Hanoi University of Science and Technology
School of Information and Communication Technology

---ꕥ📖ꕥ---

REPORT

Topic: Image Labeling Tool

Instructor: Ph.D. Dinh Thi Ha Ly
Student: Tran Quoc Khanh
Course: IT3930E – Project II

1. Introduction

In the ever-expanding landscape of data-driven applications, the accurate and efficient labeling of image data remains a critical task. To address this need, we present a user-friendly software product designed to streamline the data labeling process for image data labelers using the Windows operating system. The project aims to provide a simple yet effective solution for managing image datasets and facilitating the creation of labeled datasets for training machine learning models, conducting research, or other image-related tasks.

1.1. Project Overview

Our software, titled "Image Labeling Tool," is a Windows Form Application developed using the .NET Framework and created through the Visual Studio 2022 Integrated Development Environment (IDE). The primary goal is to enable users to easily label and organize their image datasets, improving their productivity and efficiency during data labeling tasks. All source code is available in khanha2k46pbc/project2 (github.com).

1.2. Targe Audience

The intended audience for this software comprises image data labelers, researchers, and machine learning practitioners who work with image datasets on Windows operating systems. By offering a user-friendly and interactive interface, the software caters to users with varying levels of technical expertise, making it accessible to both beginners and experienced data labelers alike.

1.3. Features

The software offers two primary modes of operation: "Move Images" and "Draw Bounding Box" (currently in development). In the "Move Images" mode, users can organize their images into labeled folders using trigger keys, allowing swift movement between images and seamless assignment to specific categories. The software's versatility allows users to add, edit, and delete labels according to their labeling requirements.

For the convenience of data labelers, the software provides an image editing feature, currently supporting the "swap color channel" functionality, which permits users to make simple image adjustments before saving the modified images back to the original dataset.

2. Requirements Analysis

2.1. Functional requirements

Functional requirements define the specific features and functionalities that the software must deliver to meet the needs of its users. In the context of the Image Labeling Tool, the following functional requirements have been identified:

Image Selection and Navigation

- The software shall allow users to select a root folder containing the image dataset for labeling.

- Users shall be able to navigate through the images in the selected dataset using either arrow keys or dedicated arrow buttons provided in the user interface.

Label Creation and Management

- The software shall provide functionality for users to create and manage labels that represent different categories or classes for the images.
- Users shall be able to add new labels, specifying the label name, trigger key (e.g., 'A' or '1') associated with the label, and the destination folder path where the images will be moved when the trigger key is pressed.

Image Movement and Labeling

- Users shall be able to assign images to specific labels by pressing the corresponding trigger key associated with each label.
- Upon pressing the trigger key, the software shall move the currently displayed image from the root folder to the destination folder associated with the chosen label.

Label Deletion

- The software shall allow users to delete existing labels, along with their associated destination folders.
- When a label is deleted, all images previously assigned to that label shall be moved back to the root folder.

Undo Functionality

- Users shall have the ability to undo the last image movement operation, effectively moving the most recently labeled image back to the root folder.
- The undo functionality should support multiple undo operations in sequence.

Image Editing

- The software shall provide a basic image editing feature that allows users to swap color channels in an image.
- Users shall be able to make color channel adjustments and save the edited image back to the root folder or a specific label's destination folder.

Refresh Data

- Users shall have the ability to refresh the software's view of the root folder and label information.
- The refresh functionality ensures that the software displays the latest changes made to the root folder and label configurations.

Mode Switching

- The software shall present users with the option to switch between the "Move Images" mode and the "Draw Bounding Box" mode (currently in development).

These functional requirements form the core capabilities of the Image Labeling Tool, enabling users to efficiently manage their image datasets and streamline the labeling process. By fulfilling these requirements, the software aims to enhance the productivity and accuracy of image data labelers on the Windows platform.

