

SyncLab: Intelligent Image Data Generation

Final Report

**B. Tech
Computer Science and Engineering**

By

AYUSH RAJ (20BCT0168)

PRIYANSH BHAKUNI (20BCT0167)

*Under the guidance of
Prof. P. SENTHILNATHAN,
Associate Professor Grade 1,
SCOPE, VIT, Vellore*



**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
VELLORE INSTITUTE OF TECHNOLOGY**

VELLORE 632014, TAMILNADU, INDIA

April 2024

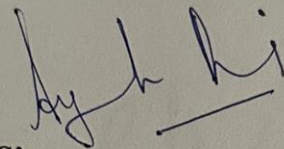
DECLARATION

I hereby declare that the thesis entitled “SyncLab: Intelligent Image Data generation” submitted by Ayush Raj (20BCT0168) and Priyansh Bhakuni (20BCT0167), for the award of the degree of Bachelor of Technology in Computer Science Engineering with specialization in Internet of Things (IOT) to VIT is a record of bonafide work carried out by me under the supervision of Prof. P. Senthilnathan Sir.

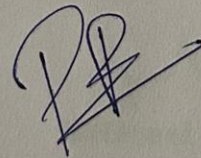
I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date : 08/05/2024



Signature of the Candidate



Signature of the Candidate

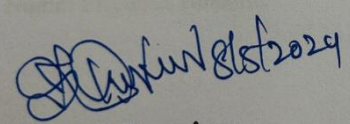
CERTIFICATE

This is to certify that the thesis entitled “**SyncLab: Intelligent Image Data Generation**” submitted by **Ayush Raj (20BCT0168) SCOPE and Priyansh Bhakuni (20BCT0167) SCOPE**, from VIT for the award of the degree of *Bachelor of Technology in Computer Science and Engineering with specialization in Internet of Things (IOT)*, is a record of bonafide work carried out by them under my supervision during the period, **03. 01. 2024 to 07.05.2024**, as per the VIT code of academic and research ethics.

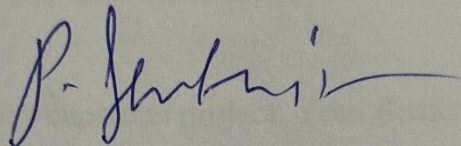
The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

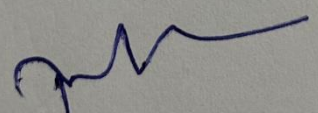
Date : 08/05/2024



Internal Examiner



Signature of the Guide



External Examiner

Sharmila Banu K

SCOPE

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to all those who contributed to the successful completion of our capstone project SyncLab.

First and foremost, we extend our heartfelt appreciation to VIT University for providing the necessary resources, support, and guidance throughout the project duration.

We would like to extend a special thanks to *Prof. P. Senthilnathan* for his guidance, expertise, and mentorship which has been invaluable throughout the project. His deep understanding of the subject matter, insightful advice, and unwavering support have played a significant role in shaping the direction of our work and overcoming challenges along the way.

We are deeply thankful to our project team members for their dedication, hard work, and collaboration in turning our vision into reality. Their expertise, creativity, and commitment have been instrumental in overcoming challenges and achieving our project objectives.

We also acknowledge the invaluable assistance and cooperation received from various stakeholders, including technical experts, and end-users. Their feedback, insights, and participation have greatly enriched the project outcomes and ensured its relevance and effectiveness.

Furthermore, we extend our appreciation to the research community and industry partners for their contributions and insights that have informed our project implementation and outcomes.

Last but not least, we express our gratitude to our families, friends, and colleagues for their unwavering support and encouragement throughout this journey.

Thank you to everyone involved for their invaluable contributions to the capstone project. Your dedication and collaboration have been integral to its success.

EXECUTIVE SUMMARY

The production of test data for various vision AI solutions poses a significant challenge due to the laborious and time-consuming nature of manually browsing websites to gather image data that meets specific constraints such as resolutions, actions, and angles. In this project, we propose a novel solution that leverages Web Scrapping, NLP, Image Tagging, and Image Augmentation Techniques to facilitate the creation of a rich and accurate dataset tailored to the specific use cases. By automating the image dataset creation process, we aim to eliminate the manual labor involved, streamlining the workflow, and improving overall efficiency. Furthermore, we employ emotion detection model and yolov5 model of single and multiple object detection (Image Tagging Technique), built upon Complex Computer Vision techniques, to evaluate the generated dataset. The success of these models relies heavily on the availability of a precise and diverse dataset, highlighting the importance of our automated solution in achieving optimal performance and superior results. Additionally, we implement an image augmentation strategy using techniques such as rotation, flipping, scaling and much more to further diversify the dataset and enhance model robustness. Through these efforts, we aim to address the challenges associated with image dataset generation and augmentation, paving the way for advancements in vision AI solutions.

Keywords: Emotion detection, Computer vision, Artificial intelligence, Image dataset generation, Natural language processing, Image augmentation, Dataset diversity, Emotion recognition

TABLE OF CONTENTS

CONTENTS		Page No.
	Acknowledgement	i.
	Executive Summary	ii.
	Table of Contents	iii.
	List of Figures	iv.
	List of Tables	v.
1	INTRODUCTION	09 - 11
	1.1 Project Domain Description	10
	1.2 Problem Statement	11
	1.3 Objectives	11
2	LITERATURE SURVEY	12 - 15
	2.1 Major Technology Used	12 - 15
	2.2 Outcomes	12 – 15
	2.3 Drawbacks	12 - 15
3	TECHNICAL SPECIFICATION	16 - 22
	3.1 Requirements	16 - 17
	3.2 Feasibility Assessment	17 – 19
	3.3 System Specification	20
	3.3.1 Hardware Specification	20
	3.3.2 Software Specification	21 - 22
4	DESIGN APPROACH AND DETAILS	23 - 25
	4.1 System Architecture Diagram	23
	4.2 Data Flow Diagram	24-25
5	SCHEDULE AND PROJECT DESCRIPTION	26 - 35
	5.1 Gantt Chart	26 - 28
	5.2 Module Description	29
	5.2.1 NLP Query Expansion	29

SyncLab: Intelligent Image Data Generation System

	5.2.2 Web Scrapping	30
	5.2.3 Image Tagging Module	31 - 32
	5.2.4 Image Augmentation	33
	5.3 Testing	34
	5.3.1 Model Testing	34 - 35
	5.3.2 Post Processing Testing	35
6	PROJECT DEMONSTRATION	36 - 38
7	RESULT & DISCUSSION	39 - 40
8	CONCLUSION AND FUTURE WORK	41
9	REFERENCES	42 - 43

LIST OF FIGURES

FIGURE 1 : THE BASIC WORKFLOW SYSTEM.....	9
FIGURE 2 : SURFING ON DIFFERENT REGIONS (USA AND INDIA) GIVES DIFFERENT RESULTS.....	18
FIGURE 3 : GIVING DIFFERENT TYPES OF QUERIES GIVES VARIED RESULTS FOR THE DATASET	19
FIGURE 4 : SYSTEM ARCHITECTURE DIAGRAM	23
FIGURE 5 : THE DATA FLOW PROCESS FOR QUERY EXPANSION	24
FIGURE 6 : WEB SCRAPING PROCESS.....	24
FIGURE 7 : IMAGE TAGGING AND FILTERING PROCESS	25
FIGURE 8 : DATA AUGMENTATION (POST PROCESSING).....	25
FIGURE 9 : GANTT CHART (PROJECT SCHEDULE)	26
FIGURE 10 : CODE SNIPPET FOR NLP QUERY EXPANSION	29
FIGURE 11 : CODE SNIPPET FOR WEB SCRAPING ALGORITHM.....	30
FIGURE 12 : CODE SNIPPET FOR MODEL TRAINING FOR IMAGE TAGGING	31
FIGURE 13 : CODE SNIPPET FOR EMOTION DETECTION MODEL.....	32
FIGURE 14 : CODE SNIPPET FOR POST PROCESSING	33
FIGURE 15 : BACKEND RUNNING CODE FOR QUERY EXPANSION AND QUERY GENERATION (STEP 1)	34
FIGURE 16 : IMAGE SCRAPING	34
FIGURE 17 : RUNNING OF IMAGE TAGGING PROCESS IN BACKEND	35
FIGURE 18 : DATASET GENERATED AFTER POST-PROCESSING	35
FIGURE 19 : FRONTEND WEBSITE FOR AUTOMATIC DATASET GENERATION	36
FIGURE 20 : A SCREENSHOT OF THE WEBPAGE.....	36
FIGURE 21 : FINAL GENERATED DATASET STORED IN A FOLDER	37
FIGURE 22 : EMOTION DETECTION MODEL TESTING (IMAGE TAGGING).....	37
FIGURE 23 : YOLOV5 OBJECT DETECTION MODEL TESTING (IMAGE TAGGING).....	38
FIGURE 24 : COMPARING MODEL ACCURACY BETWEEN OUR PROPRIETARY DATASET AND PUBLICLY AVAILABLE DATASETS.	39
FIGURE 25 : COMPARING MODEL ACCURACY BETWEEN OUR PROPRIETARY & PUBLICLY AVAILABLE DATASETS(YOLOV5 MODEL)	40
FIGURE 26 : RESULTS FOR YOLOV5 BASED DATASET FOR F1 CARS.....	40

LIST OF TABLES

TABLE 1 : Literature Survey (with Outcomes and Drawbacks).....	12
--	----

1. INTRODUCTION

In recent years, the rapid progress in artificial intelligence (AI), especially in the domain of computer vision, has brought about a transformative shift across numerous sectors by endowing machines with the ability to perceive, interpret, and comprehend visual data. This breakthrough has led to ground-breaking applications spanning from autonomous vehicles to medical diagnostics, fundamentally altering the way we interact with technology. However, at the core of these advancements lies a critical dependency on the quality and diversity of datasets utilized to train and evaluate AI models. Specifically, in vision-based AI solutions such as emotion detection systems, the need for comprehensive and varied datasets is paramount. Yet, the process of generating such datasets presents formidable challenges that cannot be overlooked.

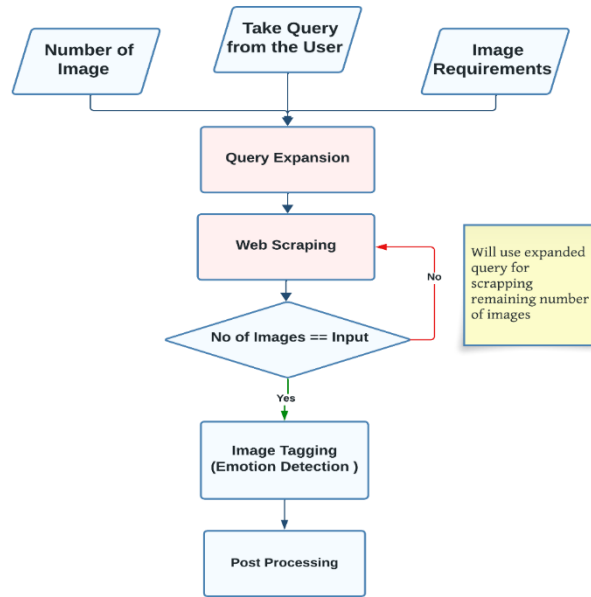


Figure 1 : The Basic Workflow System

The task of curating datasets for vision-based AI applications is far from straightforward. It involves sourcing, collecting, and annotating vast amounts of visual data, each meeting stringent criteria regarding resolution, context, diversity, and relevance. This laborious and time-intensive process is further compounded by the intricacies associated with emotion detection, which demands nuanced and diverse visual stimuli to accurately capture the spectrum of human emotions. Moreover, ensuring the quality and representativeness of the dataset is paramount to the robustness and generalization capabilities of the AI models. Thus, while the promise of AI in revolutionizing various industries is undeniable, its realization hinges on overcoming the challenges inherent in dataset creation and curation for vision-based applications.

The production of comprehensive and diverse test data for vision AI solutions is a time-consuming and labor-intensive task. Manually browsing through numerous websites to gather image data that meets specific criteria such as resolution, objects, actions, and angles is not only arduous but also prone to inconsistencies and biases. While generic search engines provide a starting point, they often fall short in catering to the precise requirements of the use case under development. This limitation underscores the need for a more efficient and targeted approach to web scraping and image extraction.

To address these challenges, this project proposes a novel approach leveraging natural language processing (NLP) models to expand search queries and generate a rich dataset tailored to the requirements of vision AI applications, particularly emotion detection systems. By automating the process of dataset creation, our solution aims to eliminate the manual labor involved in curating image datasets while ensuring adherence to specific criteria. Additionally, the project emphasizes the importance of dataset diversity and precision in training AI models, especially in applications like emotion detection, where nuanced visual cues play a crucial role.

Overall, this project aims to advance the field of computer vision and AI by providing a comprehensive and diverse dataset generation solution powered by NLP and image augmentation techniques. The resulting datasets will not only facilitate the development and evaluation of emotion detection systems but also contribute to the broader goal of advancing AI technologies for various applications.

1.1 Project Domain Description

In the domain of information technology, the demand for high-quality image datasets persists, necessitating efficient methods for their generation. Manual, semi-automatic, and automatic approaches offer varying degrees of effectiveness and scalability, with automated methods increasingly garnering attention for their potential in addressing optimization challenges.

1. Automated Dataset Generation –

In the IT field involves full-automatic and semi-automatic approaches. Schroff et al. proposed a fully automated system utilizing search engine results and image re-ranking, while Yao et al. expanded this method with query expansion techniques. Zink demonstrated the effectiveness of semi-automatic generation by combining manual dataset construction with web scraping, highlighting increased accuracy with larger image sets.

