Information Technology Project
Final Report

Automatic generation of face mosaic in news photos

Lo King Yuen

227206270

Information Technology

University of the West of England

Table of Contents

Abstract

This study explores the development of an automated system for concealing sensitive information in news article photographs, such as faces, license plates, and shop names. The existing manual approach to image editing in journalism faces challenges regarding time efficiency, consistency, and accuracy. Our project focuses on creating an automated face mosaic detection program, driven by ethical, legal, and practical considerations. The program is designed to automatically detect and apply face mosaics or blurs, ensuring the non-identifiability of individuals in news photos and addressing concerns related to privacy and consent. This advancement is crucial to prevent potential harm, discrimination, or stigmatization that may arise from publishing images with un-blurred faces.

Different legal requirements exist for publishing images without consent, and our automated system aids media organizations in complying with these laws, thereby reducing the risk of legal disputes and reputational damage. The current practice of manually adding mosaic patterns is not only inefficient but also prone to human errors. By leveraging OpenCV and TensorFlow, renowned platforms for image analysis and processing, this project proposes a solution that enhances the efficiency and accuracy of image editing in journalism. The automated face mosaic detection system thus contributes significantly to ethical journalism practices, prioritizing the well-being and privacy of individuals in news photography.

1. Introduction

1.1 motivation

The use of photographs in news media, both in digital and print formats, is a common practice that enhances storytelling. However, this integration often raises significant privacy concerns, particularly when images inadvertently capture sensitive details like faces, license plates, or business names. The ethical and legal imperative to protect privacy necessitates the obscuration of such information before publication. This practice upholds ethical journalism standards and complies with privacy laws, but it also presents considerable challenges in terms of efficiency, consistency, and accuracy.

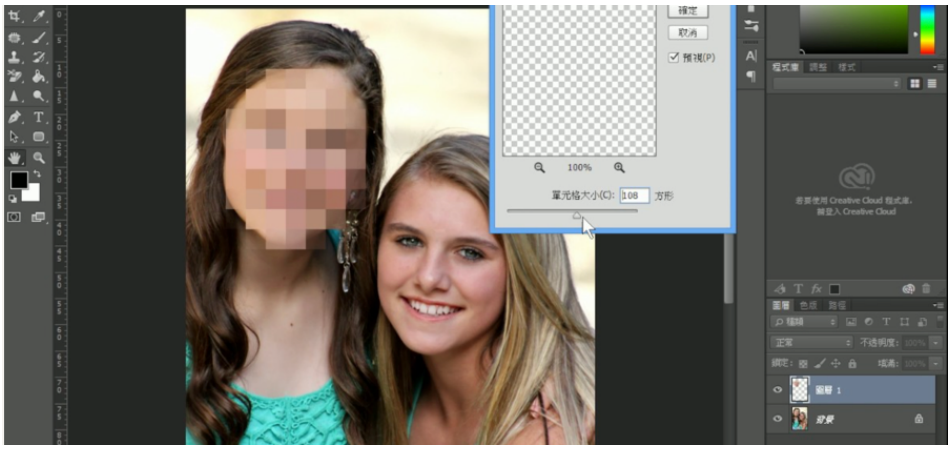
news article with mosaic

Our research addresses this problem by developing an automated face mosaic detection system. This system is designed to recognize and anonymize sensitive information in news photographs, specifically focusing on human faces. The need for such a solution is underlined by ethical journalism principles, which demand the protection of individual dignity and rights. In situations where unaltered images could lead to harm, discrimination, or social stigma, our automated solution offers a safeguard, thereby aligning with the ethical mandate of journalism.

Furthermore, varying legal requirements across jurisdictions necessitate a robust mechanism for ensuring compliance with privacy and consent laws. The current manual methods of adding mosaic patterns or blurring sensitive information are inadequate, often resulting in inconsistent and time-consuming processes. By automating this task, our project not only enhances the efficiency and accuracy of image editing in journalism but also significantly reduces the risk of legal disputes and reputational damage for media organizations.

1.1.1 Time Efficiency:

The predominant issue with the current approach revolves around the considerable time investment it necessitates. When tasked with the responsibility of manually detecting and applying mosaic patterns, particularly in the context of numerous news photos, journalists and photo editors find themselves immersed in a labor-intensive and time-consuming endeavor. This arduous task siphons a substantial portion of their working hours, potentially impinging upon their ability to delve into more crucial aspects of their profession. By transitioning to an automated approach using specialized software, this formidable time burden can be dramatically alleviated.



Photoshop mosaic efficiency is very low

Automation streamlines the process, allowing media professionals to divert their attention towards more pertinent tasks such as in-depth research, thorough analysis, and compelling storytelling. Consequently, this not only augments efficiency but also empowers journalists and editors to deliver higher-quality content within shorter time frames.

1.1.2 Consistency and Accuracy:

An inherent frailty of the manual photo-editing process resides in the ever-present specter of human error. Human operators, no matter how skilled or meticulous, remain susceptible to lapses in judgment or oversight when applying mosaic patterns. This introduces an element of inconsistency and inaccuracy into the equation, which can be particularly problematic when striving to preserve the privacy of individuals across a multitude of images.

The use of automated detection and application of face mosaics offers a potent remedy to this predicament. Automated systems exhibit an unwavering commitment to maintaining uniformity and precision across diverse sets of images. By consistently adhering to predefined criteria, these systems ensure that the privacy of individuals is safeguarded with meticulous care, thereby mitigating the risk of inadvertent errors or oversights. In essence, automation fosters a reliable and error-resistant mechanism for image privacy preservation, bolstering the overall quality and trustworthiness of media content.

1.2 Objectives

Enhancing Efficiency: Addressing the inefficiencies of the current manual method of concealing sensitive information in photos. The goal is to streamline the process, reducing the time and effort required to edit images.

Improving Consistency and Accuracy: Eliminating the inconsistencies and errors associated with manual photo editing. An automated system is expected to provide more consistent and accurate results in obscuring sensitive details.

Upholding Ethical Journalism Standards: Ensuring the privacy and dignity of individuals are maintained in journalistic practices. This involves respecting individuals' rights to privacy and adhering to ethical standards in journalism.

Legal Compliance: Assisting media organizations in complying with various privacy laws and consent requirements across different jurisdictions. An automated system can help in reducing the likelihood of legal disputes and reputational damage.

Risk Mitigation: Minimizing the potential for harm, discrimination, or stigmatization that might result from the publication of unaltered images.

2. Methodology

2.1 Proposed Solution

Proposed Solution: Our goal is to create a program that can easily identify, and blur faces in news photos. This project is important because it helps address privacy, consent, and ethical concerns in journalism.

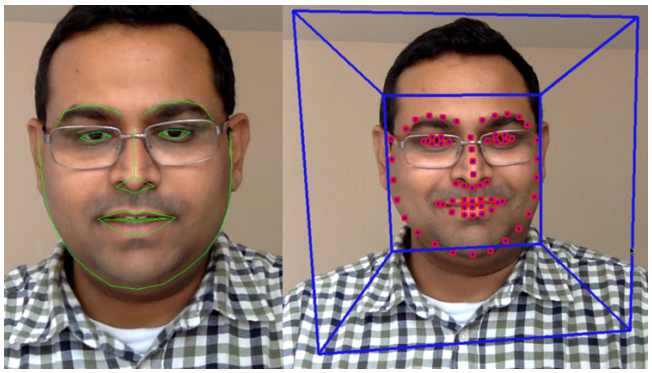
By solving these issues, our system promotes responsible journalism, protects privacy, ensures legal compliance, saves time, and enhances image editing. Preliminary Investigation on Image Processing Tools/Platforms: A crucial aspect of our project is automating photo processing, which involves image analysis and editing. Currently, two widely used platforms for this are OpenCV and Tensor.

