_path = '/path/to/lab_equipment.jpg'  # e.g., centrifuge photo
results = model(img_path, conf=0.5, save=True)  # conf threshold, save output
```

```
# Process results
for r in results:
    boxes = r.boxes  # Detected equipment
    print(r.names[int(c)])  # e.g., 'centrifuge' class 4
    annotated_img = r.plot()  # Visualize
    cv2.imwrite('detected_lab.jpg', annotated_img)
```

CLI: yolo detect predict model=best.pt source=lab_equipment.jpg

Step 4: Batch Lab Folder

text

results = model('/path/to/chemeq25/images/', save=True, project='lab_docs')

Generates detections for all images → feed classes/conf to GenAI for summaries (e.g., "5 centrifuges detected, all operational").

Tips

- Sources: Local JPG/PNG, webcam (0), video, folder.
- Output: Boxes with class/conf; extract via r.boxes.cls, r.boxes.conf.
- FAIR: Log metadata (image ID, detections JSON).

To make your YOLOv8 medical lab equipment detection → GenAI documentation pipeline FAIR-compliant (Findable, Accessible, Interoperable, Reusable), embed principles at each step for PhD reproducibility and publication. This ensures your ChemEq25-based work meets academic standards like NeurIPS datasets track.

FAIR Implementation by Process Step

Findable (F1-F4)

- Dataset: Use ChemEq25 DOI (data.mendeley.com/datasets/zptphkynt6); cite in data.yaml: url: https://doi.org/10.17632/zptphkynt6.4.
- Model/Code: GitHub repo with DOI (Zenodo); README links weights/metadata.
- Outputs: Upload detections/reports to Figshare/Zenodo with PID.

Accessible (A1-A2)

- Data: Open download (no login); support Wget/cURL.
- Model: pip install ultralytics; yolo predict model=gs://your-bucket/best.pt.
- API: Streamlit/Gradio demo (Hugging Face Spaces) for image → report.

Interoperable (I1-I3)

- Formats: YOLO annotations (COCO JSON export), reports in JSON-LD/RDF.

- Vocab: Use schema.org (e.g., "MedicalDevice" for centrifuge), ONNX for model.
- Code: Docker: FROM ultralytics/ultralytics:latest; COPY best.pt /model.pt.

Reusable (R1-R1.3)

- Licenses: CC-BY-4.0 for data/code; document provenance (ChemEq25 + your images).
- Metadata: YAML/JSON with splits, mAP (0.98), hyperparameters (epochs=50, imgsz=640).
- README: Usage, eval metrics (mAP@50, ROUGE for reports), citation.

Example FAIR data.yaml

text

```
# ChemEq25 Medical Lab Equipment (FAIR)
path: ./chemeq25
train: train/images
val: valid/images
nc: 25
names: {0: 'beaker', 4: 'centrifuge'}  # Full list
url: https://doi.org/10.17632/zptphkynt6.4
license: CC-BY-4.0
metrics: mAP@50: 0.987 [web:75]
```

Validation Tools

- F-UJI: Score FAIRness online.
- FAIR GPT: ChatGPT evaluator for metadata.
- GitHub FAIRscaper: Auto-check repo.

Follow-ups

How to make YOLO datasets Findable with persistent IDs

Best repositories for FAIR medical lab image datasets

Tools to assess FAIR compliance for custom datasets

Interoperability standards for lab equipment image

**SIMPLE PROCESS TO EXECUTE IN YOLO AS STEP BY STEP PROCESS**

Here's the simplest end-to-end YOLOv8 process you can follow, especially for your lab-equipment work.

1. Install and Quick Test

1. Install:

bash

pip install ultralytics

2. Test with a sample image:

python

```
from ultralytics import YOLO


model = YOLO("yolov8n.pt")          # pretrained COCO model
results = model("lab.jpg", save=True)
```

This should create a runs/detect/predict/ folder with an output image.ultralytics+1

2. Prepare a Simple Dataset

1. Create folders:

text