2.2. Non-functional requirements

In addition to the functional requirements that define the core features and functionalities of the Image Labeling Tool, non-functional requirements focus on the quality attributes and constraints that govern the software's behavior, performance, and usability. These requirements

ensure that the software operates effectively, efficiently, and provides a satisfactory user experience. The non-functional requirements for the Image Labeling Tool are as follows:

User Interface (UI) Design:

- The user interface shall be intuitive, user-friendly, and aesthetically pleasing to facilitate ease of use and enhance user productivity.
- The UI design shall adhere to standard Windows Form App design principles and guidelines, promoting consistency and familiarity for Windows OS users.

Performance:

- The software shall have minimal latency when loading and displaying images from the dataset to ensure smooth navigation between images.
- Image movement and file operations (e.g., moving, deleting, undoing) shall be performed swiftly to prevent any noticeable delays in the labeling process.

Responsiveness:

- The application shall respond promptly to user interactions, such as key presses and button clicks, ensuring a seamless and responsive user experience.

Reliability:

- The software shall handle exceptions and errors gracefully, providing meaningful error messages to users when unexpected situations occur.
- The application shall be stable and robust, preventing crashes or data loss during regular use.

Scalability:

- The software architecture shall be designed to accommodate potential future enhancements and additional features.
- The labeling tool should be capable of handling large image datasets without compromising performance.

Security:

- Access to the software and the labeled datasets shall be restricted to authorized users to maintain data integrity and prevent unauthorized access.
- User credentials and sensitive data (if any) shall be securely stored and transmitted, adhering to industry best practices for data security.

Compatibility:

- The Image Labeling Tool shall be compatible with various image formats commonly used in the field of image data labeling.
- The software should function smoothly on different versions of the Windows operating system, including Windows 10 and its subsequent versions.

Documentation:

- The software shall be well-documented, including a comprehensive user guide explaining its functionalities and how to use the tool effectively.
- The codebase shall be adequately documented to aid future maintenance and further development.

Accessibility:

- The application shall be designed to be accessible to users with disabilities, adhering to relevant accessibility guidelines and standards.

Usability Testing:

- The software shall undergo usability testing with a group of representative users to gather feedback and identify areas for improvement.

By addressing these non-functional requirements, the Image Labeling Tool aims to deliver a reliable, efficient, and user-friendly solution for image data labelers using Windows operating systems. The adherence to these requirements ensures a positive user experience and facilitates the seamless integration of the tool into the users' image labeling workflows.