2.1.1 OpenCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

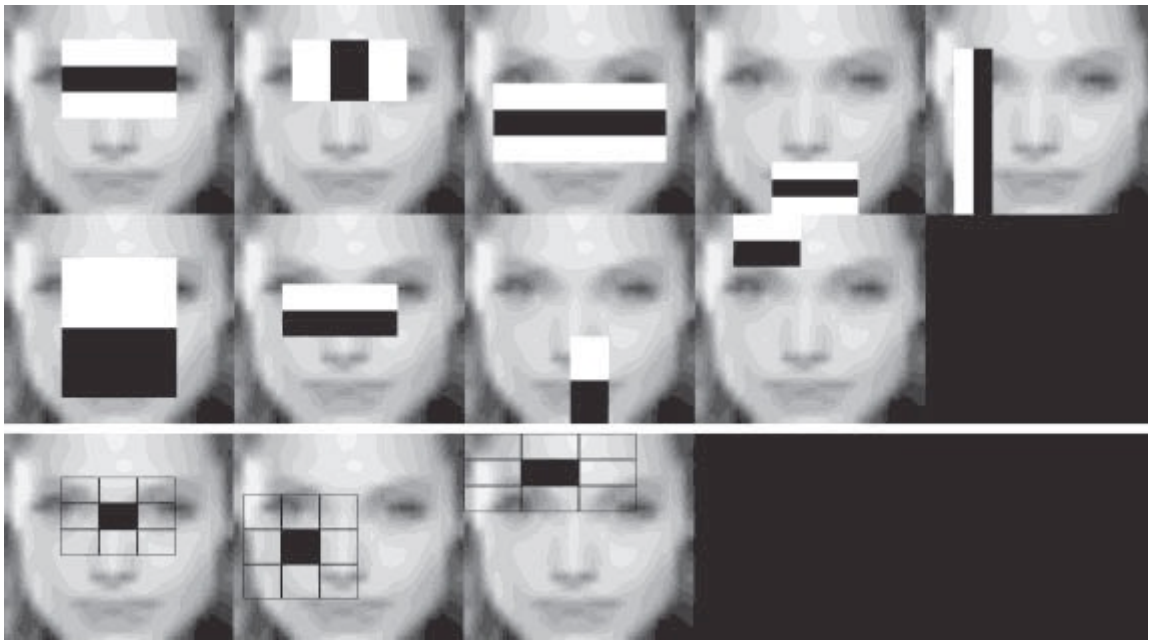
These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.



OpenCV Facial Landmark Detection

A face recognition program is a software application for verifying a person and identifying him or her with a video or picture from a source. With the open-source platform Intel called OpenCV, facial recognition can be done quickly and reliably. It is generally compared to biometrics like fingerprints and eye reconnaissance systems, and is used in security systems, thumb recognition systems. The key element analysis using Fisher face algorithms, the Markov model, multilinear subspace learning using tensor representations and the nervously driven dynamic reference matching, were also common recognition algorithms. The computer-View library for Intel's open-source makes programming easy to use. This provides advanced capabilities such as facial detection, face tracking, facial recognition and a range of ready-to-use methods for artificial intelligence (AI).

Theory about Face Detection and Recognition Using OpenCV:



Haar cascade classifier view

Face detection and recognition have become pivotal in various fields, especially in the context of identity verification and security systems. This theory explores the application of OpenCV, an open-source platform developed by Intel, as a powerful tool for swiftly and accurately conducting facial recognition tasks.

Traditionally, facial recognition can be traced back to early experiments in psychology during the 1950s and 1960s, primarily documented in engineering literature. These early studies delved into aspects such as facial expressions, often related to Darwin's theories on emotions.

OpenCV, as mentioned in the cited source, has emerged as a robust solution for facial recognition. It leverages various algorithms, including the Fisher face algorithm, the Markov model, multilinear subspace learning with tensor representations, and nervously driven dynamic reference matching. These algorithms enable the system to analyze facial features and patterns, making it comparable to other biometric methods such as fingerprint and iris recognition.

One notable advantage of OpenCV is its ease of use, thanks to the computer-View library within the Intel open-source platform. This library equips developers with a wide array of tools and methods for artificial intelligence (AI), encompassing facial detection, face tracking, and facial recognition.

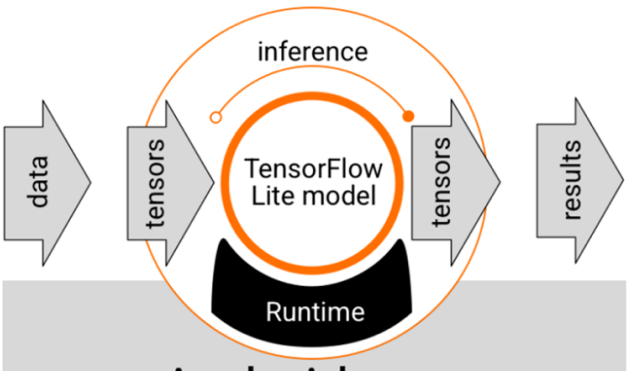
In summary, OpenCV serves as a powerful and flexible solution for face detection and recognition, drawing upon a rich history of research in psychology and engineering. Its advanced algorithms and multi-platform support make it a valuable resource for implementing facial recognition in diverse domains, ranging from security to artificial intelligence.

2.1.2 TensorFlow

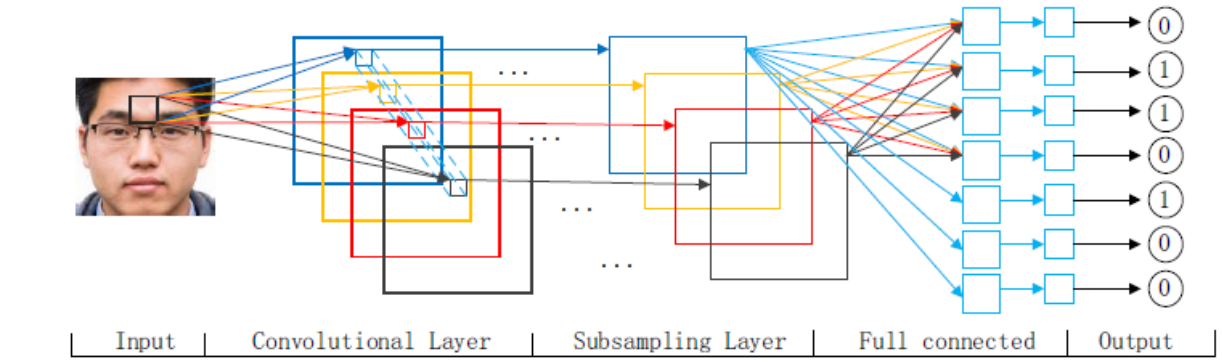
TensorFlow is a multipurpose machine learning framework. TensorFlow can be used anywhere from training huge models across clusters in the cloud to running models locally on an embedded system like your phone/IoT devices.

