Image Similarity Finder

A tool with both GUI and command-line interfaces that finds visually similar images across directories, regardless of size, format, or minor modifications.

Features

- Find images similar to a reference image across multiple directories
- Works with different image sizes and aspect ratios
- Supports various image formats (JPG, PNG, BMP, TIFF, WebP, GIF)
- Adjustable similarity threshold for fine-tuning results
- Configurable number of results to display
- User-friendly graphical interface with image preview and context menus
- **Ability to cancel long-running operations**
- Right-click on results to open images or navigate to their folders
- Command-line interface for automation and scripting
- Type-safe implementation with Pydantic models
- Robust error handling and validation
- Modular architecture with clean separation of concerns
- Easy installation and uninstallation

Architecture

The application follows a modular architecture with clear separation of concerns:

- **models.py**: Data models and validation using Pydantic
- **analyzer.py**: Image analysis and feature extraction
- **finder.py**: Core functionality for finding similar images
- **gui.py**: Graphical user interface using Tkinter
- **cli.py**: Command-line interface
- **main.py**: Main entry point for the application

Installation

Option 1: Using pip (recommended)

```
pip install imagesim
```

Option 2: From source

```
git clone https://github.com/example/imagesim.git
cd imagesim
pip install -e .
```

Option 3: Using the install script (Linux/macOS)

```
./install.sh
```

Usage

Graphical User Interface

Launch the GUI with:

```
imagesim --gui
```

or simply:

```
imagesim
```

The GUI provides: - Visual image selection - Directory browsing - Adjustable threshold with slider - Results with similarity scores - Image preview - **Cancel button for stopping long-running operations** - Context menu for additional actions

Command-line Interface

Basic usage

```
imagesim path/to/reference_image.jpg path/to/search/directory
```

Search multiple directories

```
imagesim reference_image.jpg dir1 dir2 dir3
```

Adjust similarity threshold (0-1, where 1 is identical)

```
imagesim reference_image.jpg directory --threshold 0.6
```

Limit number of results

```
imagesim reference_image.jpg directory --max-results 5
```

How It Works

The tool uses computer vision techniques to find similar images:

1. **Feature Extraction**: Each image is converted into a feature vector using Histogram of Oriented Gradients (HOG)
2. **Normalization**: Feature vectors are normalized to ensure consistent comparison
3. **Similarity Calculation**: Cosine similarity measures how similar the vectors are
4. **Result Ranking**: Images are ranked by similarity score and returned in descending order

Development

Prerequisites

- Python 3.7 or higher
- pip (Python package manager)

Setup development environment

```
# Clone the repository
git clone https://github.com/example/imagesim.git
cd imagesim

# Create a virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install development dependencies
pip install -e ".[dev]"
```

Running tests

```
pytest
```

Requirements

- Python 3.7+
- Required Python packages (automatically installed):
  - numpy: For numerical operations
  - pillow: For image processing
  - opencv-python: For computer vision algorithms
  - scikit-learn: For similarity calculations
  - tkinter: For the graphical user interface
  - pydantic: For data validation and modeling

Troubleshooting

Common Issues

1. **File not found errors**: Ensure the paths to images and directories exist and are accessible.
2. **Pydantic validation errors**: Make sure file paths and directory paths exist before running searches.
3. **Missing dependencies**: If you encounter import errors, ensure all required packages are installed:

   ```
   pip install numpy pillow opencv-python scikit-learn pydantic
   ```

4. **GUI not displaying**: Ensure Tkinter is properly installed with your Python distribution.
5. **Low similarity scores**: Try adjusting the threshold parameter to find more matches.
6. **Search taking too long**: For large directories with many images, use the Cancel button to stop the search prematurely and adjust your search parameters.

Submitting Bug Reports

If you encounter an issue not covered here, please submit a bug report with: - A detailed description of the problem - The error message and stack trace - Steps to reproduce the issue - Your environment information (OS, Python version)

License

MIT License

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

1. Fork the repository
2. Create your feature branch (`git checkout -b feature/amazing-feature`)
3. Commit your changes (`git commit -m 'Add some amazing feature'`)
4. Push to the branch (`git push origin feature/amazing-feature`)
5. Open a Pull Request