2.3. Use cases

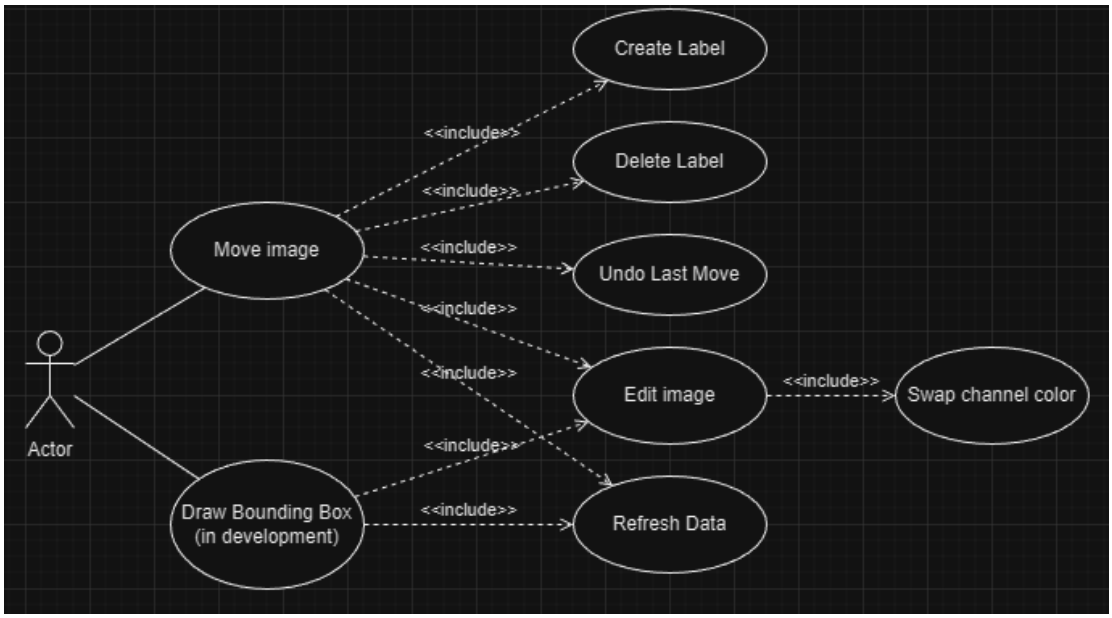

Figure 1: Use case diagram

3. System Design

3.1. Architectural overview

The Image Labeling Tool follows a layered architecture, which helps in organizing the software components and separates concerns to achieve modularity and maintainability. The architectural design of the application comprises three main layers:

Presentation Layer:

The Presentation Layer is responsible for handling the user interface (UI) components and user interactions. In this layer, the Windows Form App (.NET Framework) provides a graphical user interface that allows users to interact with the software seamlessly. The presentation layer facilitates user input and displays images, labels, and other relevant information to the users. It also receives user commands and forwards them to the appropriate components in the Business Logic Layer for processing.

Business Logic Layer:

The Business Logic Layer contains the core logic and functionality of the Image Labeling Tool. It acts as an intermediary between the Presentation Layer and the Data Access Layer. This layer is responsible for handling use cases, enforcing business rules, and orchestrating data flow and transformations. The Business Logic Layer processes user commands, manages the movement of images between folders, executes image editing operations, and maintains the label information. It ensures that the system behaves according to the defined functional requirements.

Data Access Layer:

The Data Access Layer deals with data storage and retrieval. In the case of the Image Labeling Tool, the data includes information about the image dataset, labels, and their corresponding destination folders. For simplicity, the application primarily utilizes the file system for data storage, using folder structures to organize the images based on the assigned labels. The Data Access Layer provides functionality to read and write data from/to the file system, enabling the application to persist and manage label configurations and image movements.

Architectural Advantages:

- **Modularity and Reusability**: The layered architecture promotes modularity, allowing individual layers to be developed, tested, and maintained independently. This modularity enhances reusability, making it easier to incorporate future enhancements or integrate additional features.
- **Separation of Concerns**: By dividing the application into layers, each layer focuses on specific tasks and responsibilities, leading to better code organization and maintainability.
- **Scalability**: The architecture can accommodate the addition of new features or extensions without significantly impacting existing components.
- **Ease of Testing**: With clear boundaries between layers, unit testing and integration testing can be performed more effectively.

Interaction Flow:

The user interacts with the graphical user interface (GUI) provided by the Presentation Layer, selecting modes, navigating through images, creating/deleting labels, and invoking other functionalities.

The Presentation Layer communicates user commands and inputs to the Business Logic Layer, where the core functionality is executed. The Business Logic Layer processes these commands and orchestrates the image movements, label management, and image editing operations.

When the Business Logic Layer needs to persist data, such as label configurations or image movements, it communicates with the Data Access Layer. The Data Access Layer reads and writes data to the file system, ensuring that changes are properly saved and reflected in the dataset.

The Data Access Layer provides the necessary information to the Business Logic Layer, which, in turn, updates the Presentation Layer to reflect changes in the GUI.

In summary, the layered architecture of the Image Labeling Tool fosters a structured and modular design, enabling smooth interactions between the user interface, core logic, and data storage components. This architecture provides a solid foundation for a reliable and user-friendly image labeling software product.

3.2. Description of major components/modules

The Image Labeling Tool comprises several major components or modules, each responsible for specific tasks and functionalities within the application. These modules work collaboratively to deliver a seamless image labeling experience to users. Below is a description of the major components/modules:

User Interface (UI) Module:

- This module is responsible for creating and managing the graphical user interface (GUI) presented to the users.
- It provides the main working window where images are displayed and users can interact with the software.
- The UI module includes components for navigation (arrow buttons and arrow keys) to move between images.
- It also offers interactive elements for adding, editing, and deleting labels, as well as the undo and refresh functionalities.