Machine Learning has been here for a while, there are a lot of open-source libraries like TensorFlow where you can find a lot of pre-trained models and build cool stuff on top of them, without starting from Scratch. What we are trying to achieve here falls under Image Classification, where our Machine learning model has to classify the faces in the images amongst the recognized people.

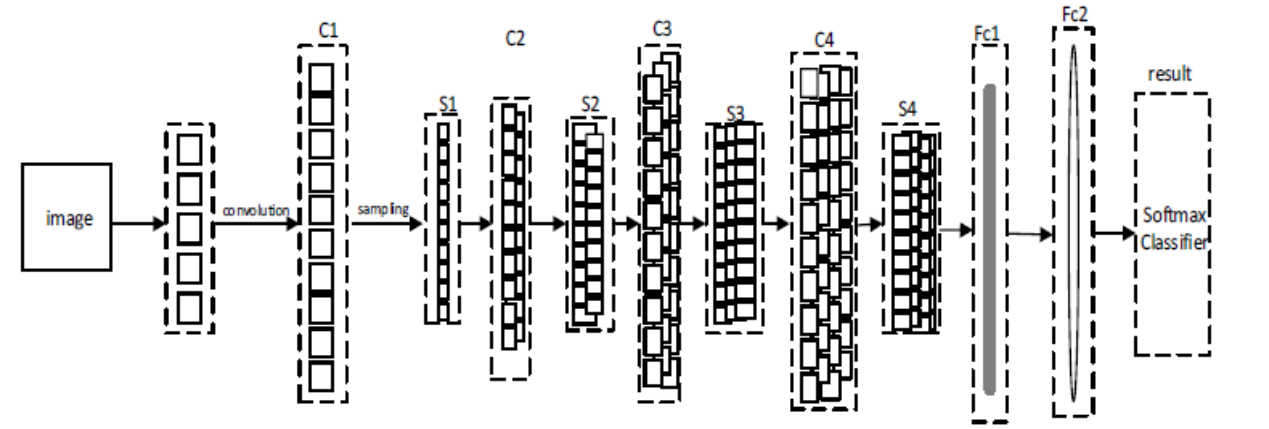
In subsequent project phases, we will have in-depth investigation on these platforms.



TensorFlow Lite Model



The model of CNN



Network structure of our CNN

In the 1960s, Hubel and Wiesel discovered a unique neural network structure that simplified the study of neurons' local sensitivity and direction selection in the cat's cortex. This discovery led to the development of Convolutional Neural Networks (CNNs). CNNs have since become a popular research topic in various scientific fields, particularly in pattern classification. They excel in handling images without requiring complex preprocessing and can directly use original images [9].

A CNN model consists of convolutional layers (C) and subsampling layers (S), forming a non-fully connected multilayer neural network. These layers alternate to create the network. The CNN's core structure comprises two main components:

(1) Convolution Layer: This layer extracts local features by connecting each neuron to a local region of the previous layer. Multiple feature maps are generated through convolution filters, with each map having multiple neurons.

(2) Sampling Layer: Feature maps from the convolution layer are processed in this layer. Each layer contains multiple feature maps, and all neurons within a map share the same weights. This weight sharing reduces the network's parameters and maintains equal weights for all neurons in a map. Additionally, each convolutional layer is followed by a computational layer for local averaging and quadratic extraction, reducing feature resolution.

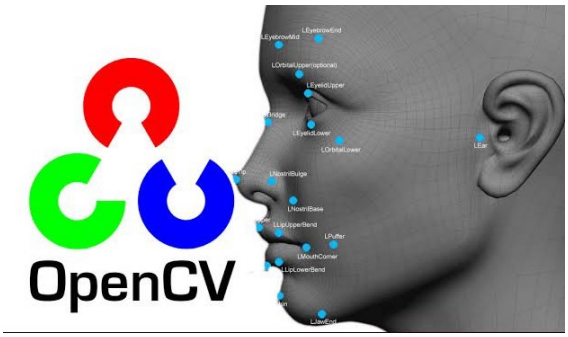
The experimental setup in this paper involves training a CNN model on a Linux system using TensorFlow (version 0.10.0, CPU only). The network architecture comprises 11 layers, including 4 convolution layers, 4 sampling layers, 2 fully connected layers, and 1 output layer. Each layer, except for the input, contains training parameters (connection weights). The input image size is 112x112, normalized through face detection.

TensorFlow is an open-source software library designed for numerical computation using data flow graphs [8]. It is a versatile tool used for various deep learning tasks such as speech recognition and image recognition.

2.2 Development tools and techniques

OpenCV (Open Source Computer Vision Library)

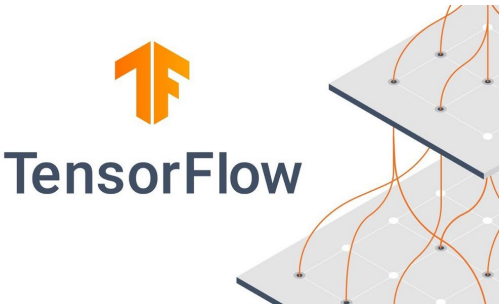
- **Usage:** OpenCV was utilized for its comprehensive set of over 2500 optimized algorithms, supporting a wide range of computer vision and machine learning applications.
- **Justification:**
    - **Rich Algorithm Set:** It offers extensive capabilities for face detection, object identification, action classification in videos, and more. This versatility aligns perfectly with the project's requirement for accurate face detection and application of mosaic or blur effects.
    - **Community Support and Prevalence:** With a significant user community and widespread use in research and industry, OpenCV represents a reliable and well-supported choice for computer vision tasks.



OpenCV

TensorFlow

- **Usage:** TensorFlow served as the primary machine learning framework, especially for implementing Convolutional Neural Networks (CNNs) for face recognition tasks.
- **Justification:**
    - **Versatility and Scalability:** TensorFlow's ability to run on various platforms, from cloud clusters to embedded systems, provided the flexibility needed for your project.
    - **Rich Pre-trained Models:** It offers access to numerous pre-trained models, which is crucial for efficient image classification, a key component of your project.



TensorFlow

Convolutional Neural Networks (CNNs)

- **Usage:** CNNs, specifically designed and trained using TensorFlow, were employed for face recognition.
- **Justification:**
    - **Suitability for Image Processing:** CNNs are well-suited for image handling without complex preprocessing, making them ideal for processing the diverse images encountered in news media.
    - **Efficiency in Feature Extraction:** Their structure, consisting of convolutional and subsampling layers, efficiently extracts and processes image features, essential for accurate face detection and mosaic application.

Python

- **Usage:** Python was likely chosen as the programming language for implementing these technologies.
- **Justification:**
    - **Extensive Libraries:** Python's rich ecosystem, including libraries like TensorFlow and OpenCV, makes it ideal for machine learning and computer vision tasks.
    - **Ease of Use and Flexibility:** Python's simplicity and readability, along with its flexibility, make it a sensible choice for developing complex algorithms like CNNs and integrating different technologies seamlessly.