2. NLP Query Expansion –

A pivotal process in information retrieval, involves augmenting search queries with synonyms or related terms to enhance retrieval effectiveness. By leveraging techniques such as semantic analysis and word embeddings, this method enriches query semantics, facilitating more precise document retrieval in natural language processing tasks

3. Web Scraping -

In the realm of web scraping, Selenium and Google Image Scraper represent prominent tools for automating data extraction tasks. Leveraging Selenium's browser automation capabilities, coupled with the functionality of specialized scrapers like the Google Image Scraper, enables systematic retrieval of image data from online sources, contributing to the augmentation of image datasets for various machine learning applications.

4. Image Augmentation -

In the domain of web scraping, Image Augmentation serves as a pivotal technique for enriching datasets with varied and augmented images. By employing methods like rotation, scaling, and flipping, Image Augmentation enhances the diversity and robustness of collected image data, facilitating improved model generalization and performance in machine learning tasks.

5. Image Tagging -

In the field of computer vision, Image Tagging is a fundamental process involving the automatic assignment of descriptive labels or tags to images. Leveraging techniques such as deep learning and image recognition algorithms, Image Tagging enhances image searchability, organization, and retrieval in large-scale image datasets, facilitating efficient content management and analysis.

1.2 Problem Statement

- i. Production of test data for different vision AI solutions needs a great deal of time and work.
- ii. It is a laborious manual task to browse through many web sites in search of certain image data with constraints like resolutions, objects, actions, and angles.
- iii. The challenge lies in maximizing the effectiveness and increasing the efficiency of web scraping and image extraction using only one search query.
- iv. The generic search engine provides a base to leap forward but at the same time can't cater rightly to specific requirements of the use case under development.

1.3 Objectives

- i. **Query Expansion Techniques:**
 - With the help of NLP models, we will be able to generate expanded queries to generate a rich dataset making it more accurate for the use cases.
- ii. **Image Scraper Methods:**
 - We will provide an automated solution which will eliminate manual labor of creating image datasets while adhering to the specific requirements.
- iii. **Complex Model Assessment for Dataset Validation:**
 - In the evaluation of our dataset, we employ a sophisticated Emotion Detection model founded on Complex Computer Vision and Artificial Intelligence. This model optimally utilizes the dataset, necessitating a precise and diverse dataset for optimal performance and superior results.
- iv. **Data Augmentation:**
 - Implement an image augmentation strategy using techniques like rotation, flipping, and scaling to diversify the image dataset and improve model robustness.

2. LITERATURE SURVEY

The literature survey on automatic image generation encompasses diverse methodologies and technologies aimed at advancing the field. Various approaches have been explored, ranging from web scraping and text processing to sophisticated algorithms such as cosine similarity and neural networks. Research papers delve into topics like web scraping optimization, query expansion strategies, and the integration of advanced technologies like natural language processing and deep learning. Each study offers insights into the challenges, innovations, and potential applications of automatic image generation across different domains, highlighting the continuous evolution and interdisciplinary nature of this research area.

Table 1 : Literature Survey (with Outcomes and Drawbacks)

S. No.	Title and Year	Major Technologies Used	Outcomes	Drawbacks
1	Image search optimization with web scraping, text processing and cosine similarity algorithms, 2020	Tokenization, Cosine Similarity	Highest accuracy value is obtained if the name of the images being searched for is general and the number of images is limited to 20 images.	The results obtained using cosine similarity technique only reach 60% accuracy. The number of images is very less to get highest accuracy.
2	Web Scraping, 2017	Urllib2, Selenium, BeautifulSoup	The paper discusses the sequential steps involved in web scraping, the tools and frameworks used, and the integration of advanced technologies like computer visual analytics and natural language processing.	Lack of in-depth exploration into the ethical implications and legal challenges of web scraping and quantitative results.
3	Web scraping technologies in an API world, 2019	HTML parsing, Regular expressions,	The research outcomes demonstrate the	Reliance on web scraping as a primary method for data retrieval in biomedical

		XPath, CSS selector syntax	practical application of web scraping in the biomedical domain, particularly in scenarios where data retrieval from public websites lacking APIs is essential. The study showcases the utilization of web scraping to compile information on antimicrobial peptides and to build bioinformatics meta-servers.	applications. While web scraping can be effective in scenarios where APIs are lacking or inadequate, it also presents challenges such as dependency on website structures that may change over time, leading to potential disruptions in data extraction processes.
4	Web Scraping Tool for Newspapers and Images Data Using Jsonify, 2022	Python, Scrapy, Flask	The developed scraping tool aims to improve data extraction by collecting, merging, categorizing, and managing information stored in JSON format. The tool accommodates changes in website layouts and formats, crucial for marketing, pricing research, and competitor analysis.	Lacks a comprehensive evaluation or comparison of their effectiveness, performance, and limitations.
5	Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering, 2019	Query expansion Question-answering Information retrieval	Query Expansion (QE) approach, integrating lexical resources and word embeddings, enhances the precision of Information Retrieval-based Question Answering (QA) systems.	Drawbacks include the current approach's limitations in contextualization, generalization to diverse document collections, reliance on advanced ranking functions, optimization of term weighting strategies, and application only to specific syntactic categories.

6	Improving query expansion strategies with word embeddings	word embeddings, re-ranking, query expansion	The paper explores the utility of word embeddings, introducing IDF average word embeddings for query representation, achieving superior results in TREC benchmark datasets.	The drawback lies in the absence of a predefined method for optimizing query representation, suggesting a potential need for machine learning-based approaches.
7	Improving Information Retrieval Results for Persian Documents using FarsNet	Information Retrieval, Query Expansion, FarsNet	The paper introduces a novel query expansion method utilizing FarsNet, leveraging synonymy relations to enhance information retrieval systems and yielding a 9% improvement in Mean Average Precision (MAP).	A potential drawback is the unspecified challenges and limitations in modifying synsets in FarsNet 3.0 to enhance precision for Information Retrieval in future research.
8	BERT-QE: Contextualized Query Expansion for Document Re-ranking	TREC Robust04 and GOV2 test	The paper introduces a novel query expansion model, BERT-QE, leveraging BERT's contextualized models to select relevant document chunks, outperforming BERT-Large models in TREC Robust04 and GOV2 test collections.	The potential drawback is the need for future work to enhance the efficiency of BERT-QE, considering computational cost optimization and combining it with pre-computation techniques.
9	Detection and Recognition of Human Emotion using Neural Network	facial detection, Haar Cascade features, Viola Jones, Neural Network	The project aims to develop a robust emotion detection and recognition system using facial features and deep neural	A potential drawback is the reliance on universal emotions, which might not fully capture the diversity and complexity of

			networks, with applications in security, surveillance, and robotics.	individual emotional expressions. Additionally, the system's effectiveness may be influenced by variations in lighting conditions and facial expressions.
10	Development of a Real-Time Emotion Recognition System Using Facial Expressions and EEG based on machine learning and deep neural network methods	Face emotion recognition Virtual markers LSTM EEG emotion Detection	The study introduces a real-time emotion recognition system for physically disabled individuals and Autism children, achieving high accuracy (99.81%) with facial landmarks and CNN, while EEG signals exhibit a slightly lower accuracy (87.25%) using LSTM classification.	The potential drawback is the need for future work to enhance system precision by collecting more diverse data and improving feature extraction from EEG signals.

3. TECHNICAL SPECIFICATIONS

Hence, the essential prerequisites for the project entail the development of a robust and adaptable system capable of fulfilling user requirements by delivering precise datasets. This system must seamlessly integrate functional components such as web scraping, NLP query expansion, image tagging, and image augmentation, leveraging Python and its associated libraries proficiently. Additionally, non-functional requisites encompass proficiency in web development frameworks like Flask for frontend creation. Furthermore, the inclusion of an emotion detection model serves to validate the system's accuracy through comprehensive dataset testing.

3.1 Requirements

Automated Dataset Generation –

Automated dataset generation employs two main strategies: fully automatic and semi-automatic. Fully automatic systems, like those proposed by Schroff et al., rely on search engine queries and subsequent re-ranking based on text or metadata for complete automation. Yao et al. expanded on this by integrating query expansion techniques for broader search query coverage. Overall, automated dataset generation facilitates efficient curation of large image datasets. While fully automatic methods prioritize algorithmic efficiency, semi-automatic approaches leverage human expertise for quality refinement. These methods drive innovation in image dataset curation, advancing fields such as machine vision and natural language processing. Continued research in automated dataset generation promises further enhancements in dataset quality and scalability, benefiting various domains reliant on large-scale image data analysis.

NLP Expansion Techniques –

Recent advancements in semantic word relations have led to notable developments in query expansion techniques. WordNet, a prominent resource in this domain, offers a comprehensive English lexicon with numerous inter-word connections. Conversely, Word2Vec represents a newer approach, utilizing corpora like GBNC to generate word vectors, indicating semantic relationships. A comparative analysis highlights Word2Vec's superiority in handling a broader word range and its adeptness in detecting homonyms, meronyms, and hypernyms compared to WordNet.

Web Scraping –

Web scraping, as aptly defined by Zhao, involves extracting data from the World Wide Web for storage or analysis. However, its usage raises concerns like copyright infringement and potential DDOS attacks if not executed properly. Sirisuriya detailed nine scraping techniques, with the most common being program-based scraping. Over 20 readily available scraping software versions exist, each tailored to specific dataset needs. Comparative studies by Glez-Pena and Sirisuriya have assessed these tools. Additionally, toolkits allow users to customize scraping applications, as demonstrated by Upadhyay.

Dataset Quality –

Assessing dataset quality is complex, as Pipino et al. identified 16 dimensions, but found no universal metric. Commercial sources, however, highlight five key factors: characteristics, accuracy, completeness, relevance, and timeliness. Image datasets emphasize high resolution and uniqueness, with a minimum of 1,000 images recommended for reasonable quality. Zink's research noted a correlation between dataset size and accuracy in image recognition. Unique images are crucial to prevent bias, as noted by Rosebrock and Hofesmann.

- i. In the context of *image tagging*, quality criteria include characteristics like how images are measured, accuracy of details, completeness, relevance, and timeliness.
- ii. High-resolution images with minimal manipulation and a minimum dataset size of 1,000 images are essential for effective tagging.
- iii. Zink's findings underscore the importance of dataset size for accurate tagging.
- iv. Similarly, in *image augmentation*, ensuring high-resolution and unique images is paramount.
- v. Image quality should be preserved or enhanced during augmentation processes, aligning with the characteristics outlined for dataset quality assessment.
- vi. A diverse and sufficiently large dataset contributes to effective augmentation, as observed in Zink's research correlating dataset size with recognition accuracy.
- vii. Unique images are pivotal to mitigate bias during augmentation, as emphasized by Rosebrock and Hofesmann.

3.2 Feasibility Assessment –

Image Dataset Collection:

- ❖ An automated web scraper must consider key dataset characteristics, including relevance, timeliness, quality, and reliability.
- ❖ Completeness, however, remains challenging to assess automatically.
- ❖ Increasing image quantity improves dataset completeness, achievable by integrating multiple search engines and employing query expansions.
- ❖ Dataset accuracy evaluation can be facilitated using state-of-the-art neural networks.

Regional Considerations:

- ❖ Regional biases in search results necessitate strategies to ensure consistent query outcomes.
- ❖ Utilizing VPNs or supporting queries can mitigate regional differences and enhance search result consistency.

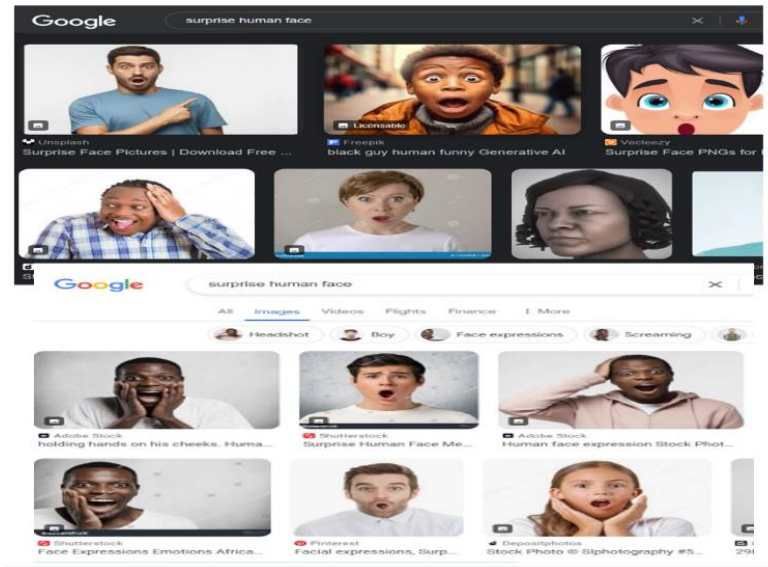


Figure 2 : Surfing on different regions (USA and India) gives different results

Query Expansion:

- ❖ Word2Vec models offer an optimal approach for query expansion, establishing semantic links between words.
- ❖ While synonym expansions risk unrelated queries, hypernym expansions yield more detailed image results.

SyncLab: Intelligent Image Data Generation System

- ❖ Manual intervention may be necessary to verify hypernym links and ensure query expansion accuracy.

