**Image Recognition System**

This project is an image recognition system using FastAPI and EfficientNetB7 for classifying and retrieving similar images based on a case base.

Features

- Training Endpoint: `/train/` - Upload a ZIP file containing images organised in folders to train or update the model.

- Search Endpoint: `/search\_similar\_images/` - Upload an image to find similar images from the case base.

Requirements

- Python 3.12 or higher

- Required Python packages (listed in `requirements.txt`)

Installation

**1. Clone the Repository**

```bash

git clone <repository-url>

cd <repository-directory>

```

**2. Create and Activate a Virtual Environment**

```bash

python -m venv .venv

source .venv/bin/activate # On Windows use: .venv\Scripts\activate

```

**3. Install Dependencies**

Create a `requirements.txt` file with the following content:

```

numpy

opencv-python

Pillow

tensorflow

scikit-learn

fastapi

uvicorn

```

Then, install the dependencies:

```bash

pip install -r requirements.txt

```

**Usage**

1. \*\*Start the Server\*\*

```bash

uvicorn main:app --reload

```

This will start the FastAPI server on `http://localhost:8000`.

2. \*\*Train the Model\*\*

- Prepare a ZIP file containing folders of images, where each folder name represents a label.

- Use the `/train/` endpoint to upload the ZIP file and train the model.

Example `curl` command:

```bash

curl -X POST "http://localhost:8000/train/" -F "zip\_file=@path/to/your\_dataset.zip"

```

3. \*\*Search for Similar Images\*\*

- Upload an image to the `/search\_similar\_images/` endpoint to find similar images from the case base.

Example `curl` command:

```bash

curl -X POST "http://localhost:8000/search\_similar\_images/" -F "file=@path/to/your\_image.jpg"

```

**Code Overview**

- \*\*`main.py`\*\*: Contains the FastAPI application with endpoints for training and searching.

- \*\*`create\_model()`\*\*: Defines and compiles the EfficientNetB7 model.

- \*\*`preprocess\_image()`\*\*: Preprocesses images for better feature extraction.

- \*\*`extract\_features()`\*\*: Extracts features from images using the trained model.

- \*\*`load\_case\_base()`\*\* and \*\*`save\_case\_base()`\*\*: Functions for handling the case base.

- \*\*`calculate\_similarity()`\*\* and \*\*`retrieve\_similar\_cases()`\*\*: Functions for finding similar images.

**Error Handling**

- \*\*CORS Errors\*\*: Ensure that requests are made from allowed origins and that the server is properly configured to handle CORS.

- \*\*File Errors\*\*: Make sure the ZIP file for training is correctly formatted and that image files are properly encoded.

**Logging**

- Logs are generated to track operations and errors. Check the console output for detailed logs.

**Contact**

For any questions or issues, please contact [yaohan.jobs@hotmail.com].