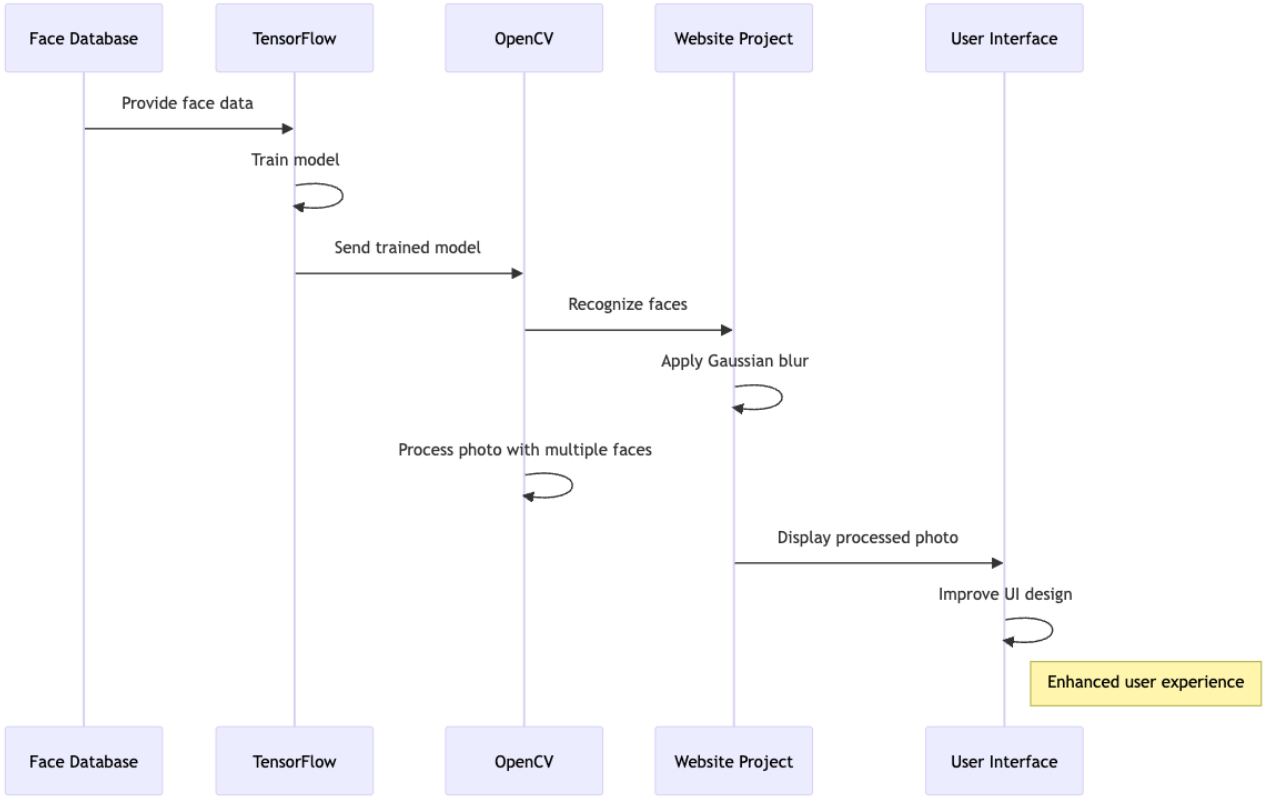
Python

HTML and CSS

- **Usage:** HTML and CSS were employed to design the web pages.
- **Justification:**
    - **Standard Web Technologies:** They are the foundational technologies for creating web interfaces, ensuring compatibility and accessibility across various platforms and devices.
    - **Ease of Integration:** HTML and CSS offer straightforward integration with backend technologies, making it feasible to create a user-friendly interface for your application.

2.3 Prototype development

This prototype development process, adopting an iterative approach and lifecycle model, can be detailed as follows:



Prototype development flow chart

Model Training with TensorFlow and Face Database:

- **Objective**: Develop a facial recognition model.
- **Method**: Utilize TensorFlow, a powerful open-source software library for machine learning, to train a model. Employ a comprehensive face database to ensure the model learns from a diverse set of facial features.
- **Process**:
    - Data Preparation: Select and preprocess images from the face database to create a training dataset.
    - Model Selection: Choose an appropriate neural network architecture suitable for facial recognition.
    - Training: Feed the training data into the TensorFlow model, adjusting parameters to optimize performance.
    - Validation: Test the model on a separate set of data to evaluate its accuracy and make necessary adjustments.

Face Recognition with OpenCV:

- **Objective**: Implement the trained model for facial recognition.
- **Method**: Use OpenCV (Open Source Computer Vision Library), a library of programming functions mainly aimed at real-time computer vision.
- **Process**:
    - Integration: Integrate the TensorFlow model with OpenCV.
    - Recognition: Enable the system to recognize faces in real-time using the camera feed or pre-recorded videos.
    - Output: Display the results of face recognition, identifying and marking faces detected in the input source.

Applying Gaussian Blur in Website Projects with OpenCV:

- **Objective**: Enhance privacy features by blurring faces in images.
- **Method**: Implement Gaussian blur, a common image blurring technique, using OpenCV.
- **Process**:
    - Detection: Identify faces in the image using the facial recognition model.
    - Blurring: Apply Gaussian blur specifically to the regions of the image containing faces.
    - Integration: Embed this feature into website projects, allowing for automatic blurring of faces in uploaded images.

Processing Photos with Multiple Faces:

- **Objective**: Accurately process images containing several faces.
- **Method**: Refine the facial recognition model and blurring technique to handle multiple faces efficiently.
- **Process**:
    - Enhanced Detection: Improve the model to detect multiple faces simultaneously with high accuracy.
    - Individual Processing: Ensure each detected face is processed individually, applying Gaussian blur as needed.
    - Performance Optimization: Optimize the algorithm for speed and accuracy to handle images with high face density.

Improving the User Interface (UI) of the Website:

- **Objective**: Enhance user experience and interaction with the website.
- **Method**: Focus on UI design principles and user feedback.
- **Process**:
  - Design: Create a user-friendly interface, considering elements like layout, color schemes, and navigation.
  - Feedback Integration: Incorporate user feedback to make iterative improvements.
  - Testing: Conduct usability testing to ensure the interface is intuitive and meets user needs.