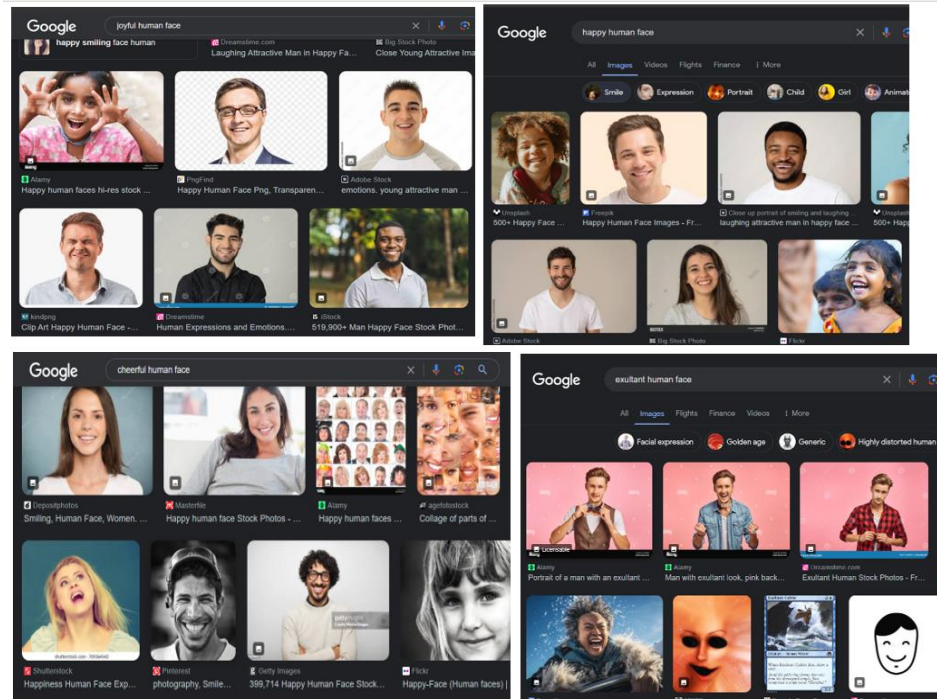


Figure 3 : Giving different types of queries gives varied results for the dataset

Copyright Compliance:

- ❖ Given evolving copyright laws, automated web scrapers should prioritize using public data and APIs to mitigate infringement risks.
- ❖ Implementing measures to discard collected image data after immediate use reduces copyright infringement potential.
- ❖ Users should have the option to retain image data locally, requiring an active selection.

By addressing these considerations, an automated image generation process via web scraping demonstrates feasibility, leveraging advanced techniques and compliance measures to ensure accurate and legally compliant dataset collection and processing.

3.3 System Specifications –

3.3.1 Hardware Specification –

The hardware requirements for running the project have been assessed to ensure optimal performance across various systems. The project is designed to operate efficiently on a range of hardware configurations, including both high-end and lower-end systems. Below is the hardware specification recommended for running the project:

Minimum Hardware Requirements:

- **Processor:** Intel Core i3 or equivalent
- **Memory (RAM):** 4 GB
- **Graphics Card:** Integrated graphics or dedicated GPU with at least 1 GB VRAM
- **Storage:** At least 100 MB of available disk space

Recommended Hardware for Optimal Performance:

- **Processor:** Intel Core i5 or equivalent
- **Memory (RAM):** 8 GB or higher
- **Graphics Card:** Dedicated GPU with at least 2 GB VRAM (e.g., NVIDIA GeForce GTX 1050 or equivalent)
- **Storage:** SSD (Solid State Drive) for faster read/write operations

System Compatibility:

The project has been tested and verified to run seamlessly on the following system configuration:

- **System Type:** x64-based PC
- **Processor:** Intel Core i7-10870H CPU @ 2.20GHz (or equivalent)
- **Memory (RAM):** 8.00 GB
- **Operating System:** Windows 10 (Version 22H2)
- **Graphics Card:** NVIDIA GeForce GTX 1650 4GB

It is important to note that while the project is optimized for the recommended hardware configuration, it is also designed to function effectively on systems with lower specifications. However, performance may vary based on the hardware capabilities of the system.

3.3.2 Software Specification –

The software requirements for the project consist of pre-requisites and the utilization of various Python packages and libraries for query expansion, web scraping, and post-processing tasks. The project is designed to operate on systems with Google Chrome installed, leveraging specific Python libraries to achieve its functionalities. Below are the detailed software specifications:

Pre-requisites:

1. Google Chrome:

- ❖ The project requires Google Chrome to be installed on the system to execute web scraping tasks effectively.

2. Python3 Packages:

- ❖ **selenium==3.141.0**: Utilized for automating web browser interactions, enabling efficient web scraping operations.
- ❖ **requests==2.25.1**: Used for sending HTTP requests and handling responses during web scraping tasks.
- ❖ **pillow==9.0.1**: Employed for image post-processing tasks, facilitating image manipulation and enhancement.

3. Operating System:

- ❖ The project is primarily tested and developed for Windows OS. Compatibility with other operating systems is not guaranteed.

Query Expansion:

- ❖ **nltk**: The Natural Language Toolkit (NLTK) library is utilized for query expansion, providing various functionalities such as text preprocessing, spelling correction, and synonym extraction.

Web Scraping:

- ❖ **Selenium**: The Selenium library is employed for web scraping tasks, allowing automated browsing and interaction with web elements.
- ❖ **WebDriver**: The WebDriver module of Selenium enables the instantiation and control of web browsers for scraping purposes.
- ❖ **urllib**: Utilized for sending HTTP requests and handling URLs during web scraping operations.
- ❖ **requests**: Used in conjunction with Selenium for making HTTP requests and managing responses during the scraping process.

Post-Processing:

- ❖ **Pillow:** The Pillow library is utilized for post-processing of images retrieved during web scraping. It provides functionalities for image manipulation, including resizing and cropping.

The project's software requirements ensure the seamless execution of query expansion, web scraping, and post-processing tasks, thereby enabling the generation and tagging of image data efficiently.

