**PeepWorkflow Documentation**

**Description**

The PeepWorkflow class automates the process of loading, preprocessing, and generating eigenfaces from facial images, either from a folder or a DataFrame. It includes methods for adding Laplace noise (currently a placeholder) and prepares data for machine learning. This workflow is designed for facial recognition and privacy protection via eigenfaces.

**Usage**

1. Installation

Install required libraries:

*pip install numpy pandas Pillow scikit-learn tqdm*

1. Import

*from src.modules.peep\_workflow import PeepWorkflow*

1. Data Preparation

Loading: Load face images from a folder or DataFrame. Folder: Images in a single folder, consistent naming. DataFrame: pandas DataFrame with 'userFaces' (PIL Images) and 'imageId'.

Resizing: Handled internally, IMAGE\_SIZE in src.config. Grayscale: Handled internally.

1. PeepWorkflow Object

workflow = PeepWorkflow(image\_folder="path/images", subject\_prefix="subject\_1") # Folder workflow = PeepWorkflow() # DataFrame

image\_folder: (Optional) Folder path. subject\_prefix: (Optional) Prefix to filter subjects.

1. Loading and Preprocessing

Folder:

*workflow.run\_from\_folder()*

DataFrame:

*workflow.run\_from\_dataframe(df) # df is the DataFrame.*

Eigenface Generation

Automatic in run\_from\_folder and run\_from\_dataframe.

1. Retrieving Results

Eigenfaces:

*eigenfaces = workflow.get\_eigenfaces() # List of PIL Image objects.*

Machine Learning Data (X, y):

*X, y = workflow.run\_from\_folder() # Or workflow.run\_from\_dataframe(df) X: NumPy array of flattened eigenfaces. y: NumPy array of subject numbers.*

**Tests**

Run unit tests (tests/test\_peep\_workflow.py):

1. Open a terminal.
2. Navigate to the project root directory.
3. Run the tests:

*python -m unittest discover tests or python -m unittest tests.test\_peep\_workflow*

Important: Use python -m unittest from project root. Don't run the file directly.

**Next Steps**

* Implementation of PEEP:

PEEP method with algorithms:

* + Laplacian (main one to implement)
  + Gaussian
  + Uniform
* Add Facial Recognition: Classifier (SVM, k-NN) on eigenfaces. Predict identity. Evaluate accuracy.
* Experiment with Epsilon: Balance privacy and accuracy. Higher epsilon: stronger privacy, reduced accuracy. Lower epsilon: weaker privacy, improved accuracy.