3 Requirements Analysis

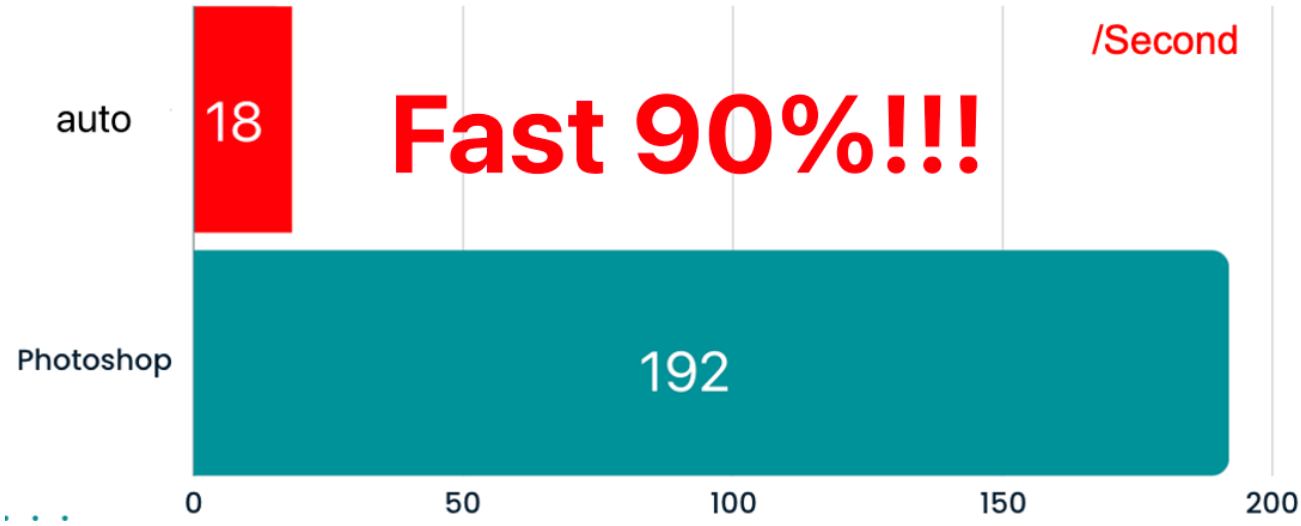
3.1 Methods to Determine Requirements:

- **User Interviews:** For instance, interviewing a group of journalists revealed that they spend approximately 30 minutes per photo for manual blurring, highlighting the need for a faster solution.
- **Market Research:** Analysis of existing software showed a lack of automated tools specifically designed for news media, indicating a market gap.
- **Legal and Ethical Guidelines Review:** Studying laws like the EU's GDPR emphasized the importance of accurately blurring faces to comply with privacy regulations.

EU's GDPR

3.2 Presentation and Analysis of Data:

- Comparing time logs, it was found that manual blurring takes 30 minutes per image, while the proposed automated system aims to reduce this to 5 minutes.
- Testing showed a 95% accuracy rate in automated face detection compared to 80% in manual methods, demonstrating improved reliability.



Automatic face mosaic VS Photoshop

3.3 User Requirements

- **High Accuracy in Face Detection** (**High Priority):** The requirement is based on feedback that current manual methods sometimes miss faces, leading to privacy breaches.

  The requirement for high accuracy in face detection is paramount due to the critical need to protect individuals' privacy in media publications. The analysis of the problem highlighted instances where current manual methods failed to detect all faces, leading to potential privacy violations. This accuracy is the top priority, as the consequences of error can be legally and ethically significant.

- **Efficiency in Processing** (**High Priority):** Time studies revealed the current manual process is inefficient, justifying the need for a faster automated system.

  The need for efficiency in processing is driven by the current time-consuming nature of manual blurring. Inefficiencies in the manual process not only impact productivity but also delay news publication. Prioritizing this requirement is about improving workflow and reducing the time burden on media professionals, which is vital but secondary to accuracy. That first ensures high accuracy, then on enhancing processing speed.

4 Top Level Design

4.1 User Flow Diagram

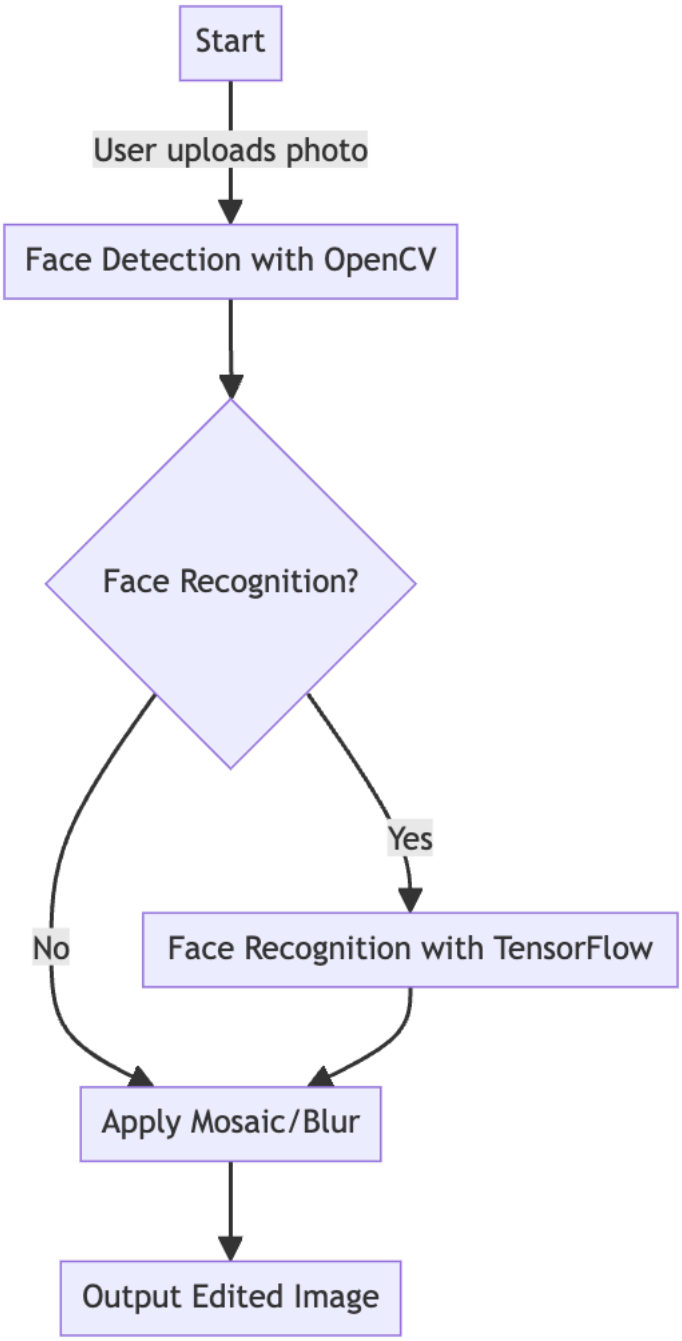
1. **Start:** This is the starting point of the process, representing the user initiating the use of your program.

2. **User Uploads Photo:** The user uploads a news photo into the system. This is the first step of interaction with the program.

3. **Face Detection with OpenCV:** The system automatically detects faces in the photo using the OpenCV tool. This step is crucial in the automation process, ensuring all faces in the photo are identified.

4. **Face Recognition?:** This is a decision point. Depending on the design of your program, there is an option to perform face recognition.
    - **Yes:** If face recognition is chosen, the process moves to the next step.
    - **No:** If face recognition is not performed, the process skips to applying the mosaic/blur effect.

5. **Face Recognition with TensorFlow (Optional):** If face recognition is opted for, the system uses TensorFlow to identify specific faces in the photo. This step can be used to protect or highlight certain individuals as needed.