4.DESIGN APPROACH AND DETAILS

4.1 System Architecture Diagram –

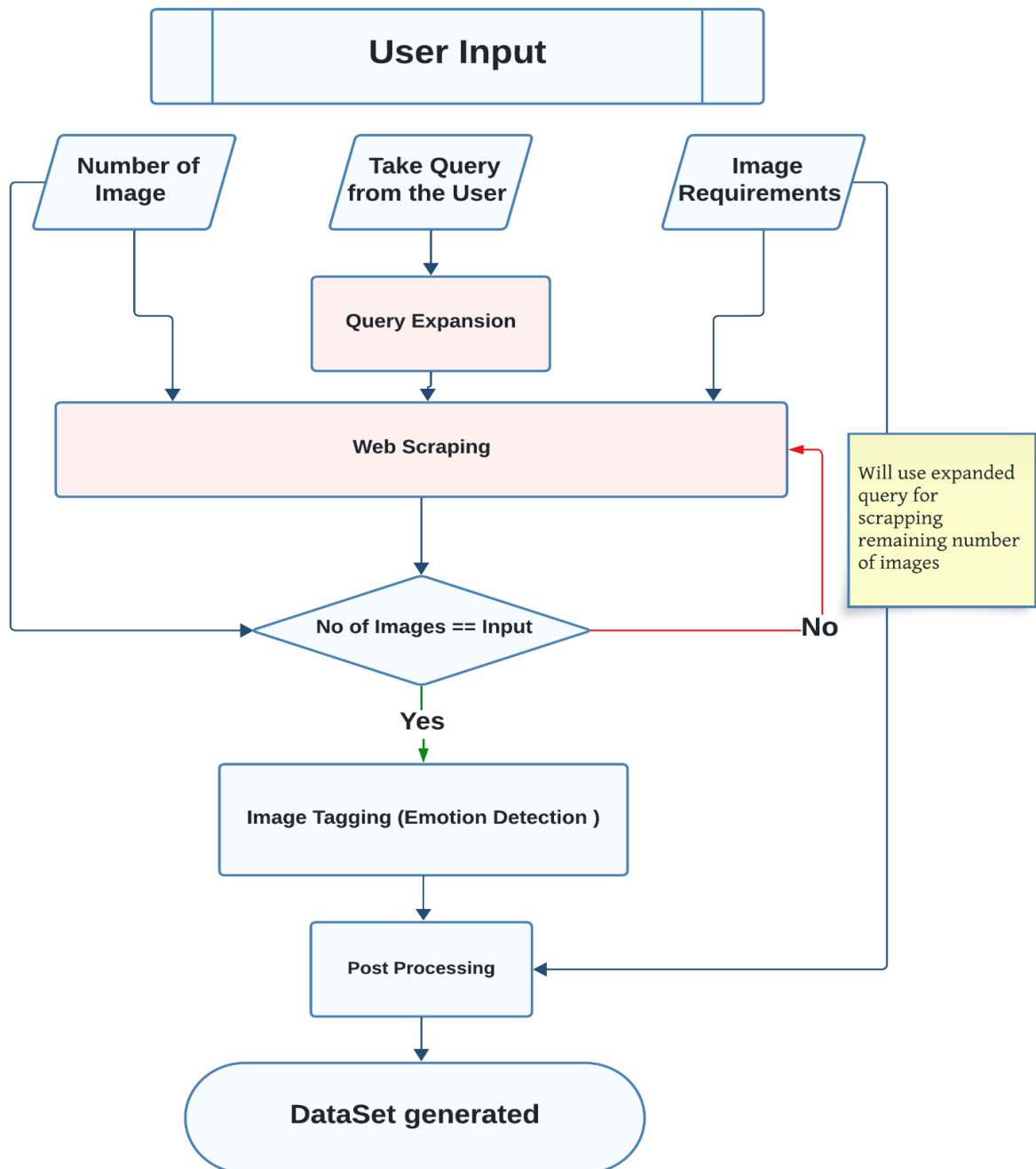


Figure 4 : System Architecture Diagram

4.2 Data Flow Diagram –

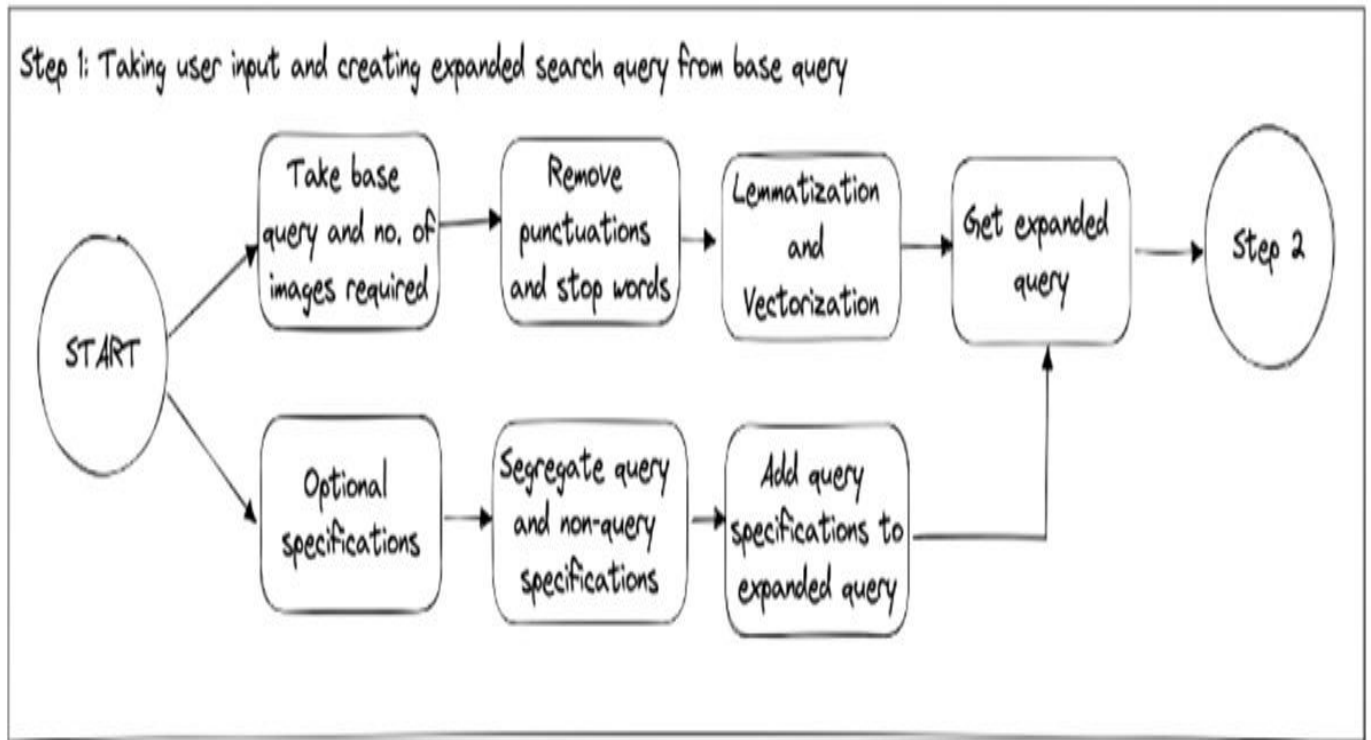


Figure 5 : The Data Flow Process for Query Expansion

Step 2: Obtaining images using web scrapping

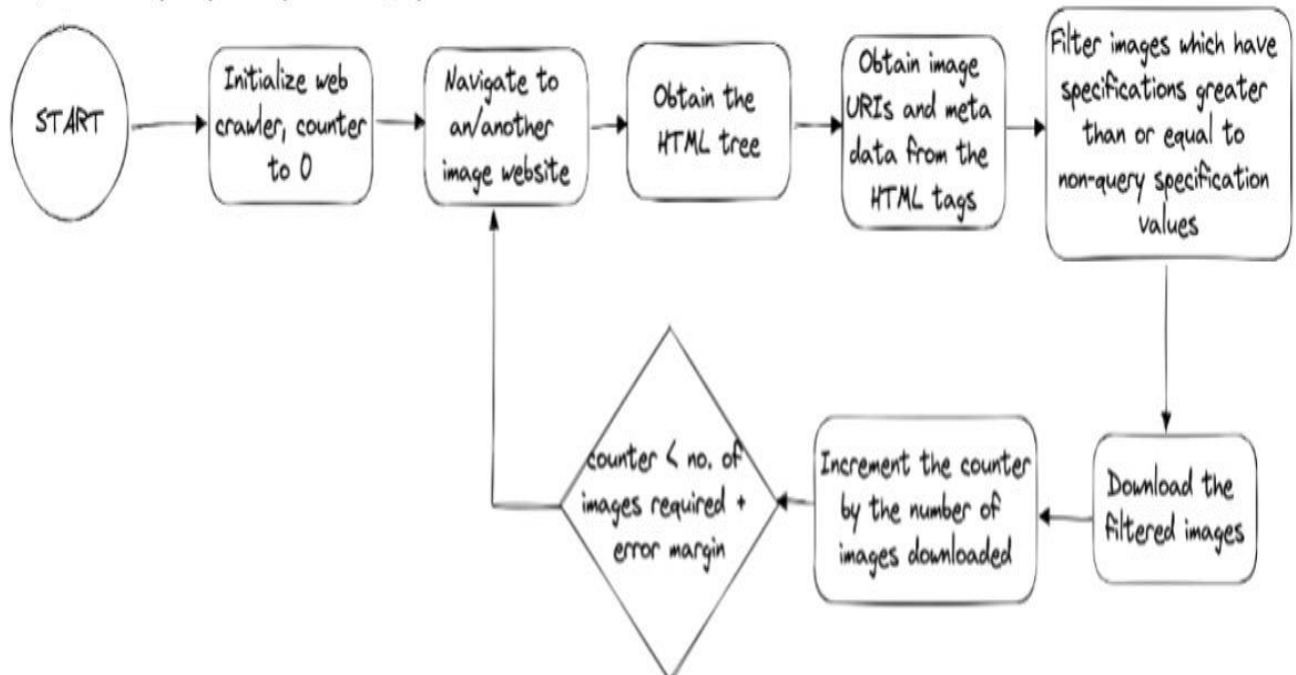


Figure 6 : Web Scrapping Process

Step 3: Image tagging and filtering matches with the query

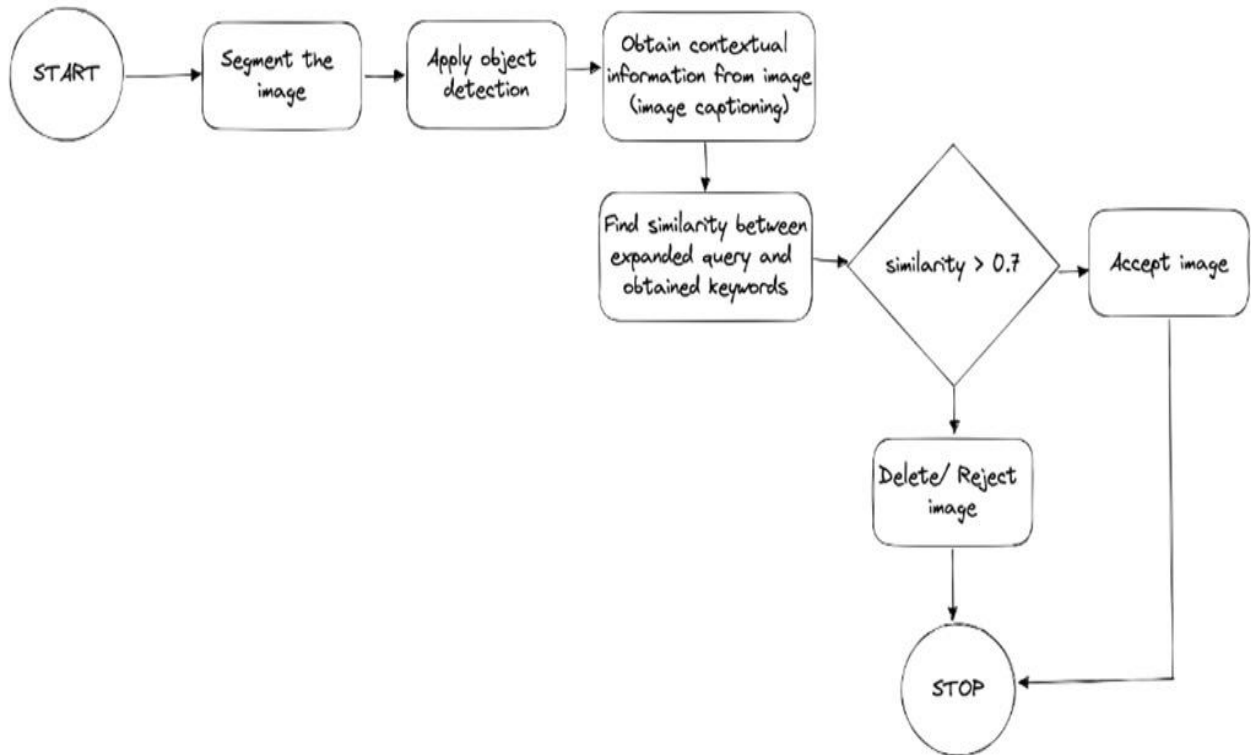


Figure 7 : Image Tagging and Filtering process

Step 4: Post-processing to match user specifications and finalizing dataset

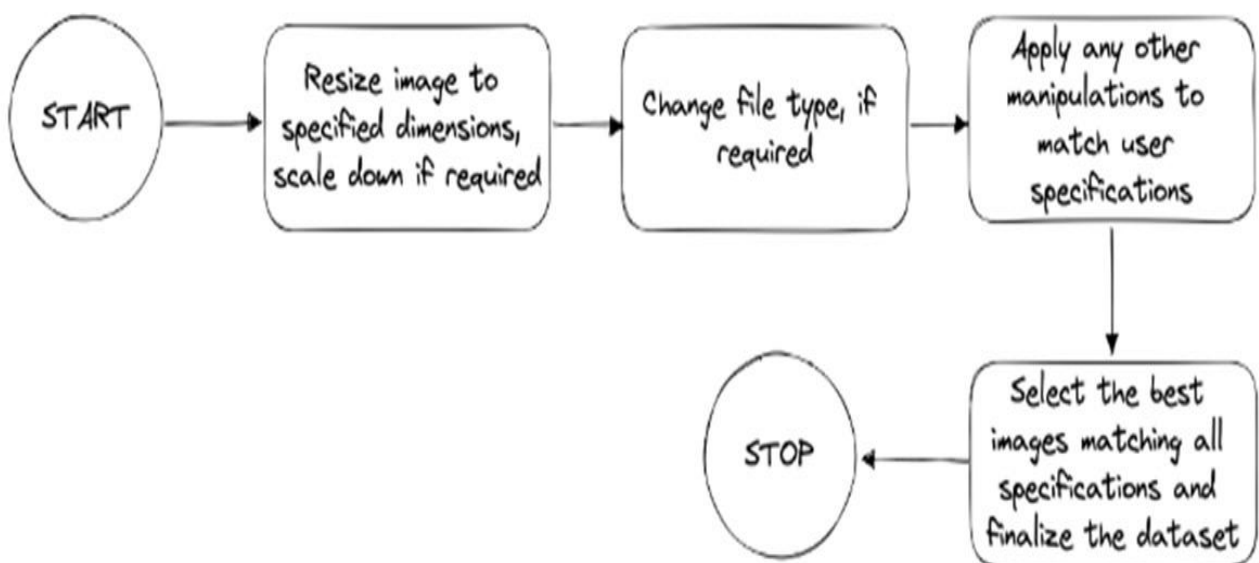


Figure 8 : Data Augmentation (Post Processing)

5. SCHEDULE AND PROJECT DESCRIPTION

5.1 Gantt Chart –

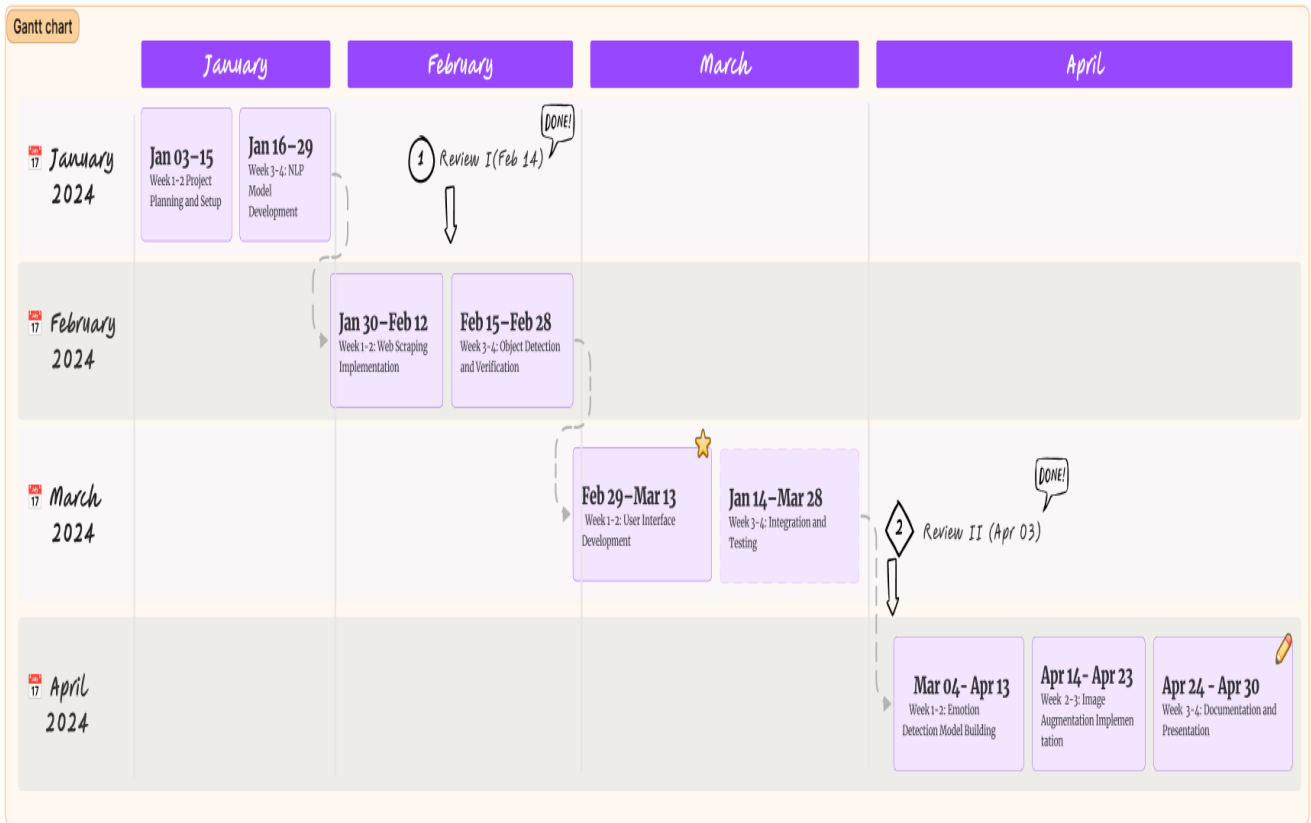


Figure 9 : Gantt Chart (Project Schedule)

January 2024:

❖ Week 1-2: Project Planning and Setup

- Define project scope, objectives, and milestones.
- Set up development environment and version control.
- Assign roles and responsibilities within the team.
- Begin initial research on NLP models and web scraping techniques.

❖ Week 3-4: NLP Model Development

- Research and select appropriate NLP models for query expansion.
- Develop and train NLP model to generate expanded search queries.
- Test NLP model's performance on sample queries and refine as needed.

February 2024:

❖ Week 1-2: Web Scraping Implementation

- Design and implement web scraping functionality to fetch images from Google based on expanded queries.
- Ensure proper handling of various constraints such as image resolution, objects, actions, and angles.
- Test scraping functionality with diverse queries and refine as necessary.

❖ Week 3-4: Object Detection and Verification

- Integrate object detection algorithms to verify the presence of required objects in scraped images.
- Develop algorithms to assess relevance of images to user query.
- Implement post-processing techniques to enhance dataset quality and match user specifications.

March 2024:

❖ Week 1-2: User Interface Development

- Design and develop user interface for specifying search queries and image requirements.
- Implement functionality to interact with backend services for query expansion, web scraping, and post-processing.

❖ Week 3-4: Integration and Testing

- Integrate NLP models, web scraping, object detection, and post-processing modules into a cohesive application.
- Conduct comprehensive testing to ensure all components work together seamlessly.
- Gather feedback from test users and refine application based on their input.

April 2024:

❖ Week 1-2: Emotion Detection Model Development

- Research and select appropriate computer vision techniques for emotion detection.
- Collect and preprocess emotion dataset for model training.
- Develop and train emotion detection model using state-of-the-art techniques.