```
lab_dataset/
  images/
    img1.jpg
    img2.jpg
  labels/
    img1.txt
    img2.txt
```

2. Each .txt file (YOLO format) per image:

text

class_id x_center y_center width height

All values are normalized (0–1).roboflow+1

3. Create data.yaml

In the same folder:

text

path: lab_dataset

train: images

val: images

nc: 1

names: ['centrifuge']  # or ['centrifuge','microscope',...]

4. Train (One Simple Command)

bash

yolo detect train model=yolov8n.pt data=lab_dataset/data.yaml epochs=30 imgsz=640

YOLO will create runs/detect/train/weights/best.pt.learnopencv+1

5. Run on New Lab Images

bash

yolo detect predict model=runs/detect/train/weights/best.pt source="new_lab_images/"

All annotated outputs go to runs/detect/predict2/, ready for documentation generation.

How To Work In Bash And Python?

Here's how to work with BASH (CLI) vs PYTHON (API) for YOLOv8—both do the same thing, choose based on preference.

BASH (Terminal Commands) - Simpler for Quick Runs

1. Install

bash

pip install ultralytics

2. Test Predict

bash

yolo predict model=yolov8n.pt source=lab.jpg

→ Output in runs/detect/predict/lab.jpg

3. Train

bash

yolo detect train data=lab_dataset/data.yaml model=yolov8n.pt epochs=30

→ Model in runs/detect/train/weights/best.pt

4. Predict with Custom Model

bash

yolo detect predict model=runs/detect/train/weights/best.pt source=new_lab.jpg

PYTHON (Scripts/Notebooks) - Flexible for Customization

1. Basic Script (run_yolo.py)

python

from ultralytics import YOLO

*# Load model*

model = YOLO("yolov8n.pt")  *# or "path/to/best.pt"*

*# Predict*

results = model("lab.jpg", save=True, conf=0.5)

print(results[0].names)  *# Classes detected*

Run: python run_yolo.pypython.plainenglish+1

2. Train in Python

python

model = YOLO("yolov8n.pt")

results = model.train(data="lab_dataset/data.yaml", epochs=30, imgsz=640)

3. Batch Process Lab Folder

python

results = model("/path/to/lab_images/", save=True)

for r in results:

    print(r.boxes.cls)  *# Detected equipment IDs*

Quick Comparison

| Task | BASH | Python |
|---|---|---|
| Test | yolo predict model=yolov8n.pt source=img.jpg | model("img.jpg") |
| Train | yolo detect train data=data.yaml | model.train(data="data.yaml") |
| Batch | yolo predict source=folder/ | model("folder/") |

Recommendation: Start with BASH (copy-paste commands), graduate to Python for FAIR metadata/GenAI integration.

How To Work In Visual Studio Web Project Python Using Yolo Dataset.Give Me Step By Step Process For Medical Lab Images And Text Using Genai

Here's the complete step-by-step process to build a Visual Studio Code web project with YOLOv8 + GenAI for medical lab equipment documentation (upload image → detect centrifuge/microscope → generate report).

Step 1: VS Code Setup

1. Open VS Code → New Folder → lab-doc-web
2. Install Python extension (Microsoft)
3. Open terminal: `Ctrl+`` (backtick)

Step 2: Create Project Structure

text

lab-doc-web/

├── app.py # Flask backend

├── model.pt # Your trained YOLO (from ChemEq25)

├── templates/

│ └── index.html # Frontend

├── static/

│ └── style.css # Optional styling

└── requirements.txt

Step 3: requirements.txt

text

flask==3.0.0

ultralytics

openai

pillow

torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118

Install: pip install -r requirements.txtYouTube

Step 4: Backend - app.py (YOLO + GenAI)

python

```
from flask import Flask, render_template, request, jsonify
from ultralytics import YOLO
from PIL import Image
from openai import OpenAI
import io

app = Flask(__name__)
model = YOLO('model.pt')  # Replace with your trained model
client = OpenAI(api_key='your-openai-key')  # Get free key at openai.com
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/analyze', methods=['POST'])
def analyze():
    file = request.files['image']
    img = Image.open(file.stream)
```

```
    # Step 1: YOLO Detection
    results = model(img, conf=0.5, verbose=False)
    detections = [results[0].names[int(cls)] for cls in results[0].boxes.cls]

    # Step 2: GenAI Documentation
    prompt = f"Medical lab image detected: {detections}. Generate professional equipment inventory/maintenance report."
    response = client.chat.completions.create(
        model="gpt-4o-mini",
        messages=[{"role": "user", "content": prompt}]
    )
   return jsonify({
        'detections': detections,
        'report': response.choices[0].message.content
    })

