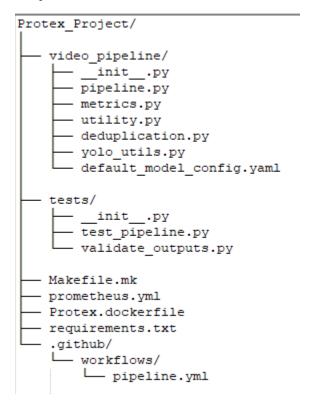
Project Structure Overview



Gihub Link: https://github.com/deepyad/ProtexAl

Module-by-Module Explanation

1. video_pipeline/pipeline.py

- Purpose: The main pipeline script.
- Responsibilities:
 - o Parses command-line arguments.
 - o Loads configuration and detection model.
 - o Reads video, deduplicates frames, runs detection, saves images.
 - o Tracks metrics (frame counts, timings, drops, detections).
 - o Generates a Markdown report.
 - Exposes Prometheus metrics.

Entrypoint:

python -m video_pipeline.pipeline --input-video <video> --output-dir <dir> --model-config <yaml>

2. video_pipeline/yolo_utils.py

- Purpose: Model loading and object detection logic.
- Responsibilities:
 - o Loads YOLOv8 model based on config.
 - Runs inference and formats detection results.

3. video_pipeline/deduplication.py

- Purpose: Frame deduplication.
- Responsibilities:
 - Uses perceptual hashing to remove near-duplicate frames before further processing.

4. video_pipeline/metrics.py

- Purpose: Metrics tracking and Prometheus integration.
- Responsibilities:
 - o Defines counters, histograms, and gauges for pipeline observability.
 - o Singleton pattern for global access.

5. video_pipeline/utility.py

- Purpose: General utility functions.
- Responsibilities:
 - o Generates Markdown/HTML reports.
 - o Calculates deduplication and detection metrics.
 - o Other helpers as needed.

6. video_pipeline/default_model_config.yaml

- Purpose: Default configuration for the pipeline and detection model.
- Responsibilities:
 - o Specifies model weights, confidence threshold, deduplication threshold, etc.

7. tests/test_pipeline.py

- Purpose: Automated test suite (pytest).
- Responsibilities:
 - o Integration tests for pipeline execution and outputs.
 - Unit tests for deduplication, detection, and metrics.

COCO annotation validation and error handling check

8. tests/validate_outputs.py

- Purpose: Output validation script.
- Responsibilities:
 - o Checks existence and integrity of output images and COCO annotation files.
 - Validates COCO structure and cross-references.

9. Makefile.mk

- Purpose: Automation for build, run, test, lint, and clean tasks.
- Responsibilities:
 - o Provides easy CLI for common developer and CI/CD tasks.

10. Protex.dockerfile

- Purpose: Docker containerization.
- Responsibilities:
 - o Installs dependencies.
 - Sets up the pipeline for reproducible, portable execution.

11. prometheus.yml

- Purpose: Prometheus monitoring configuration.
- Responsibilities:
 - Scrapes metrics from the pipeline container for observability.

12. .github/workflows/pipeline.yml

- Purpose: CI/CD automation (GitHub Actions).
- Responsibilities:
 - o Runs builds, tests, and validation on every push or PR.

How to Run the Pipeline

1. Build the Docker Image

docker build -f Protex.dockerfile -t video-pipeline:latest.

2. Run the Pipeline

docker run --rm -p 8000:8000

- -v /path/to/input:/input
- -v /path/to/output:/output

video-pipeline:latest

- --input-video /input/sample.mp4
- --output-dir /output
- --model-config /app/default_model_config.yaml

3. Run Tests

From project root

PYTHONPATH=./video_pipeline pytest tests/

or use Makefile:

make test

4. Validate Outputs

python tests/validate_outputs.py/path/to/output

5. View Prometheus Metrics

• Visit http://localhost:8000/metrics after running the container.