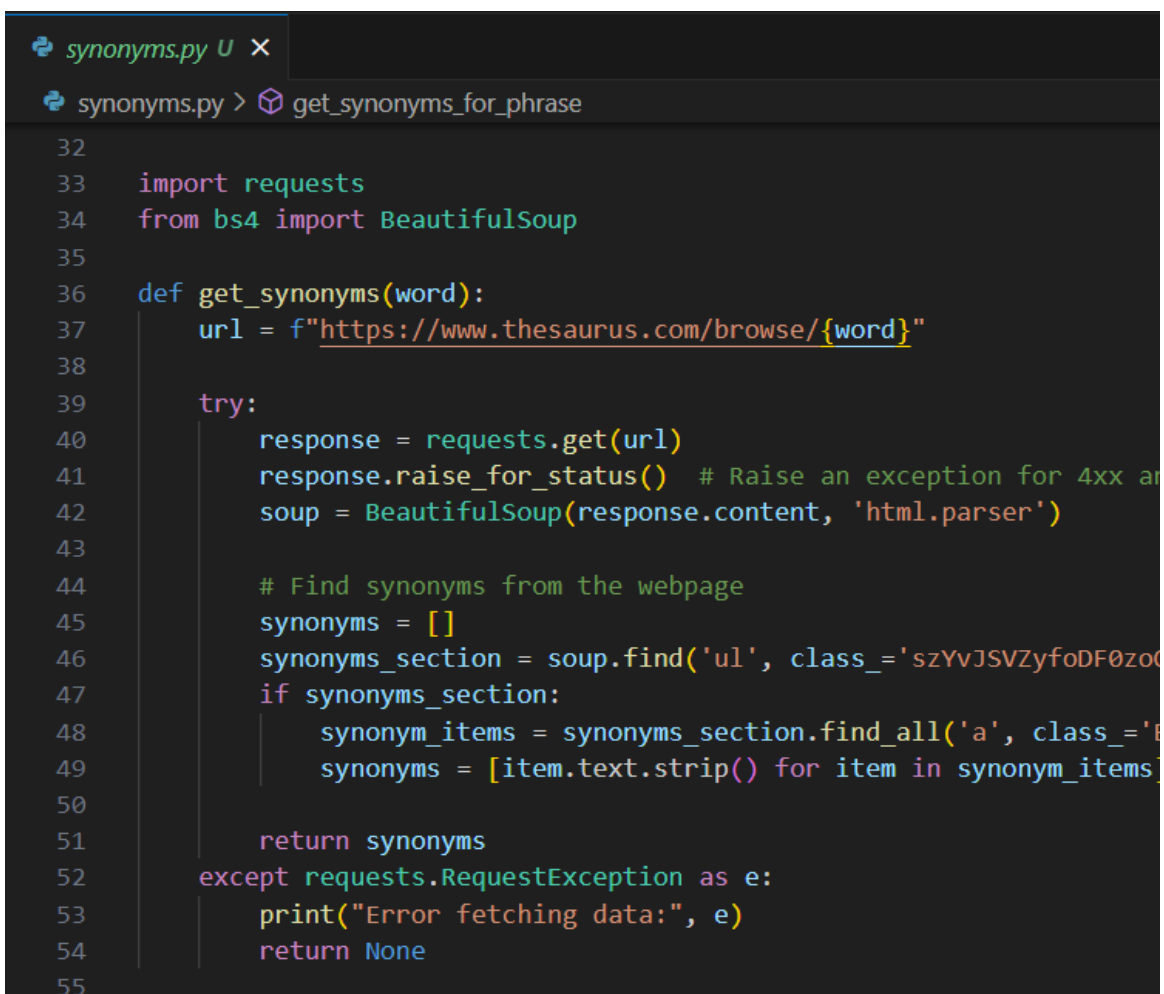
❖ Week 3-4: Image Augmentation Implementation

- Implement image augmentation techniques such as rotation, flipping, and scaling to diversify dataset.
- Integrate augmentation functionality into dataset preparation pipeline.
- Test augmented dataset on emotion detection model and assess performance improvements.

5.2 Module Description

5.2.1 NLP Based Query Expansion Module -

- ❖ **Preprocessing:** lemmatizing words to their base form for standardization.
- ❖ **Spell Correction:** Identify and correct spelling errors in the processed text using the NLTK corpus of English words.
- ❖ **Synonym Expansion:** Utilize Word Sense Disambiguation (WSD) to find synonyms for each word.
- ❖ **Query Expansion:** Generate an expanded query by incorporating corrected spellings and synonyms, removing redundant keywords
- ❖ **Feedback and Logging:** Provide informative messages throughout the expansion process to track the transformation of the base query and the expansion steps.



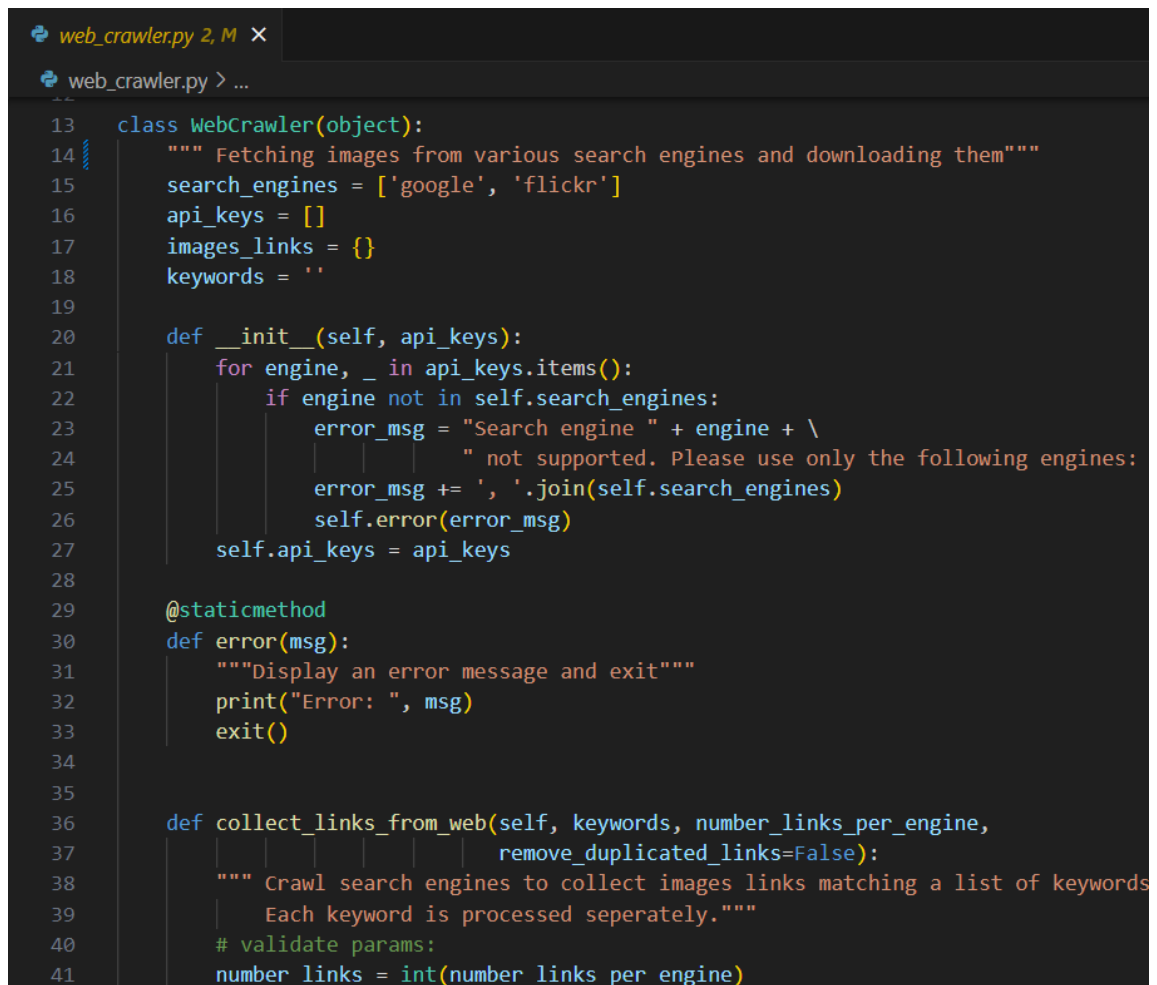
```
synonyms.py U X
synonyms.py > get_synonyms_for_phrase

32
33 import requests
34 from bs4 import BeautifulSoup
35
36 def get_synonyms(word):
37     url = f"https://www.thesaurus.com/browse/{word}"
38
39     try:
40         response = requests.get(url)
41         response.raise_for_status() # Raise an exception for 4xx and 5xx status codes
42         soup = BeautifulSoup(response.content, 'html.parser')
43
44         # Find synonyms from the webpage
45         synonyms = []
46         synonyms_section = soup.find('ul', class_='szYvJSVZyfoDF0zoc')
47         if synonyms_section:
48             synonym_items = synonyms_section.find_all('a', class_='t')
49             synonyms = [item.text.strip() for item in synonym_items]
50
51         return synonyms
52     except requests.RequestException as e:
53         print("Error fetching data:", e)
54         return None
55
```

Figure 10 : Code Snippet for NLP Query Expansion

5.2.2 Web Scrapping Module -

- ❖ **Initialization:** Set up necessary configurations including WebDriver path, image storage directory, search key, and optional parameters such as number of images, headless mode, and resolution constraints.
- ❖ **Gathering Image URLs:** Utilize Selenium to navigate to Google Images, extract image links, and handle pagination for gathering specified number of image URLs.
- ❖ **Downloading Images:** Iterate through collected image URLs, download images, and save them to the designated directory while optionally filtering by resolution and renaming files.
- ❖ **Exception Handling:** Handle exceptions gracefully during URL extraction and image downloading processes to ensure robustness.
- ❖ **Driver Management:** Effectively manage WebDriver instances, including handling browser window size and closing WebDriver after scraping is completed.
- ❖ **Optional Features:** Support for features like headless browsing, resolution filtering, and keeping original filenames as per user preferences.
- ❖ **Feedback and Logging:** Provide informative messages throughout the scraping process to inform the user of progress, errors, and completed tasks.



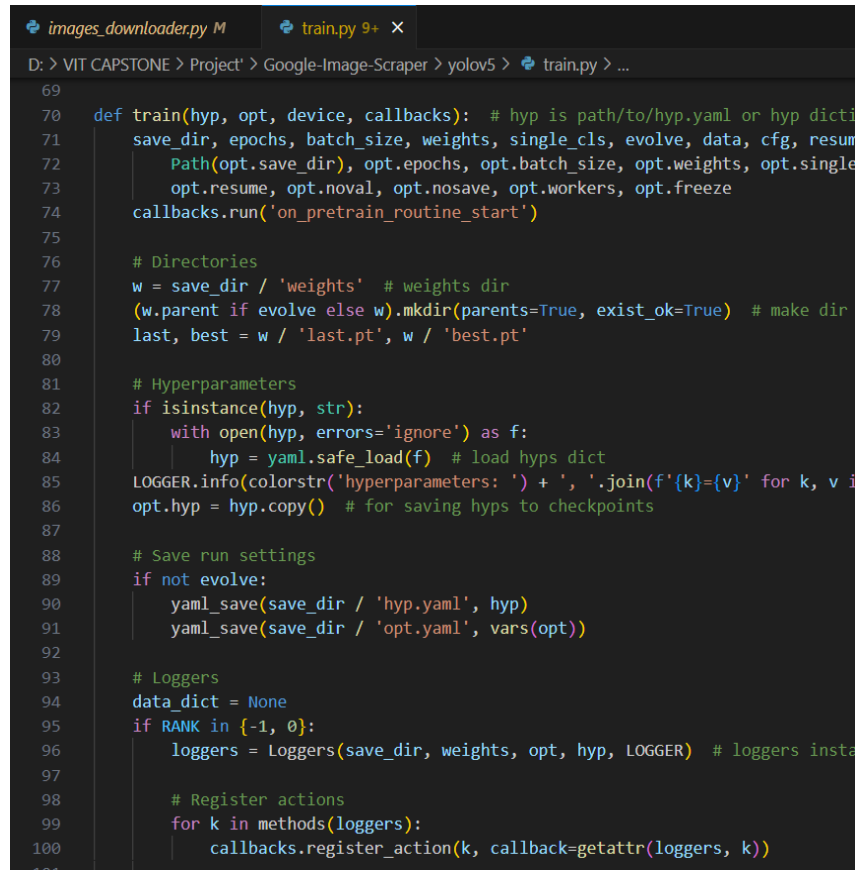
```
web_crawler.py 2, M x
web_crawler.py > ...

13 class WebCrawler(object):
14     """ Fetching images from various search engines and downloading them"""
15     search_engines = ['google', 'flickr']
16     api_keys = []
17     images_links = {}
18     keywords = ''
19
20     def __init__(self, api_keys):
21         for engine, _ in api_keys.items():
22             if engine not in self.search_engines:
23                 error_msg = "Search engine " + engine + \
24                     " not supported. Please use only the following engines:
25                     error_msg += ', '.join(self.search_engines)
26                 self.error(error_msg)
27             self.api_keys = api_keys
28
29     @staticmethod
30     def error(msg):
31         """Display an error message and exit"""
32         print("Error: ", msg)
33         exit()
34
35
36     def collect_links_from_web(self, keywords, number_links_per_engine,
37                               remove_duplicated_links=False):
38         """ Crawl search engines to collect images links matching a list of keywords
39             Each keyword is processed seperately."""
40         # validate params:
41         number_links = int(number_links_per_engine)
```