Label Management Module:

- The Label Management module handles the creation, deletion, and updating of labels within the Image Labeling Tool.
- It stores information about each label, including the label name, trigger key, and destination folder path.
- This module ensures that labels are properly managed, and their associations with images are maintained during the labeling process.

Image Navigation Module:

- The Image Navigation module enables users to navigate between images in the selected dataset.
- It receives user inputs (arrow keys or arrow buttons) from the UI module and determines the current image to display.
- This module ensures a smooth and responsive image navigation experience for users.

Image Movement Module:

- The Image Movement module handles the movement of images between the root folder and the destination folders associated with labels.
- It receives user inputs (trigger keys) from the UI module and manages the appropriate movement of the current image to the selected label's folder.
- This module also supports the undo functionality, allowing users to revert the last image movement operation.

Image Editing Module:

- The Image Editing module provides basic image editing functionality, such as color channel swapping.
- It allows users to modify images before saving them to the root folder or a specific label's destination folder.
- This module enhances the utility of the Image Labeling Tool by enabling users to make simple image adjustments.

Data Management Module:

- The Data Management module interacts with the Data Access Layer and is responsible for reading and writing data to the file system.
- It stores and retrieves label configurations, image movements, and other relevant information required for the proper functioning of the software.
- This module ensures that data is accurately saved and updated as users interact with the application.

Mode Selection Module:

- The Mode Selection module allows users to choose between the "Move Images" mode and the "Draw Bounding Box" mode (in development).
- It switches the application's behavior and user interface based on the selected mode.

Error Handling Module:

- The Error Handling module is responsible for handling exceptions and errors that may occur during the software's execution.
- It displays meaningful error messages to users, helping them understand the cause of any unexpected behavior.

File System Integration Module:

- The File System Integration module facilitates the integration of the Image Labeling Tool with the file system.
- It interacts with the Data Access Layer to read and write data to specific directories and manage the organization of images based on labels.
- These major components/modules work together to provide a cohesive and efficient image labeling experience for users. Each module is designed to address specific tasks, promoting code modularity, maintainability, and ease of future enhancements. The collaboration of these components ensures that the software meets its functional requirements and offers a user-friendly interface for image data labelers.

4. Implementation

In this section, we will discuss the implementation details of the Image Labeling Tool. The tool was designed as a Windows Form Application using the .NET Framework and developed through the Visual Studio 2022 IDE. The implementation comprises two main functionalities: "Move Images" and "Draw Bounding Box" (in development).

4.1. Move Images

The "Move Images" functionality allows users to label and organize images by moving them into corresponding folders based on user-defined labels. The following components and modules were implemented to achieve this functionality:

4.1.1. ModeSelectionForm.cs

The ModeSelectionForm serves as the entry point of the application. It allows users to choose between "Move Images" and "Draw Bounding Box" modes. Upon selecting "Move Images," the application transitions to the MoveImagesForm.

4.1.2. MoveImagesForm.cs

The MoveImagesForm is the main working window of the "Move Images" functionality. It allows users to navigate between images, add labels, and perform various image management operations. The major components and modules within this form are as follows:

Label Management Module (Label.cs):

- The Label class represents a label entity, including its name, trigger key, and associated folder path.
- The Label class also includes a refresh() method, which scans the folder path to update the list of image paths for each label.

RootFolder.cs:

- The RootFolder class handles the management of the root folder containing the images to be labeled.
- It includes a refresh() method that scans the root folder to update the list of image paths based on the user's selection to label images in all subdirectories or just the top directory.

AddLabelForm.cs:

- The AddLabelForm is a modal dialog that allows users to add new labels to the application.
- It verifies user inputs, checks for duplicate labels, and ensures the validity of the folder path before creating a new label.

ColorChannelSwap.cs:

- The ColorChannelSwap class provides the functionality to edit and save images by swapping their color channels.
- It offers five color channel swap options (RGB, GRB, GBR, BRG, and BGR) and allows users to save the edited image to a new file.

Transaction.cs:

- The Transaction class represents a single image movement transaction, recording the source and destination paths when an image is moved to a labeled folder.