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5000)
```

Save in VS Code → F5 to run (or python app.py)henrynavarroYouTube

Step 5: Frontend - templates/index.html

xml

```
<!DOCTYPE html>
<html>
<head>
    <title>Medical Lab Equipment Documentation</title>
    <style>
        body { font-family: Arial; max-width: 800px; margin: auto; padding: 20px; }
        #preview { max-width: 400px; max-height: 300px; }
        button { background: #007bff; color: white; padding: 10px 20px; border: none; cursor: pointer; }
```

```
    </style>
</head>
<body>
    <h1>🧪 Medical Lab Equipment Analyzer</h1>
    <input type="file" id="imageInput" accept="image/*">
    <br><br>
    <button onclick="analyze()">📊 Generate Report</button>
    <br><br>
    <img id="preview">
    <div id="results"></div>

    <script>
        document.getElementById('imageInput').addEventListener('change', function(e) {
            document.getElementById('preview').src = URL.createObjectURL(e.target.files[0]);
        });

        async function analyze() {
            const file = document.getElementById('imageInput').files[0];
            if (!file) return alert('Upload image first!');

            const formData = new FormData();
            formData.append('image', file);

            document.getElementById('results').innerHTML = 'Analyzing...';

            try {
                const response = await fetch('/analyze', { method: 'POST', body: formData });
                const data = await response.json();
```

```
            document.getElementById('results').innerHTML = `
                <h3>✅ Detected Equipment:</h3>
                <ul>${data.detections.map(d => `<li>${d}</li>`).join('')}</ul>
                <h3>📄 AI Documentation:</h3>
                <prestyle="background:#f5f5f5;padding:15px;border-radius:5px">${data.report}</pre>
            `;
        } catch(e) {
            document.getElementById('results').innerHTML = 'Error: ' + e;
        }
    }
  </script>
</body>
</html>
```

Step 6: Run Your Web App

1. Train YOLO first (ChemEq25 dataset → model.pt)
2. Add model.pt to project folder
3. Terminal: python app.py
4. Browser: http://localhost:5000
5. Upload lab image → Instant detection + report!

Step 7: VS Code Shortcuts

- Ctrl+Shift+P → Python: Select Interpreter
- F5 → Debug Flask app
- Ctrl+F5 → Run without debugging
- Live Server extension for HTML preview

Deploy (Optional)

bash

```
pip install gunicorn
gunicorn --bind 0.0.0.0:5000 app:app
```

List Of Dataset Available In Gen Ai For Multimodal Learning

Here's a comprehensive list of top multimodal datasets for GenAI (vision-language-audio-video) learning, perfect for your PhD in medical lab equipment documentation. These combine images/text (VLMs like LLaVA-Med) or more modalities.

Vision-Language (Image + Text)

| Dataset | Size | Key Features | Best For |
|---|---|---|---|
| LAION-5B roboflow | 5.85B image-text pairs | Web-scraped, CLIP-filtered, multilingual | Pretraining Stable Diffusion, CLIP |
| CC12M | 12M image-caption | Common Crawl, cleaned | Captioning, VQA |
| COCO Captions | 330K images, 1.5M captions | Object detection + captions | Detection + description |
| Flickr30k | 31K images, 158K captions | Dense descriptions | Retrieval, captioning Memory |
| Visual Genome | 108K images, 3.8M objects | Region descriptions, relationships | Visual reasoning, grounding roboflow |
| RF100-VL | 164K images | 100 diverse domains (medical incl.) | VLM benchmarking roboflow |

Audio-Language (Speech + Text)

| Dataset | Size | Key Features | Best For |
|---|---|---|---|
| AudioCaps | 46K clips + captions | Audio descriptions | Audio captioning Memory |
| VoxCeleb2 | 1M+ utterances | Speaker ID + video | Speaker recognition encord |

Video-Language (Video + Text)

| Dataset | Size | Key Features | Best For |
|---|---|---|---|
| InternVid | 7M videos, 234M clips | Instructional videos | Video-text alignment labellerr |
| HowTo100M | 1.2M videos | YouTube tutorials + subtitles | Video understanding innovatiana |
| MovieQA | Movie clips + QA | Plot/dialogue understanding | Video QA innovatiana |

Medical/Multimodal Specialized

- MIMIC-CXR: 377K chest X-rays + reports (ideal for lab extension)
- PadChest: Chest X-rays + multilingual reports
- ChemEq25: Your lab equipment (images + potential captions)

BASH