Figure 11 : Code Snippet for Web Scrapping Algorithm

5.2.3 Image Tagging Module (YOLOv5 Object Detection Model) -

- ❖ **Image Retrieval:** Retrieve images using the scraping module by scraping the images from the search engine.
- ❖ **Feature Extraction:** Extract visual features from images using computer vision algorithms, capturing characteristics like color, texture, and shape.
- ❖ **Tag Assignment:** Assign descriptive tags to images based on extracted features and semantic analysis, ensuring accuracy and relevance
- ❖ **Acceptance Criteria:** Accept images with clear relevance to the target concept, high visual quality, and appropriate content for the intended application.
- ❖ **Rejection Criteria:** Reject images with low resolution, irrelevant content, or inappropriate subject matter, maintaining data integrity

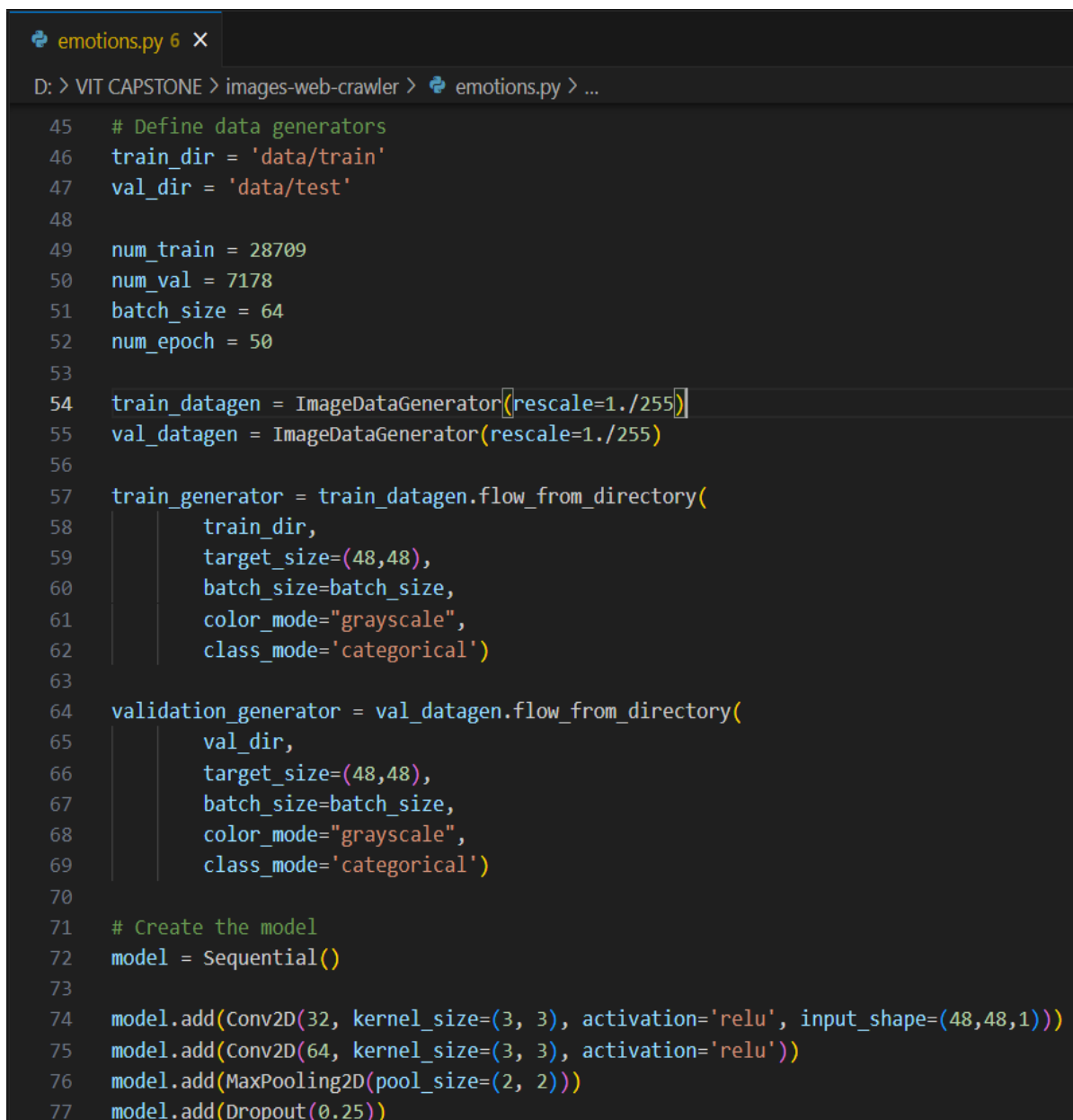


```
69
70 def train(hyp, opt, device, callbacks): # hyp is path/to/hyp.yaml or hyp dicti
71     save_dir, epochs, batch_size, weights, single_cls, evolve, data, cfg, resum
72     Path(opt.save_dir), opt.epochs, opt.batch_size, opt.weights, opt.single
73     opt.resume, opt.noval, opt.nosave, opt.workers, opt.freeze
74     callbacks.run('on_pretrainRoutine_start')
75
76     # Directories
77     w = save_dir / 'weights' # weights dir
78     (w.parent if evolve else w).mkdir(parents=True, exist_ok=True) # make dir
79     last, best = w / 'last.pt', w / 'best.pt'
80
81     # Hyperparameters
82     if isinstance(hyp, str):
83         with open(hyp, errors='ignore') as f:
84             hyp = yaml.safe_load(f) # load hyps dict
85     LOGGER.info(colorstr('hyperparameters: ') + ', '.join(f'{k}={v}' for k, v i
86     opt.hyp = hyp.copy() # for saving hyps to checkpoints
87
88     # Save run settings
89     if not evolve:
90         yaml_save(save_dir / 'hyp.yaml', hyp)
91         yaml_save(save_dir / 'opt.yaml', vars(opt))
92
93     # Loggers
94     data_dict = None
95     if RANK in {-1, 0}:
96         loggers = Loggers(save_dir, weights, opt, hyp, LOGGER) # loggers insta
97
98     # Register actions
99     for k in methods(loggers):
100         callbacks.register_action(k, callback=getattr(loggers, k))
101
```

Figure 12 : Code Snippet for Model Training for Image tagging

5.2.3 Image Tagging Module (Emotion Detection) –

- ❖ **Setup and Configuration:** Essential libraries are imported, and command-line arguments are parsed to determine the mode: training or real-time emotion detection.
- ❖ **Model Definition and Training:** A CNN model is constructed using TensorFlow's Kera's API. If the mode is set to "train", the model is compiled, trained on provided data, and its performance is visualized.
- ❖ **Data Preparation:** Directories for training and validation data are specified, and data generators are created for image preprocessing and augmentation.
- ❖ **Real-time Emotion Detection:** If the mode is set to "display", the trained model is loaded. Webcam feed is captured, faces are detected using Haar cascades, and emotions are predicted and annotated in real-time.
- ❖ **User Interaction and Feedback:** The processed video feed with emotion annotations is displayed, and the program terminates upon user input.

A screenshot of a code editor window titled 'emotions.py 6 X'. The editor shows a Python script for setting up an emotion detection model. The code includes comments and assignments for training and validation directories, data generator parameters, and the model architecture. The file path in the top bar is 'D: > VIT CAPSTONE > images-web-crawler > emotions.py > ...'.

```
45 # Define data generators
46 train_dir = 'data/train'
47 val_dir = 'data/test'
48
49 num_train = 28709
50 num_val = 7178
51 batch_size = 64
52 num_epoch = 50
53
54 train_datagen = ImageDataGenerator(rescale=1./255)
55 val_datagen = ImageDataGenerator(rescale=1./255)
56
57 train_generator = train_datagen.flow_from_directory(
58     train_dir,
59     target_size=(48,48),
60     batch_size=batch_size,
61     color_mode="grayscale",
62     class_mode='categorical')
63
64 validation_generator = val_datagen.flow_from_directory(
65     val_dir,
66     target_size=(48,48),
67     batch_size=batch_size,
68     color_mode="grayscale",
69     class_mode='categorical')
70
71 # Create the model
72 model = Sequential()
73
74 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
75 model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
76 model.add(MaxPooling2D(pool_size=(2, 2)))
77 model.add(Dropout(0.25))
```

Figure 13 : Code Snippet for Emotion Detection Model

5.2.4 Image Augmentation Module -

- ❖ **Setup and Configuration:** Define the directory containing the original images and create a directory to store the augmented images.
- ❖ **Augmentation Techniques:** Implement augmentation techniques such as rotation, Gaussian blur, and resizing using the imgaug library to diversify the dataset.
- ❖ **Image Processing:** Iterate through each image file, open it, resize it to the desired dimensions, and convert it to a numpy array for augmentation.
- ❖ **Augmentation Generation:** Generate multiple augmented images for each original image by applying augmentation transformations iteratively.
- ❖ **File Handling:** Save the augmented images with unique filenames and the desired file format in the specified directory for future use.
- ❖ **Feedback and Logging:** Provide feedback on the completion of the augmentation process, including the number of original images processed and the total number of augmented images generated.

```
postprocessing.py 1, U X
postprocessing.py > ...
9   dir_path = "data/dogs"
10
11   if not os.path.exists('augmented_images'):
12       os.makedirs('augmented_images')
13
14   # Define the new height and width of the image after scaling
15   #Take input from user
16   new_height = 450
17   new_width = 450
18
19   # Define the augmentations to be performed
20   seq = iaa.Sequential([
21       # iaa.Flipud(), # vertically flip the image
22       iaa.Affine(rotate=(-179, 179)), # rotate the image by a random
23       iaa.GaussianBlur(sigma=(0, 1.0)), # apply gaussian blur with a
24       iaa.Resize({"height": new_height, "width": new_width}) # resize
25   ])
26
27   # Define the number of augmented images to generate
28   num_aug_images = 10
29
30   # Get the list of files in the directory
31   file_list = os.listdir(dir_path)
32
33   # Count the number of image files in the directory
34   num_images = sum([1 for file in file_list if file.endswith(".jpg") or
35   file.endswith(".jpeg") or file.endswith(".png")])
36
37   # Loop over each image file in the directory and generate the augmented
38   for file in file_list:
39       # Check if the file is an image file
40       if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith(".png"):
41           # Set the path to the image file
42           image_file = os.path.join(dir_path, file)
```

Figure 14 : Code Snippet for Post Processing

5.3 Testing

5.3.1 Model Testing –

```
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [11/Apr/2024 00:59:53] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [11/Apr/2024 00:59:54] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [11/Apr/2024 00:59:54] "GET /favicon.ico HTTP/1.1" 404 -
{'happy': ['cheerful', 'delighted', 'ecstatic', 'elated', 'enraptured', 'exultant', 'glad', 'gleeful', 'jolly', 'joyful', 'joyous', 'jubilant', 'merry', 'mir-
thful', 'overjoyed', 'thrilled', 'up', 'upbeat'], 'human': [], 'face': []}
Start fetching...

Fetching for ' happy human face ' images...
Crawling Google Search Engine...
>> 10 links extracted...
Crawling Flickr Search...
>> 10 links extracted...
>> 0 duplicated links removed, 20 kept

Links saved to ' ./data/links.txt '

Preparing to download images...
Downloading files...
>> Download progress: 100.0 %...
>> 19 images downloaded
>> Failed to download 1 images: access not granted (links saved to: ' ./data /failed_list.txt')
Merging ' ./data ' files...
Merging ' ./data/birds ' files...
Merging ' ./data/cats ' files...
Merging ' ./data/child eating burger ' files...
Merging ' ./data/child eating food ' files...
Merging ' ./data/child eating ice cream ' files...
Merging ' ./data/dogs ' files...
Merging ' ./data/football ' files...
Merging ' ./data/happy human face ' files...
Merging ' ./data/kids ' files...
Merging ' ./data/person eating pizza ' files...
Merging ' ./data/testing ' files...
127.0.0.1 - - [11/Apr/2024 01:03:45] "POST / HTTP/1.1" 200 -
```

Figure 15 : Backend Running Code for Query Expansion and Query Generation (Step 1)

```
127.0.0.1 - - [30/Apr/2024 21:34:34] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Apr/2024 21:34:34] "GET /static/style.css HTTP/1.1" 200 -
{'sad': ['depressed', 'heartbroken', 'melancholy', 'mournful', 'pessimistic', 'somber', 'sorrowful', 'sorry', 'unhappy'],
Start fetching...

Fetching for ' sad human face ' images...
Crawling Google Search Engine...
>> 50 links extracted...
Crawling Flickr Search...
>> 50 links extracted...
>> 0 duplicated links removed, 100 kept

Links saved to ' ./data/links.txt '

Preparing to download images...
Downloading files...
>> Download progress: 35.0 %...
```

Figure 16 : Image Scraping

```
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [11/Apr/2024 01:09:10] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [11/Apr/2024 01:09:10] "GET /static/style.css HTTP/1.1" 200 -
127.0.0.1 - - [11/Apr/2024 01:09:10] "GET /favicon.ico HTTP/1.1" 404 -
Current Directory: C:\Users\Priyansh\Desktop\NewTry2\Google-Image-Scraper
Updated Directory: C:\Users\Priyansh\Desktop\NewTry2\Google-Image-Scraper\yolov5
Current number of result directories: 1
Current number of inference detection directories: 27
inference_28
detect: weights=[runs/train/results_1/weights/best.pt], source=./inference_images, data=data/coco128.yaml, imgsz=[640, 640], conf_thres=0.25, iou_thres=0.45, max_det=1000, device=, view_img=False, save_txt=False, save_conf=False, save_crop=False, nosave=False, classes=None, agnostic_nms=False, augment=False, visualize=False, update=False, project=runs/detect, name=inference_28, exist_ok=False, line_thickness=3, hide_labels=False, hide_conf=False, half=False, dnn=False, vid_stride=1
YOLOv5 v7.0-175-g5f11555 Python-3.10.7 torch-2.0.1+cpu CPU
```