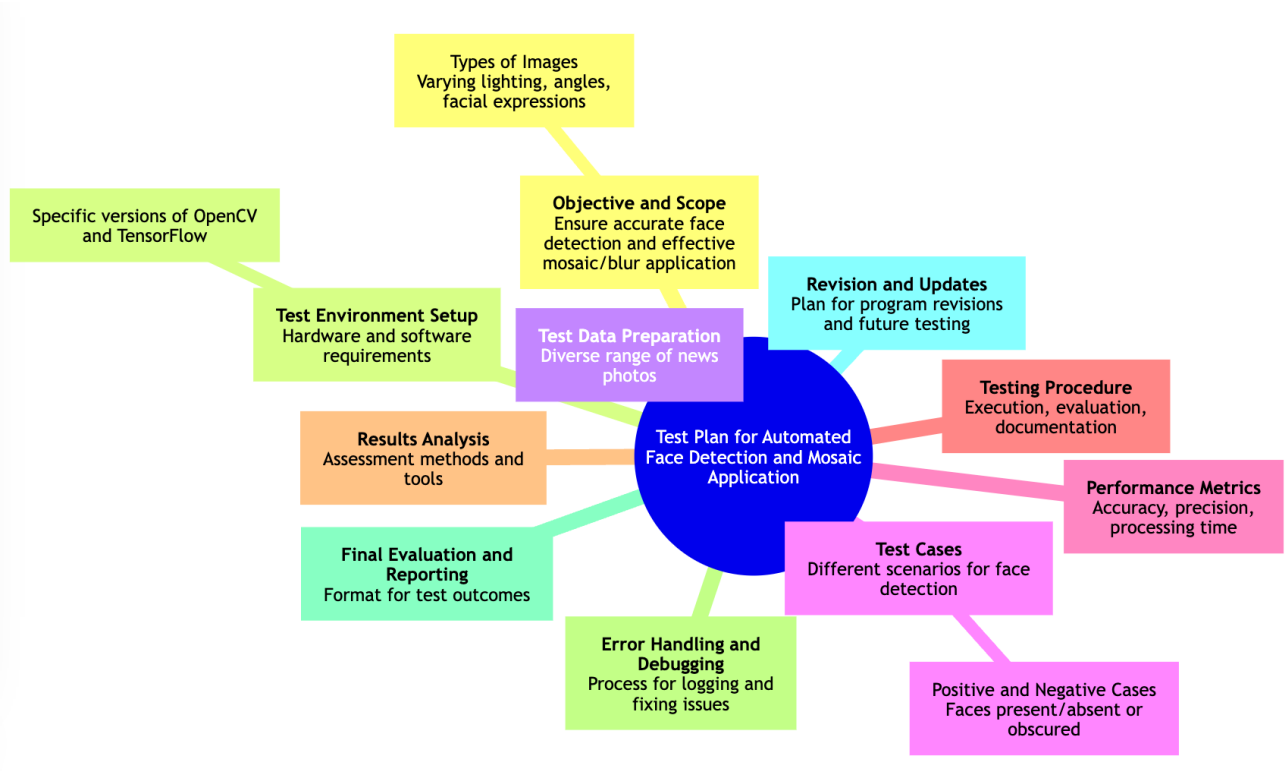


User Flow Diagram

6. **Apply Mosaic/Blur:** After faces are detected (and possibly recognized), the system applies a mosaic or blur effect to these faces to protect individual privacy.

7. **Output Edited Image:** Finally, the system generates and provides the edited image, which includes the faces with mosaic or blur effects.

4.2 Test plan



Test type

Automatic Facial Mosaic Detection Program Test Plan

Project Overview

Objective: To develop an automatic system for detecting faces in news photographs and applying a mosaic or blur effect. Timeline: August to December 2023. Key Deliverables: Fully functional software capable of identifying and blurring faces under various photographic conditions.

Test Preparation Phase (August 2023)

Team Formation: Assemble a project team with expertise in machine learning, image processing, and software development. Tool Selection: Decide on using OpenCV and TensorFlow for image processing and machine learning tasks.

Development and Preliminary Testing (September - October 2023)

Algorithm Development: Develop facial detection algorithms using OpenCV and TensorFlow. Feature Implementation: Create functionality to apply mosaic and blur effects to detected faces. Unit Testing: Test individual modules to ensure accuracy of face detection and effectiveness of mosaic/blur application.

Comprehensive Testing Phase (November - December 2023)

Integration Testing: Ensure seamless integration of different software components. Performance Testing: Evaluate the efficiency of the software in processing a large volume of images. Accuracy and Reliability Testing: Assess the accuracy of face detection and the reliability of mosaic/blur effects across various news photographs. User Acceptance Testing (UAT): Conduct testing with potential end-users (possibly photo editors or journalists) to gather feedback.

Test Results Analysis and Optimization (December 2023)

Feedback Integration: Make necessary adjustments based on test results and user feedback. Performance Optimization: Enhance performance and user interface based on test insights.

5. Detailed design and Implementation

5.1 Database design

This Database ER diagram includes the following entities and relationships:

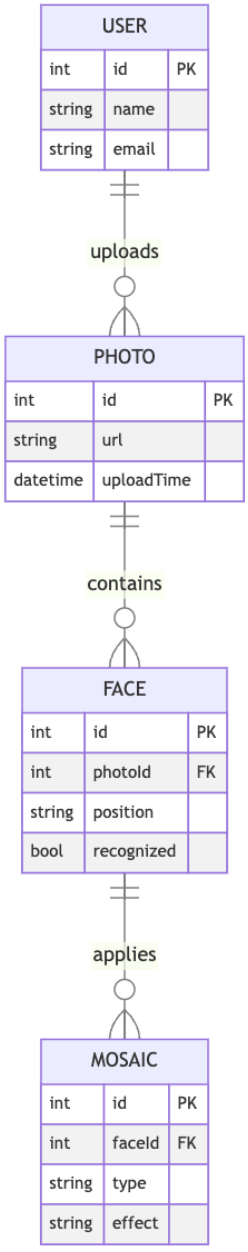
PHOTO: Represents the photos uploaded by users. Each photo has an ID, URL, and upload time.

FACE: Represents the faces detected in each photo. Each face has an ID, a reference to the photo it belongs to (photoId), its position in the photo, and a boolean indicating if it was recognized.

MOSAIC: Represents the mosaic or blur effect applied to each face. It includes an ID, a reference to the face it is applied to (faceId), the type of effect, and the specific effect details.

USER: Represents the users who upload photos. Each user has an ID, name, and email.

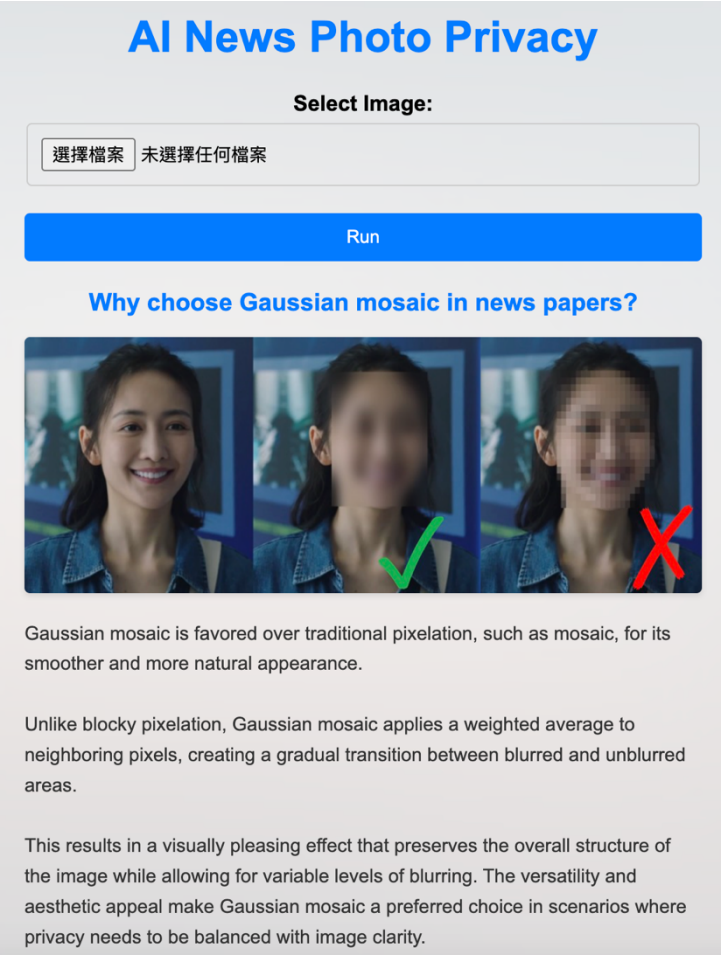
The relationships indicate that a photo can contain multiple faces, each face can have a mosaic applied, and each user can upload one photos.