```
# HuggingFace
pip install datasets
from datasets import load_dataset
ds = load_dataset("laion/laion-aesthetics")  # LAION subset
# COCO
```

List Of Dataset Available In Gen Ai For Multimodal Learning Using Medical Image Processing And Medical Lab Equipments

Here's a curated list of multimodal GenAI datasets specifically for medical image processing + lab equipment, ideal for your PhD thesis (vision-language for radiology reports, equipment detection + documentation). These pair images with text reports/descriptions for VLMs like LLaVA-Med or custom YOLO+GenAI pipelines.pmc.ncbi.nlm.nih+1

| Dataset | Modalities | Size | Key Features | Download |
|---|---|---|---|---|
| MIMIC-CXR | Chest X-rays + reports | 377K images, 227K reports | Free-text radiology impressions/findings | PhysioNet |
| PadChest | Chest X-rays + reports | 160K images, multilingual labels | Spanish/English reports, 174 findings | BIC/Rotating |
| IU-Xray | Chest X-rays + reports | 7K image-report pairs | Indication/impression sections | NIH |
| OpenI | Multi-modality (CT/MRI/X-ray) + reports | 3.6K studies | Diverse anatomy | Indiana Network |
| CheXpert | Chest X-rays + labels/text | 224K images | Uncertainty labels + reports | Stanford |

Multimodal Medical Equipment/Lab (Detection + Reports)

| Dataset | Modalities | Size | Key Features | Download |
|---|---|---|---|---|
| ChemEq25 | Lab equipment images + annotations | 4,599 images, 25 classes | Centrifuge, microscope, beakers (extend with reports) | Mendeley DOI |
| Hospital Assets | Medical devices (YOLO) | Custom (16 classes) | Dialysis pumps, lab tools | Paper PDF |
| MINIM Dataset | Multi-modality synthetic (OCT, fundus, CXR/CT) + text | Diverse pairs | GenAI-generated for training VLMs | |
| MedYOLO | 3D medical objects (BRaTS, LIDC, abdominal CT) | Multi-dataset | Organs/lesions detection + reports | |

Video/Audio-Medical Multimodal

| Dataset | Modalities | Size | Key Features | Download |
|---|---|---|---|---|
| Observer Dataset | Video/audio + transcripts (clinician-patient) | Anonymized interactions | De-identified multimodal clinical data | |
| UP-Fall | IMU + thermal + radar (falls in labs) | Multisensor falls | Lab safety/equipment monitoring | |

datas are lab equipment images and text GIVE THE Dataset Size

Inter-annotator Agreement

Data Quality Score

FAIR Compliance

No public datasets exist with paired lab equipment images + text descriptions/reports

Current Lab Equipment Datasets (Image-Only)

| Dataset | Size | Inter-annotator Agreement | Data Quality Score | FAIR Compliance |
|---|---|---|---|---|
| ChemEq25 | 4,599 images (25 classes) | N/R (single annotator) | YOLOv11 mAP@50=0.987 | High (Mendeley DOI 10.17632/zptphkynt6.4) |
| Hospital Assets | Custom (~1K images, 16 classes) | N/R | YOLOv8 P=94%, R=92% | Low (paper PDF only) |
| LabEquipVis | 2,584 images | N/R | Good (computer lab gear) | Medium (ScienceDirect/Kaggle) |

Medical Imaging (Image + Text—Adaptable)

| Dataset | Size | IAA | Quality | FAIR |
|---|---|---|---|---|
| MIMIC-CXR | 377K images + 227K reports | $\kappa$=0.39-0.77 (subset) | F1=0.30-0.89 | High (PhysioNet DOI) |
| PadChest | 160K images + reports | F1=0.93 (RNN labels) | Macro-F1=0.93 | High (DOI 10.1016/j.is.2020.101716) |

Create "MedLabText"

ChemEq25 images + synthetic reports (GPT-4o-mini)

Size: 4,599 pairs

IAA: Multi-rater validation (2 annotators)

Quality: ROUGE/BLEU vs. expert reports

FAIR: Zenodo DOI + GitHub

Why Gap Exists: Lab equipment lacks radiology-style reporting infrastructure.

Quick Prototype: # Generate text for ChemEq25

prompt = "Describe this lab equipment image: centrifuge detected."