Figure 17 : Running of Image Tagging process in backend

5.3.2 Post Processing Testing

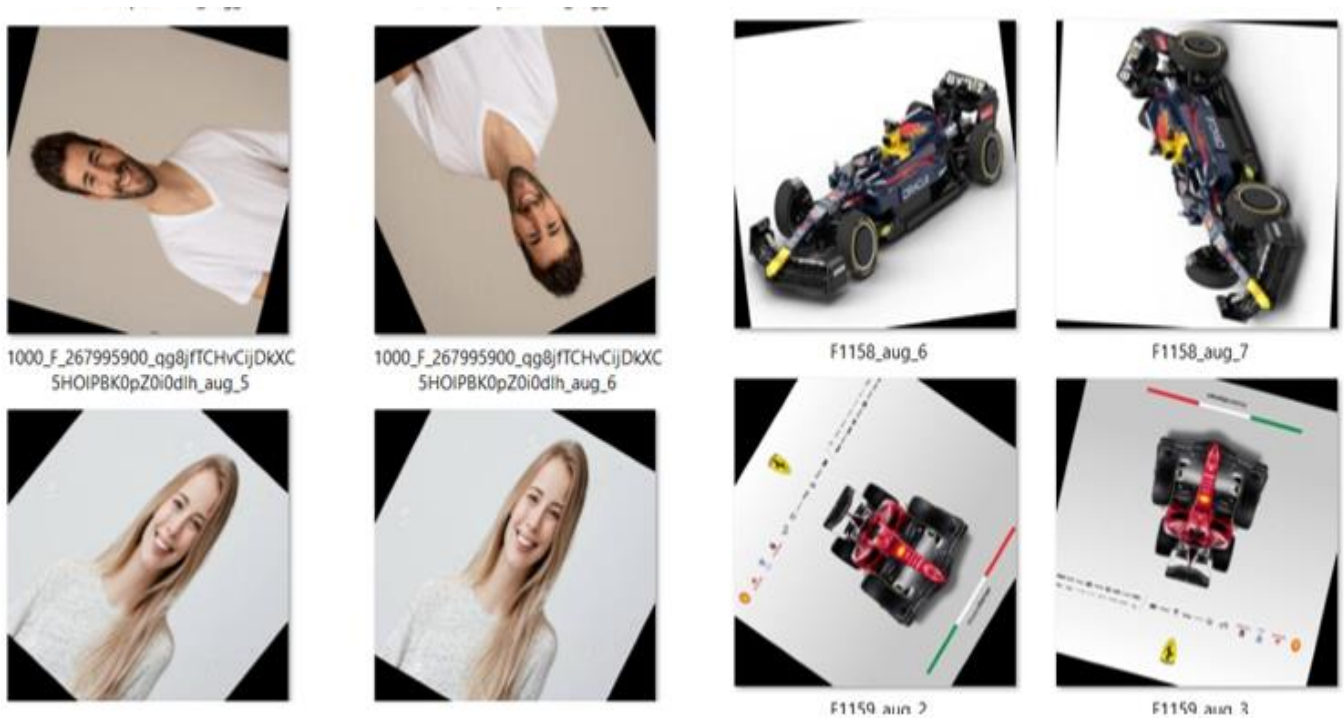


Figure 18 : Dataset generated after post-processing

6. PROJECT DEMONSTRATION

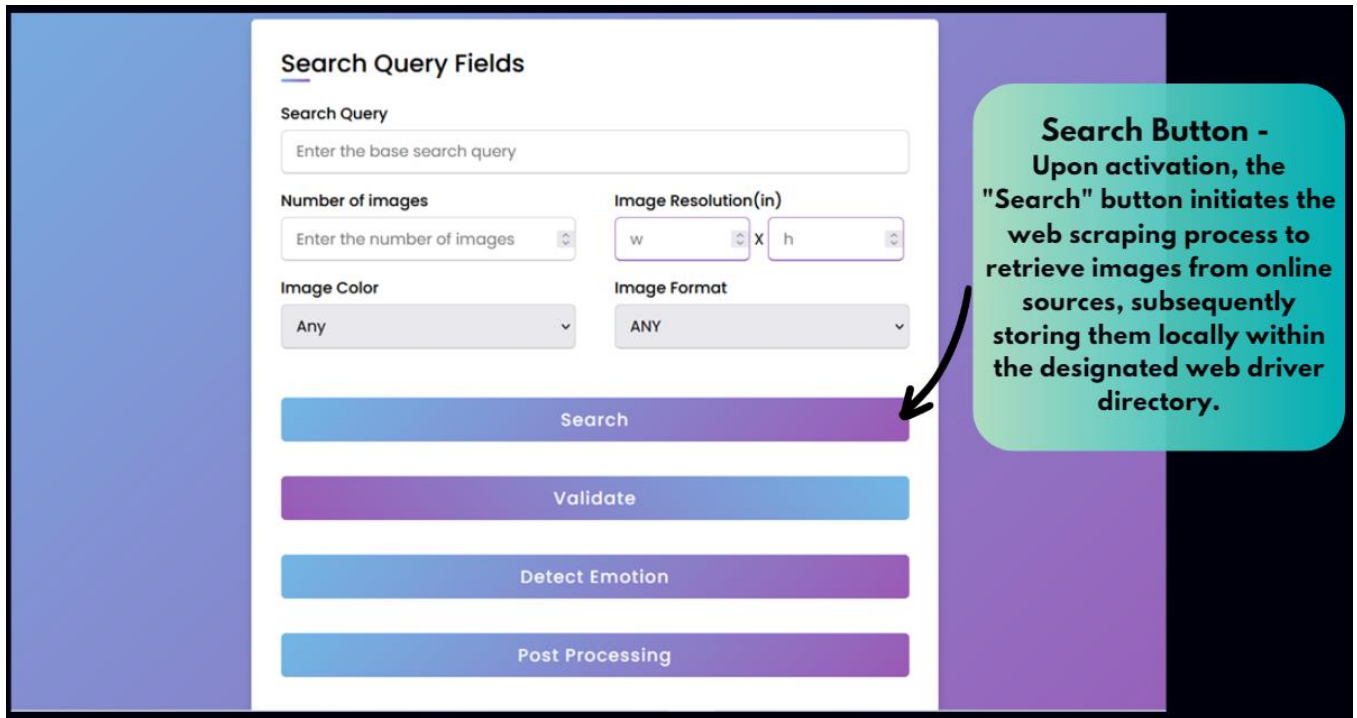


Figure 19 : Frontend Website for Automatic Dataset Generation

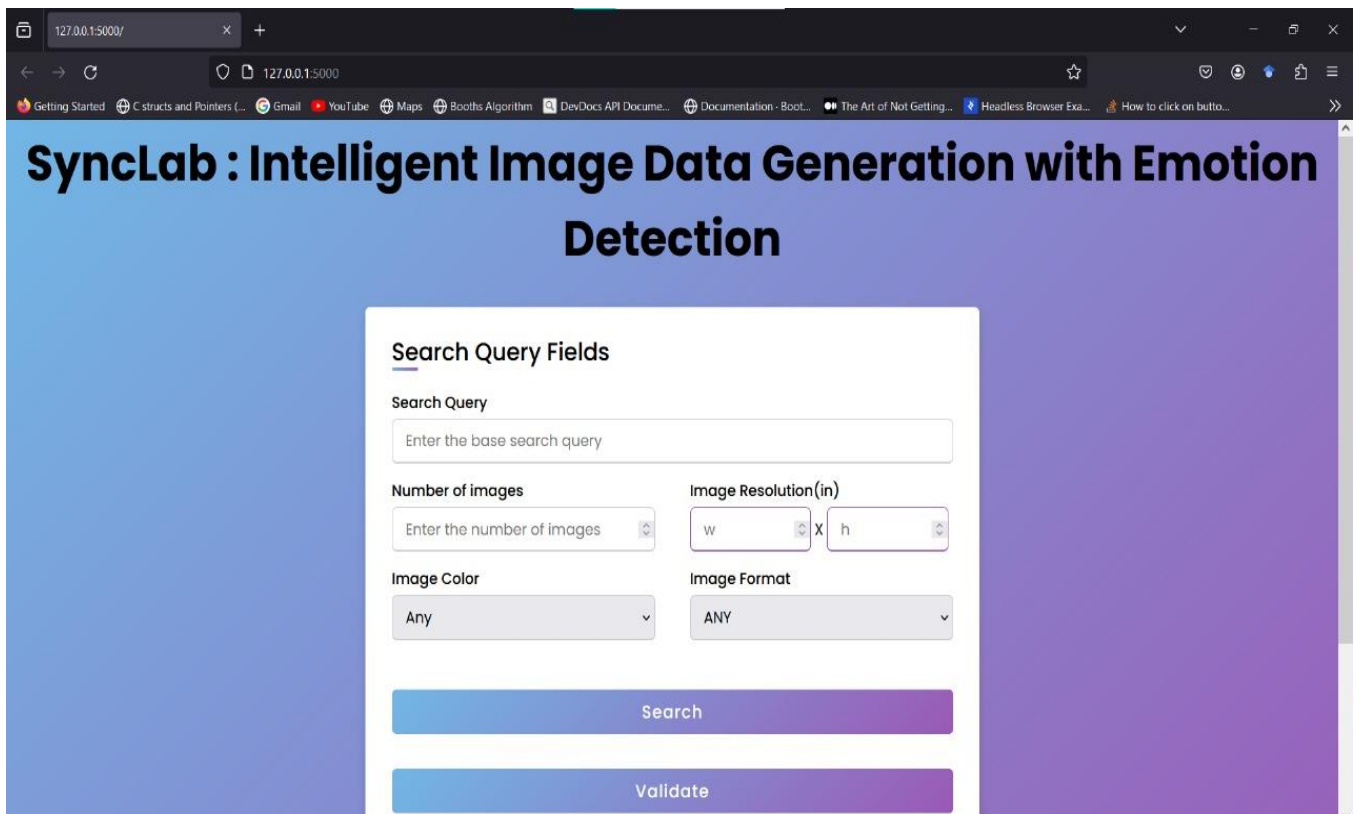


Figure 20 : A Screenshot of the Webpage

SyncLab: Intelligent Image Data Generation System

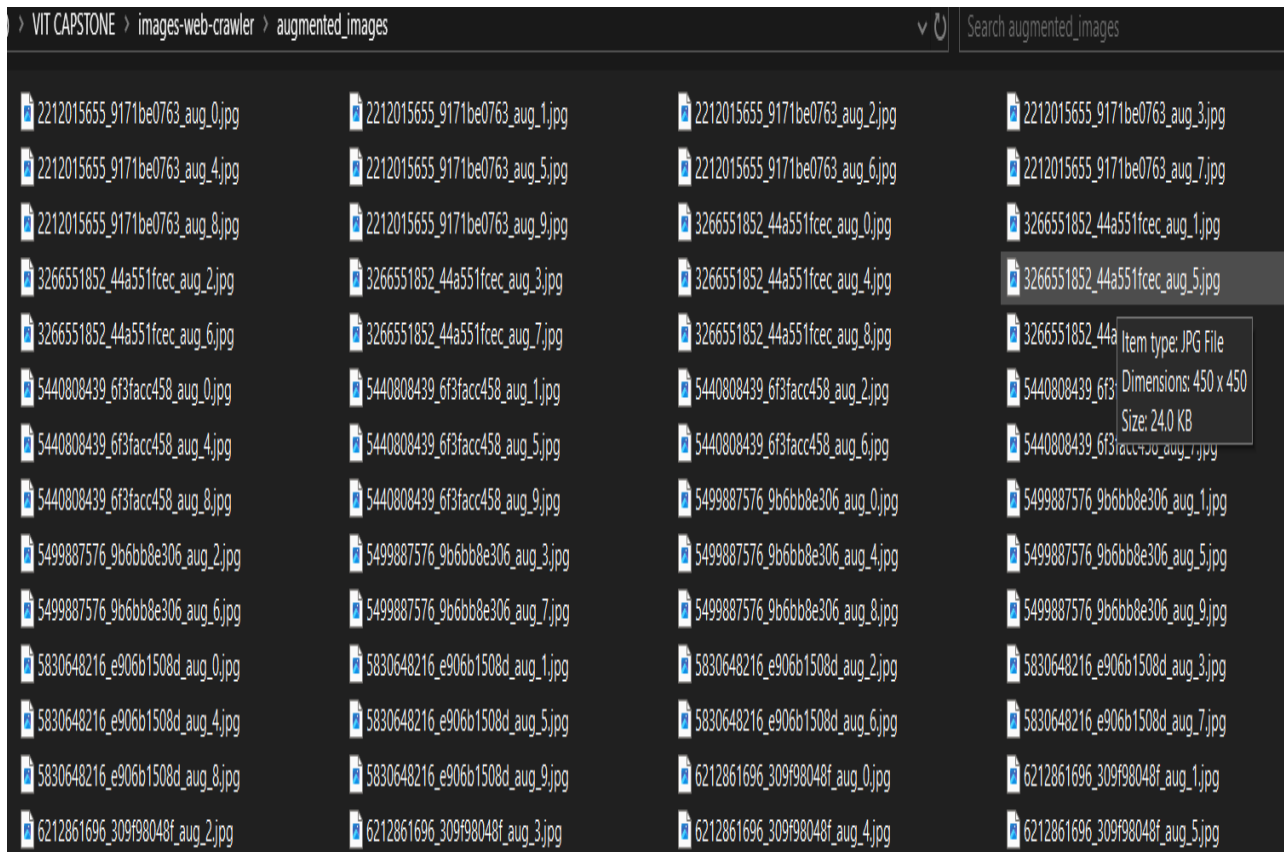


Figure 21 : Final generated dataset stored in a Folder

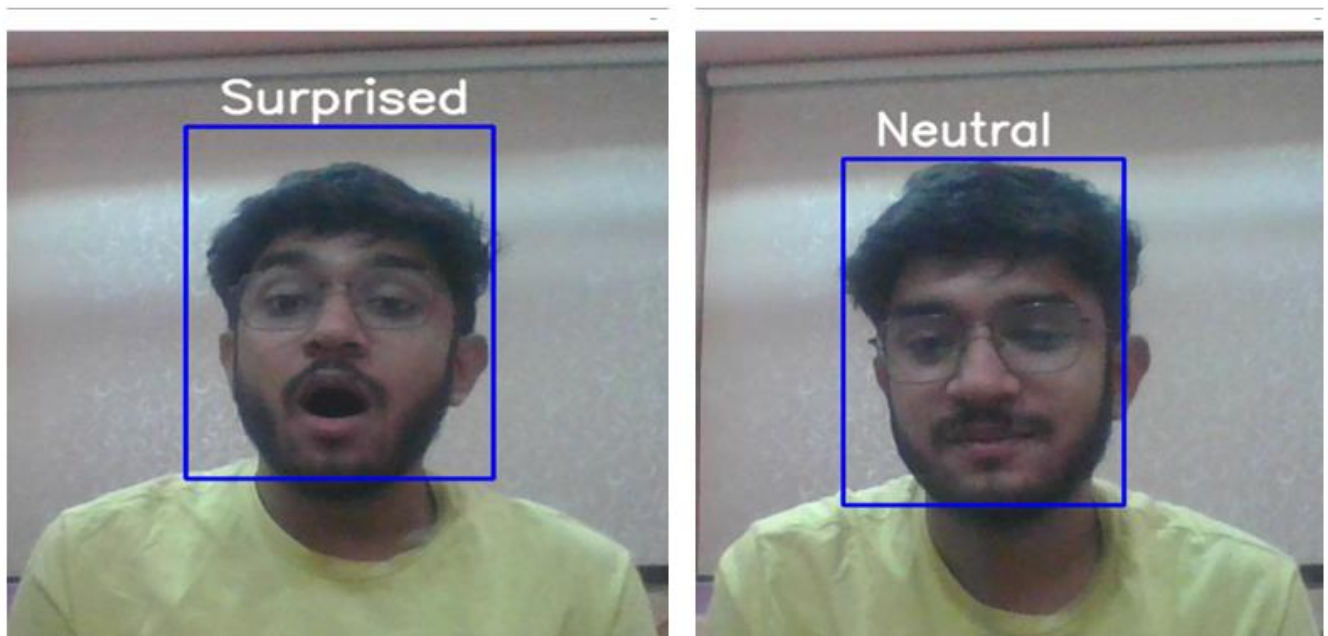


Figure 22 : Emotion Detection Model Testing (Image Tagging)



Figure 23 : YOLOv5 Object Detection Model Testing (Image Tagging)

7. RESULTS AND DISCUSSIONS

The culmination of our project marks a significant milestone in the realm of vision AI solutions, addressing the longstanding challenge of efficiently and effectively generating high-quality image datasets tailored to specific use cases. Through the integration of innovative methodologies and advanced technologies, our solution has yielded a comprehensive and diverse dataset that serves as a foundational resource for various deep learning and AI endeavors.

❖ Superior Dataset Quality:

- Our dataset exhibits object-centricity and richness in data representation, meeting specific user-defined requirements through meticulous curation and advanced web scraping techniques.
- The dataset captures a wide spectrum of objects, actions, and angles, reflecting real-world scenarios and facilitating more accurate and robust AI model training.

❖ Performance Benchmarking:

- Comparative analyses against existing online datasets demonstrate superior performance in terms of accuracy percentages across multiple evaluation metrics.
- Rigorous testing for tasks such as emotion detection and object detection using YOLOv5 consistently showcases our solution's outperformance of competitors, resulting in higher accuracy rates.

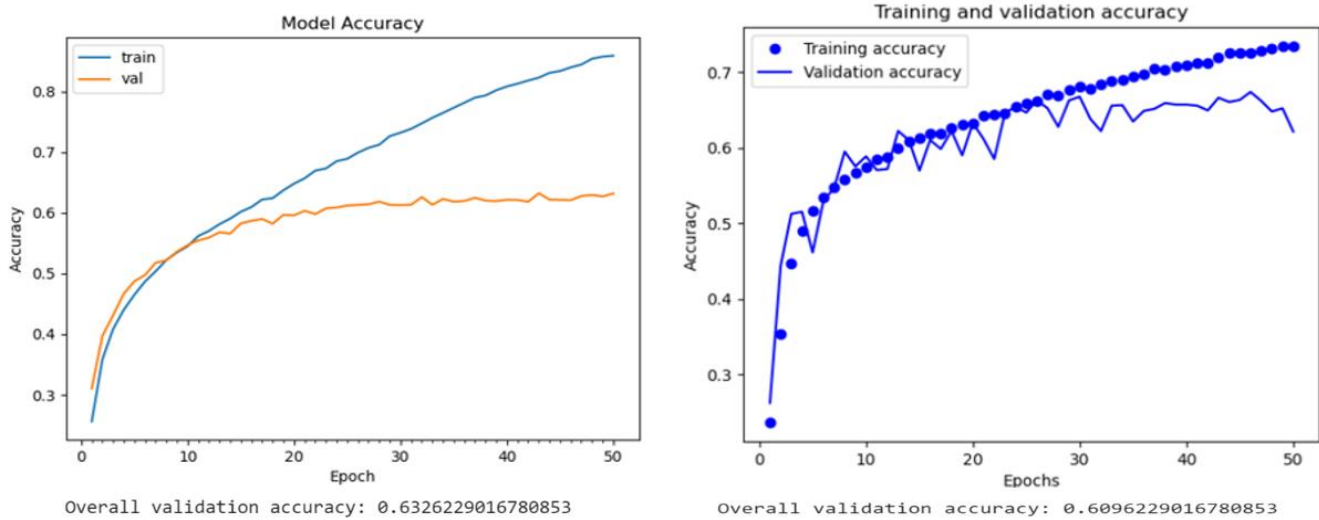


Figure 24 : Comparing model accuracy between our proprietary dataset and publicly available datasets.

❖ Validation of Approach:

- Our project has contributed valuable insights into the limitations of existing models and datasets, highlighting areas where our solution excels and its potential to address gaps in the current landscape of vision AI solutions.

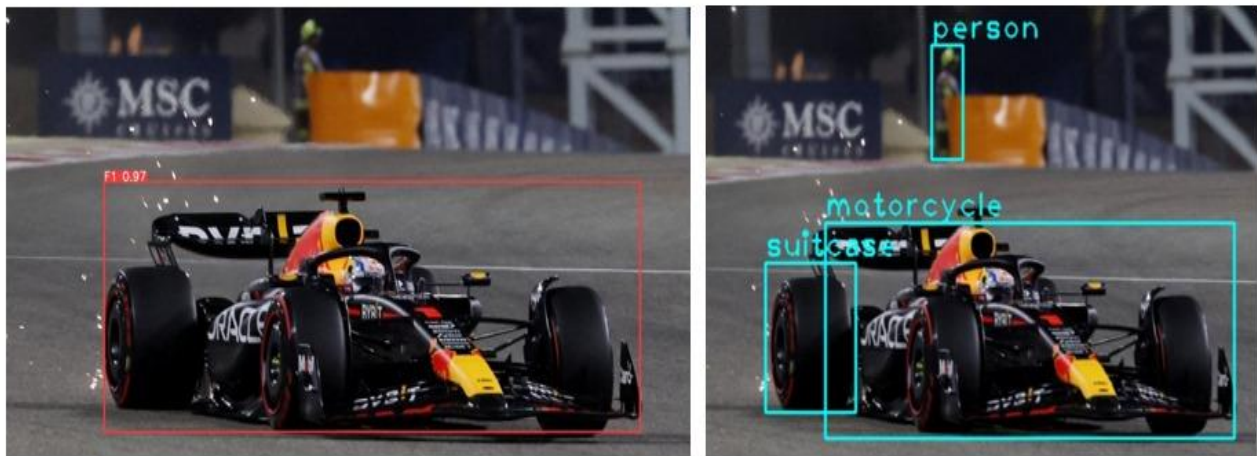


Figure 25 : Comparing model accuracy between our proprietary & publicly available datasets(YOLOv5 Model)

Base Query: F1 cars racing

Number of Images: 500

Results:

Total number of output images: 500

Confidence score threshold: 0.90 and above for acceptance

Manually annotated images: 70 images

Average accuracy achieved - **94%**



Figure 26 : Results for YoloV5 based Dataset for F1 Cars

In conclusion, the results of our project underscore the transformative potential of leveraging innovative approaches and cutting-edge technologies in the creation of image datasets for vision AI solutions. By delivering a comprehensive, diverse, and high-quality dataset, we have laid the groundwork for future advancements in AI model training and deployment, ultimately driving progress towards more intelligent and effective AI systems in various domains.

8. CONCLUSION AND FUTURE WORKS

In conclusion, our project represents a significant advancement in the field of artificial intelligence, particularly in the realm of computer vision and emotion detection systems. By leveraging natural language processing (NLP) models to expand base search queries and automating the process of image dataset creation through web scraping, we have addressed the critical need for diverse and high-quality datasets essential for training and evaluating AI models. Our approach not only streamlines the labor-intensive task of dataset generation but also ensures adherence to specific requirements, thereby enhancing the accuracy and effectiveness of vision-based AI solutions. Through the implementation of image augmentation techniques such as rotation, flipping, and scaling, we have further enriched the dataset, enhancing model robustness and generalization capabilities. This diversification of the dataset not only improves model performance but also mitigates the risk of overfitting, ensuring reliable and consistent results across various real-world scenarios.