4.1.3. User Interface (UI)

The User Interface of the "Move Images" functionality was designed using Windows Forms in Visual Studio. It includes the following key components:

- Navigation Buttons: Users can navigate between images using the arrow buttons or arrow keys.
- Label List: The UI displays a list of available labels, including their names and corresponding trigger keys.
- Add Label Button: Users can click this button to open the AddLabelForm and create new labels.
- Refresh Button: This button updates the information in the root directory and labels.
- Image Display: The main area where images are shown for labeling.

4.2. Draw Bounding Box (In Development)

The "Draw Bounding Box" functionality is still under development and will enable users to label objects within images by drawing bounding boxes around them. This functionality is intended to improve the accuracy and granularity of image labeling.

5. Features

The Image Labeling Tool offers a range of features that cater to image data labelers using the Windows operating system. These features are designed to streamline the process of organizing and labeling images effectively. The following are the key features of the application:

Move Images:

- The tool allows users to move images from a designated root folder into labeled folders based on user-defined labels.
- Users can navigate between images using arrow keys or dedicated navigation buttons.
- Labels are associated with trigger keys, enabling quick and efficient image movement.

Label Management:

- Users can add new labels with unique names, trigger keys, and folder paths through an intuitive Add Label dialog.
- The tool checks for duplicate names, trigger keys, and folder paths to ensure accurate label creation.
- Labels are automatically refreshed with updated image paths when changes occur in their corresponding folders.

Root Folder Selection:

- Users can choose between labeling images only in the top directory or in all subdirectories of the root folder.
- The Root Folder feature provides flexibility in managing and labeling large datasets organized in various directory structures.

Undo Functionality:

- The application allows users to undo their image movements, restoring images to their original locations in the root folder.
- The undo feature provides users with the flexibility to correct label assignments in case of accidental or erroneous movements.

Image Editing (Color Channel Swap):

- Users can edit images by swapping color channels (RGB, GRB, GBR, BRG, and BGR) using a simple and intuitive interface.
- The edited images can be saved to a new file, preserving the original images while allowing for further analysis or comparison.

In-Development: Draw Bounding Box (Future Enhancement):

- The tool will introduce the "Draw Bounding Box" functionality, empowering users to label objects within images by drawing bounding boxes around them.
- This feature will enhance the accuracy and granularity of image labeling, providing more detailed and precise annotations.

The Image Labeling Tool offers an efficient and user-friendly interface, combining essential functionalities for image data labeling. It simplifies the process of organizing and labeling large image datasets, making it a valuable tool for researchers, data scientists, and other professionals involved in computer vision, machine learning, and image analysis tasks.

6. User Guide

The User Guide provides step-by-step instructions on how to use the Image Labeling Tool efficiently. This guide is designed to assist image data labelers using the Windows operating system in effectively organizing and labeling their image datasets.

6.1. Installation

Download the Image Labeling Tool from here: khanha2k46pbc/project2 (github.com)
You do not need to install any additional libraries.

6.2. Launching the Application

After the downloading, unzip it, open the file **project2/Project-2/bin/Debug/Project-2.exe,** if you don't care about the source code, you can remove it and keep only **Debug** folder. The application will launch, displaying the Mode Selection window.

6.3. Mode Selection

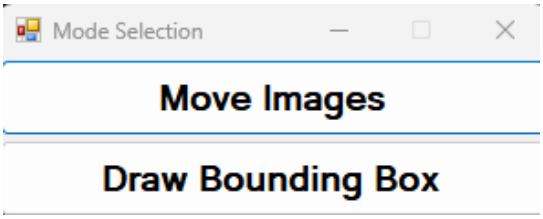

Figure 2: Mode Selection

In the Mode Selection window, choose between "Move Images" and "Draw Bounding Box" modes by clicking on the respective buttons.

For this User Guide, let's select the "Move Images" mode, which is the primary functionality of the tool.

6.4. Move Images Functionality

Upon selecting the "Move Images" mode, the main working window will appear.

Click the "Choose Root Folder" button to select the folder containing the images to be labeled. This will open a folder selection dialog.