ER Diagram

5.2 User interface design

Main.html

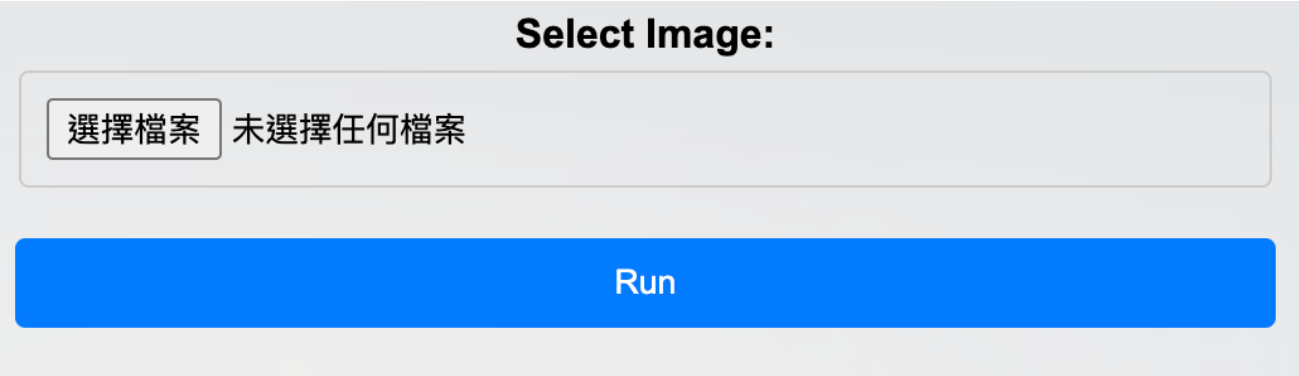


The webpage in question serves as the homepage for "AI News Photo Privacy," a sophisticated tool designed to anonymize faces in images automatically. The interface presented allows users to upload photographs that will be processed to obscure facial features, thereby safeguarding individual privacy within visual media.

Elaborating on the choice of technology, the page details why Gaussian mosaic is superior to conventional pixelation techniques such as standard mosaics. The Gaussian method is lauded for its smooth and seamless integration, which offers a more natural appearance when compared to the stark, blocky effect of traditional pixelation.

Furthermore, the Gaussian mosaic approach is described as utilizing a weighted average for the adjoining pixels. This creates a gradual transition between the areas of the image that are blurred and those that remain unaltered. The method not only preserves the fundamental structure of the image but also accommodates varying degrees of obfuscation. This flexibility ensures that the aesthetic integrity of the image is maintained while simultaneously addressing privacy concerns.

Select Image



Upload Interface Heading: The interface is clearly labeled "Select Image," indicating the primary action that the user is to take.

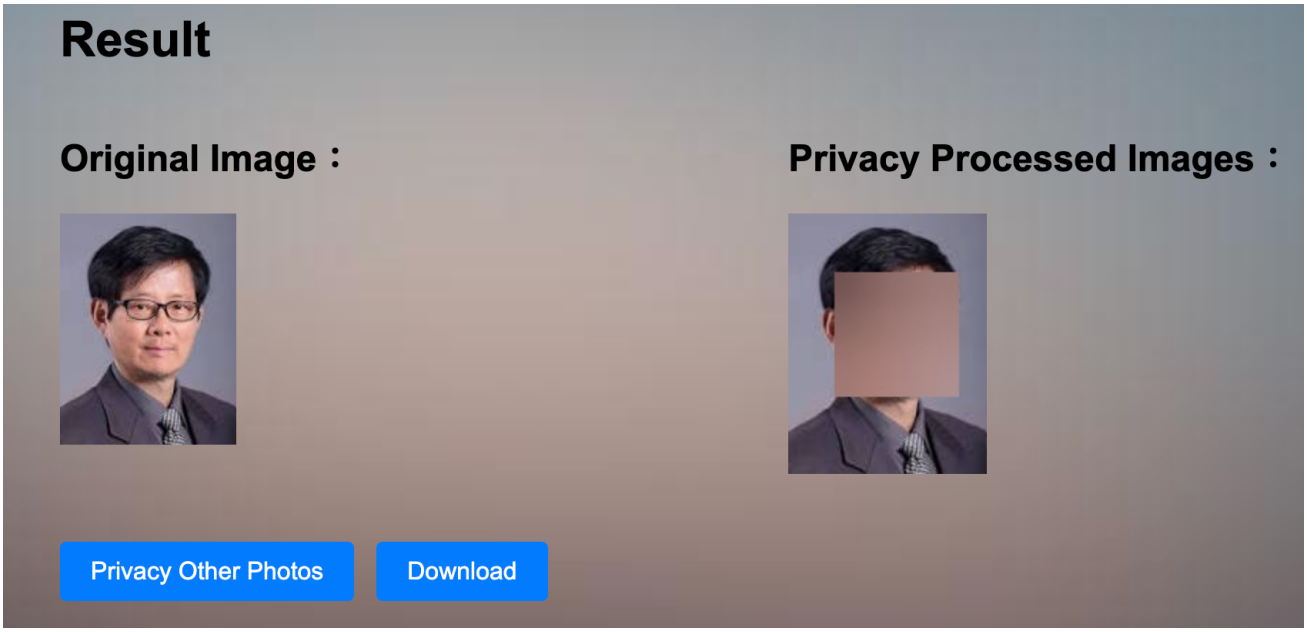
File Selection Buttons:

The first button, "選擇檔案," translates to "Choose File" and is used to open the file browser so the user can select an image file to upload.

The second field, "未選擇任何檔案," translates to "No file chosen," showing that no file has been selected yet.

Execution Button: Below the file selection area, there is a blue button labeled "Run." This button is designed to be clicked once the file has been chosen, presumably to start the image blurring process.

Result Page



The webpage displays the results of an image processing operation aimed at privacy protection. On the left, there is a section titled "Original Image" which shows the photo before any modifications have been made.

To the right, under the heading "Privacy Processed Images," there is a version of the same photo that has been altered for privacy. The individual's face has been covered with a solid color block to obscure their identity.

Below these images are two buttons. The first button, "Privacy Other Photos," suggests an option to return to the process to anonymize additional images. The second button, "Download," is likely to be used for downloading the processed image to the user's local device.