# → "Model XYZ centrifuge, operational, calibrated 2026-03-20"

Benchmark: LLaVA-Med fine-tune → first multimodal lab equipment dataset!

Action: Download ChemEq25 → generate 1K report pairs → Zenodo upload (Week 2 milestone).

give the datas of lab equipment images and text also GIVE THE Dataset Size Inter-annotator Agreement Data Quality Score FAIR Compliance in laboratory field using yolo

Current Metrics (Text-Only Catalog)

| Metric | Value | Notes |
|---|---|---|
| Dataset Size | ~200 equipment entries (~100 unique items) | Covers stopwatches, slides, cover glass, haemometers, centrifuges, pipettes, microscopes, etc. sample-dataset.xlsx |
| Inter-annotator Agreement | N/A | Single source (inventory list); needs multi-rater validation for PhD |
| Data Quality Score | High (detailed specs: models, sizes, brands like Racer, Blue Star, Rohem) | Complete descriptions (thickness, pkt sizes); minor typos |
| FAIR Compliance | Low (raw Excel) | Upgrade: Zenodo DOI + JSON export + image pairing |

Transform to Multimodal YOLO Dataset

Excel (text) + Lab Images (ChemEq25) = MedLabMulti (images + descriptions)

| Target Metric | Planned Value | How |
|---|---|---|
| Size | 4,599 images + 200 text pairs | ChemEq25 images + your Excel descriptions |
| IAA | $\kappa$>0.8 (2 raters) | Validate 20% descriptions |

| Target Metric | Planned Value | How |
|---|---|---|
| Quality | mAP>0.95, BLEU>0.3 | YOLO detection + caption similarity |
| FAIR | High | DOI via Zenodo, COCO JSON, CC-BY |

As a beginner, working with YOLO datasets for medical laboratory equipment (e.g., microscopes, centrifuges, pipettes) involves collecting images, labeling, and training YOLOv8. Studies show 90%+ accuracy on hospital assets like CT scanners and stethoscopes.umpir.ump+1

**Step 1: Collect Images (200-500/class)**

Capture diverse photos in lab lighting/angles.

- Smartphone camera: 10-20 images per equipment type (e.g., microscope, test tubes).
- Sources: Public datasets (Roboflow Universe: "medical equipment"), Google Images (download ethically).
- Augment: Rotate, flip, brightness via Roboflow (free tier).
- Classes example: microscope, pipette, centrifuge, petri_dish, scale.

**Step 2: Label Dataset (No-Code Beginner Way)**

Use free tools for bounding boxes.

1. Upload images to **Roboflow** (roboflow.com): Create free account, new project > Object Detection.
2. Label: Draw boxes around equipment, assign class (e.g., "microscope").
3. Generate version: Auto-split 80/10/10 train/val/test; augment for 3-5x data.
4. Export: YOLOv8 format (ZIP with images/labels/data.yaml).

Alternative: LabelImg desktop tool (pip install labelImg).

**Step 3: Setup Environment (Google Colab Free)**

No local GPU needed.

1. Open Colab: colab.research.google.com.
2. Upload ZIP from Roboflow.
3. Install: !pip install ultralytics.
4. Train:

python

```
from ultralytics import YOLO
model = YOLO('yolov8n.pt')  # Beginner nano model
model.train(data='data.yaml', epochs=50, imgsz=640, batch=16)
```

5. Results: runs/detect/train/weights/best.pt (90%+ mAP possible with 3K+ images).nptel+1

**Step 4: Test & Visualize**

python

```
model = YOLO('best.pt')
results = model('test_lab.jpg')  # Draws boxes
results[0].show()
```

Metrics: Precision 94%, Recall 92% on 16 med equipment classes.

**Beginner Tips**

- Start small: 3 classes, 200 images → scale up.
- Hardware: Colab T4 GPU free; local needs NVIDIA.
- Issues: Low accuracy? Add more varied images/augment.
- Deploy: Export ONNX for web/app integration.

Example datasets exist for hospital tools (94% precision). What equipment classes are you targeting?