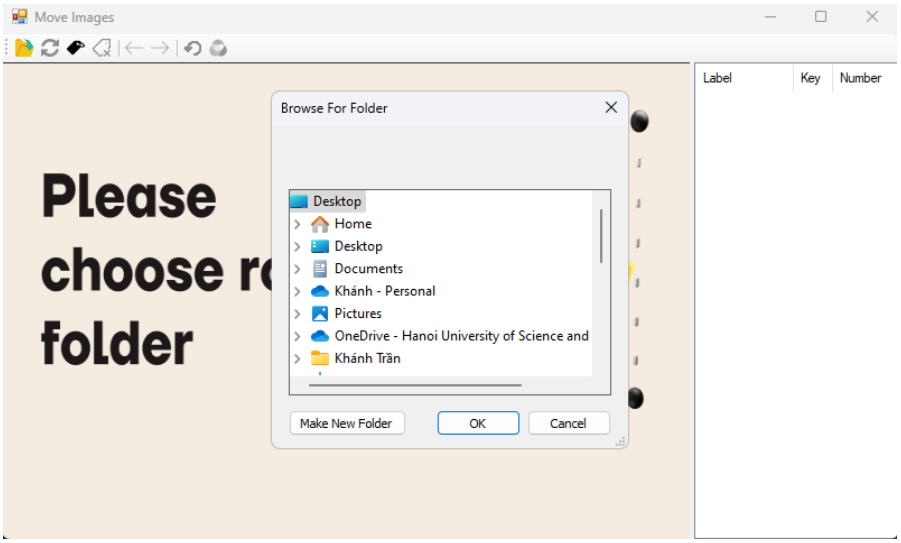

Figure 3: Choose Root Folder

After selecting the root folder, the image navigation area will display the first image from the folder.

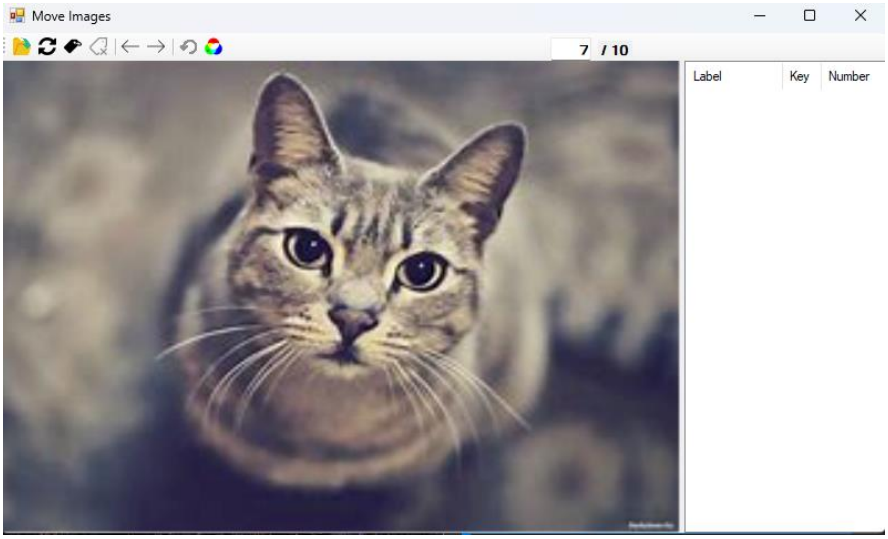

Figure 4: First image

To navigate between images, use the arrow keys on your keyboard or click the dedicated arrow buttons on the application window.

6.5. Label Management

To add a new label, click the "Add Label" button. This will open the "Add Label" dialog.

In the "Add Label" dialog, provide a unique label name, a single-character trigger key, and the folder path where the labeled images will be moved.