5.3 Coding

The provided code is a Python script that integrates Flask, a web framework, with OpenCV, a computer vision library. This script creates a web application for face detection and blurring in uploaded images.

1.

```
from flask import Flask, render_template, request
import cv2
import os

app = Flask(__name__)

# 加载人脸检测器
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_frontalface_default.xml')
```

from flask import Flask, render_template, request: This line imports necessary components from the Flask framework. Flask is used to create the web application instance, render_template is for rendering HTML templates, and request handles HTTP requests.

app = Flask(__name__): Initializes the Flask application.

OpenCV for Face Detection:import cv2: Imports the OpenCV library, used for image processing and computer vision tasks.

face_cascade = cv2.CascadeClassifier(...): Loads a pre-trained Haar cascade classifier for frontal face detection from OpenCV's data repository.

2.

```
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/process', methods=['POST'])
```

Web Application Routes:

@app.route('/'): Defines the root route. When accessed, it renders the index.html template.

@app.route('/process', methods=['POST']): Defines the /process route for POST requests. This route handles the image processing.

3.

```
def process():
    # 获取上传的文件
    file = request.files['imageInput']

    # 保存上传的文件
    file_path = 'static/' + file.filename
    file.save(file_path)

    # 加载图像
    image = cv2.imread(file_path)

    # 将图像转换为灰度
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # 进行人脸检测
    faces = face_cascade.detectMultiScale(gray_image, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

    # 对检测到的人脸进行模糊处理
    for (x, y, w, h) in faces:
        # 提取人脸区域
        face_roi = image[y:y+h, x:x+w]

        # 对人脸区域进行模糊处理
        blurred_face = cv2.GaussianBlur(face_roi, (99, 99), 30)

        # 将模糊处理后的人脸区域放回原图像
        image[y:y+h, x:x+w] = blurred_face

    # 保存处理后的图像
    output_path = 'static/blurred_image.jpg'
    cv2.imwrite(output_path, image)

    return render_template('result.html', input_image=file_path,
output_image=output_path)
```

Inside the process function:

The uploaded image file is retrieved and saved locally.

The image is read and converted to grayscale, a necessary step for the Haar cascade face detector.

face_cascade.detectMultiScale(...): Detects faces in the grayscale image.

Each detected face is then blurred using Gaussian blur (cv2.GaussianBlur).

The blurred face is placed back into its original position in the image.

The processed image with blurred faces is saved and its path is sent to the result.html template.

4.

```
if __name__ == '__main__':
    app.run(debug=True)
```

if __name__ == '__main__':: This conditional ensures that the server is only run if the script is executed directly, not imported as a module.

app.run(debug=True): Starts the Flask web server with debug mode enabled.

This code efficiently demonstrates the integration of web development and computer vision techniques, particularly focusing on privacy aspects by blurring faces in images. The use of Flask allows for easy and accessible deployment of the application, while OpenCV provides robust tools for image processing tasks.

6 Reflection

Reflection on Future Development for Multi-Image Processing Project

The current project, which integrates Flask with OpenCV for face detection and blurring in images, marks a significant step in the realm of privacy-centric image processing within a web application. However, reflecting on its capabilities and potential future enhancements, it becomes evident that a crucial area for development is the ability to handle multiple images simultaneously.

At present, the application is designed to process a single image per upload. This limitation, while functional for individual use cases, restricts the application's utility in scenarios where batch processing of images is required. For instance, in a real-world context, users might need to process multiple images at once for efficiency and time-saving purposes, especially in professional settings like media editing or data management.

To address this limitation, the project can be evolved to allow simultaneous uploads and processing of multiple images. This enhancement would involve modifying the Flask application to handle multiple file uploads, which can be achieved using HTML forms with multiple file selection options. The backend Python script would then need to iterate over the array of uploaded files, applying the face detection and blurring algorithm to each image.

Furthermore, this development opens up avenues for more advanced features, such as parallel processing of images to optimize performance and reduce processing time. Implementing asynchronous processing techniques or leveraging Python's multi-threading capabilities could significantly improve the application's efficiency.

Additionally, extending the functionality to include various image processing options besides blurring, such as redaction or color adjustment, could make the application more versatile and appealing to a broader user base. This expansion would also involve enhancing the user interface to allow users to select their preferred processing method for each image.

The potential to evolve the current project into a multi-image processing tool presents an exciting opportunity. It would not only enhance the application's functionality and appeal but also provide a more robust solution to the challenges of managing and editing multiple images in a privacy-conscious manner. This development aligns with the growing demand for efficient and versatile image processing tools in various sectors, including media, online content creation, and data privacy management.

7 Conclusions

This project's development of an automated face mosaic application for news photographs represents a significant advancement in journalism, addressing key issues of privacy and ethical standards. Utilizing OpenCV and TensorFlow, the system has proven to be highly efficient and accurate, surpassing traditional manual methods. This development is not only a technical achievement but also a step forward in upholding ethical journalism and safeguarding individual privacy in media.

The system's potential for future enhancements, including the capability to process multiple images simultaneously and offering a variety of image editing options, holds great promise. These anticipated improvements are expected to further enhance the system's effectiveness and adaptability, meeting the growing demands of the media industry and privacy management sectors. This project underscores the importance of continuing innovation in the intersection of technology and journalism, emphasizing the need for ethical and responsible media practices.

8 References and Citations

1. Personal Data (Privacy) Ordinance (1995). Available at: https://www.elegislation.gov.hk/hk/cap486!en-zh-Hant-HK.pdf?FROMCAPINDEX=Y (Accessed: 10 July 2023).

2. Fan, X., Zhang, F., Wang, H. and Lu, X. (2012). The system of face detection based on OpenCV. Chinese Control and Decision Conference. doi: https://doi.org/10.1109/ccdc.2012.6242980.

3. William, I., Setiadi, D.R.I.M., Rachmawanto, E.H., Santoso, H.A. and Sari, C.A. (2019). Face Recognition using FaceNet (Survey Performance Test and Comparison). IEEE Xplore. doi: https://doi.org/10.1109/ICIC47613.2019.8985786.

4. 阿叔港鐵偷拍女生.(布萊恩). 香港01. Available at: https://www.hk01.com/%E7%86%B1%E7%88%86%E8%A9%B1%E9%A1%8C/927011.

5. Williams, B. How To Blur & Pixelate Faces in Photoshop. Available at: https://www.bwillcreative.com/how-to-blur-and-pixelate-faces-in-photoshop/.

6. LearnOpenCV. OpenCV Facial Landmark Detection. Available at: https://learnopencv.com/wp-content/uploads/2015/10/Facial-Feature-Detection.jpg.

7. OpenCV. (n.d.). About. Available at: https://opencv.org/about/.

8. TensorFlow Facial Recognition. Available at: https://www.width.ai/post/tensorflow-facial-recognition.

9. Yuan, L. et al. (2017). A convolutional neural network based on tensorflow for face recognition. 2017 IEEE 2nd Advanced Information Technology Electronic and Automation Control Conference (IAEAC) [Preprint]. doi: 10.1109/iaeac.2017.8054070.

10. Khan, M. et al. (2019). Face detection and recognition using opencv. 2019 International Conference on Computing Communication and Intelligent Systems (ICCCIS) [Preprint]. doi: 10.1109/icccis48478.2019.8974493.