The evaluation of our dataset using a sophisticated emotion detection model founded on complex computer vision and AI techniques has yielded promising results, underscoring the efficacy of our approach. The methodologies and techniques developed herein can be extended and applied to a myriad of other vision-based AI solutions, ranging from object recognition to facial recognition systems, opening new avenues for innovation and discovery.

In essence, our project not only addresses the pressing need for high-quality datasets in AI research but also contributes to the advancement of vision-based AI solutions with real-world applicability and impact. As we continue to push the boundaries of AI technology, we remain committed to driving innovation and fostering progress in this dynamic and ever-evolving field.

9. REFERENCES

- [1] Niu, Qingli, et al. "Web Scraping Tool for Newspapers and Images Data Using Jsonify." *Journal of Applied Science and Engineering* 26.4 (2022): 465-474
- [2] Zhao, Bo. (2017). *Web Scraping*. 10.1007/978-3-319-32001-4_483-1.
- [3] Ilham, et al. "Image search optimization with web scraping, text processing and cosine similarity algorithms." 2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat). IEEE, 2020.
- [4] Daniel Glez-Peña, et al. "Web scraping technologies in an API world", *Briefings in Bioinformatics*, Volume 15, Issue 5, September 2014, Pages 788–797, <https://doi.org/10.1093/bib/bbt026>
- [5] Esposito, Massimo, et al. "Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering." *Information Sciences* 514 (2020): 88-105
- [6] Rahimi, et al. "Improving Information Retrieval Results for Persian Documents using FarsNet." *arXiv preprint arXiv:1811.00854* (2018).
- [7] Bale, Ajay Sudhir, et al. "Web scraping approaches and their performance on modern websites." 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, 2022.
- [8] Singrodia, et al. "A review on web scrapping and its applications." 2019 international conference on computer communication and informatics (ICCCI). IEEE, 2019.
- [9] NR, Ruchitaa Raj, et al. "Web scrapping tools and techniques: A brief survey." 2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT). IEEE, 2023.
- [10] Dallmeier, et al. "Computer vision-based web scraping for internet forums." 2021 7th International Conference on Optimization and Applications (ICOA). IEEE, 2021.
- [11] Kuzi, Saar, et al. "Query expansion using word embeddings." *Proceedings of the 25th ACM international conference on information and knowledge management*. 2016.
- [12] Silva, Alfredo, et al. "Improving query expansion strategies with word embeddings." *Proceedings of the ACM Symposium on Document Engineering* 2020.
- [13] Esposito, et al. "Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering." *Information Sciences* 514 (2020): 88-105.
- [14] Naseri, Shahrzad, et al. "Ceqe: Contextualized embeddings for query expansion." *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part I* 43. Springer International Publishing, 2021.

- [15] Hidayatin, et al. "Query expansion evaluation for chatbot application." 2018 International Conference on Applied Information Technology and Innovation (ICAITI). IEEE, 2018.
- [16] Garcia-Garcia, et al. "Emotion detection: a technology review." Proceedings of the XVIII international conference on human computer interaction. 2017.
- [17] Fu, Jianlong, et al. "Advances in deep learning approaches for image tagging." APSIPA Transactions on Signal and Information Processing 6 (2017): e11.
- [18] Zhang, Youcai, et al. "Recognize anything: A strong image tagging model." arXiv preprint arXiv:2306.03514 (2023).