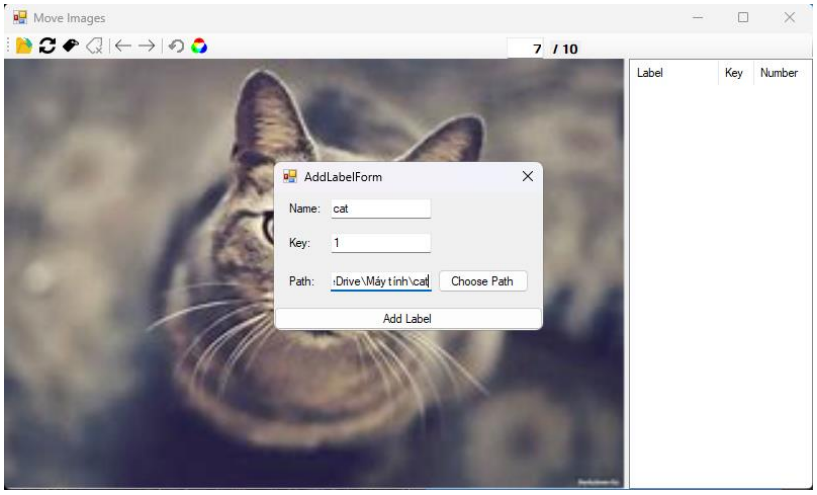

Figure 5: Add Label

Click the "Add" button to create the new label. The label will now appear in the label list on the main working window.

The "Refresh" button can be used to update the information in the root directory and the label list.

6.6. Moving Images

To label an image, press the trigger key associated with the desired label. The labeled image will be moved to the corresponding folder.

The application supports undo functionality, allowing you to revert the last image movement if needed.

6.7. Image Editing (Color Channel Swap)

In the main working window, click the "Edit Image" button to access the image editing feature.

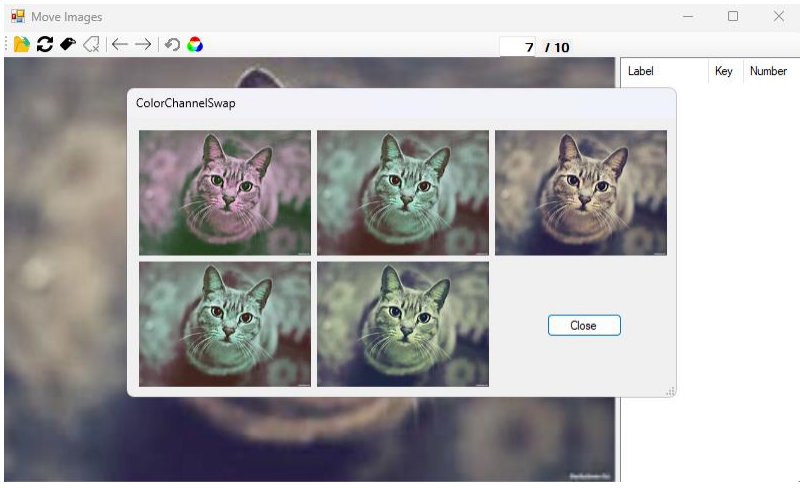

Figure 6: Color Channel Swap dialog

The Color Channel Swap dialog will open, displaying the original image along with five color channel swap options (RGB, GRB, GBR, BRG, and BGR).

Click on any of the color channel swap options to preview the edited image.

Click the respective color channel swap image to save the edited image with the selected color channel swap to a new file.

6.8. Exiting the Application

To exit the Image Labeling Tool, close the main working window by clicking the "X" button in the top right corner.

If you are currently in the "Draw Bounding Box" mode (in development), save your progress before exiting the application.

6.9. Draw Bounding Box (In Development)

Note: This section will be updated once the "Draw Bounding Box" functionality is fully developed and integrated into the application.

Congratulations! You have now learned how to use the Image Labeling Tool to efficiently organize and label your image datasets. The tool provides an intuitive and effective solution for image data labeling tasks, simplifying the process and improving overall productivity.

For any questions or issues, please refer to the provided contact information or support resources.

7. Future Enhancements

The Image Labeling Tool has been designed with extensibility in mind, allowing for future enhancements and improvements to meet evolving user needs. Here are some potential future enhancements that could be implemented to further enhance the functionality and usability of the application:

Fully Implement "Draw Bounding Box" Functionality:

- Complete the development of the "Draw Bounding Box" functionality to enable users to label objects within images by drawing bounding boxes around them.
- Provide intuitive tools for resizing, moving, and deleting bounding boxes to improve annotation precision.
- Include options for labeling multiple objects in a single image and saving annotations in standard formats like COCO or Pascal VOC.

Support for Additional Image Formats:

- Extend the tool's image handling capabilities to support a broader range of image formats, such as JPEG2000, TIFF, and RAW formats.
- Enable users to import and label images in diverse formats commonly used in computer vision and machine learning tasks.

Customizable Image Editing Options:

- Enhance the image editing feature to provide users with more editing options, such as adjusting brightness, contrast, and color levels.
- Implement additional image filters and transformations to cater to users' varied image preprocessing requirements.

Export and Import Label Configurations:

- Allow users to export and import label configurations to transfer label settings between different instances of the application.
- This feature will facilitate collaboration and sharing of label setups among team members working on similar labeling projects.

Multi-platform Compatibility:

- Extend the application's compatibility beyond Windows and make it accessible on other platforms, such as macOS and Linux.
- Implement a web-based version of the tool to offer platform-independent usage with cloud-based storage options.

Integration with Cloud Storage Services:

- Integrate the application with popular cloud storage services (e.g., Google Drive, Dropbox) for seamless image storage and sharing.
- Enable users to access and label images stored in the cloud directly from the application.

Machine Learning Integration:

- Introduce integration with machine learning frameworks and libraries to leverage the labeled datasets for training and evaluation.
- Provide an option to export labeled data in common formats used by machine learning models.

Batch Image Labeling:

- Introduce batch processing capabilities to label multiple images simultaneously, reducing manual effort and saving time.
- Implement image recognition algorithms to semi-automate the labeling process based on predefined patterns or object detection models.

User Interface Customization:

- Allow users to customize the user interface, such as choosing color themes or rearranging UI components to suit individual preferences.

User Feedback and Analytics:

- Implement a feedback mechanism within the application to collect user suggestions and bug reports for continuous improvement.
- Incorporate analytics to gather usage statistics, helping identify popular features and areas for enhancement.

These future enhancements will not only expand the capabilities of the Image Labeling Tool but also make it a more versatile and user-friendly application for image data labelers and researchers. The development of these features can be prioritized based on user feedback, use cases, and emerging trends in image labeling and computer vision technologies.

8. Conclusion

The Image Labeling Tool presented in this project offers a valuable solution for image data labelers using the Windows operating system. With its user-friendly interface and powerful functionalities, the application streamlines the process of organizing and labeling image datasets, contributing to more efficient data annotation and analysis. Throughout the development of this tool, several key insights and achievements have been realized.

The tool's primary functionality, "Move Images," enables users to organize images by associating them with user-defined labels and moving them to corresponding folders. The label

management system allows for easy addition, deletion, and editing of labels, providing flexibility and adaptability to users' labeling projects. The undo feature adds an extra layer of control, enabling users to correct labeling errors and revert image movements as needed.

Additionally, the image editing capability, currently supporting color channel swapping, empowers users to perform basic image modifications directly within the application. The forthcoming "Draw Bounding Box" functionality, still in development, promises to further enhance the precision and granularity of image labeling, making the tool even more versatile for diverse labeling tasks.

The Image Labeling Tool is not only a valuable asset for researchers and data scientists working on computer vision projects but also a foundation for future enhancements and improvements. The extensibility of the application opens the door for further features, such as multi-platform compatibility, cloud integration, and machine learning support, catering to evolving user needs and technological advancements.

While the current version of the Image Labeling Tool addresses specific image labeling requirements, the development journey does not end here. Continuous feedback from users and stakeholders will be instrumental in refining the tool and directing future development efforts. As the tool gains popularity and adoption, user suggestions, bug reports, and emerging trends will guide us in prioritizing future enhancements and solidifying the tool's position as a reliable and indispensable resource for the image data labeling community.

In conclusion, the Image Labeling Tool demonstrates the successful application of software engineering principles to deliver a practical and efficient solution for image data labelers. Its combination of intuitive user interface, powerful labeling capabilities, and potential for future enhancements make it a promising tool for simplifying complex image labeling tasks and fostering advancements in computer vision and artificial intelligence.