AI Image Webscraper and Preprocessing Pipeline

Table of Contents

1. Introduction

Project Overview

The project involved building an automated web scraping and preprocessing system to gather and filter images from multiple search engines. The goal was to streamline the collection of relevant images while reducing the need for manual intervention, achieving scalability and efficiency in handling large datasets.

Scope

The project required integrating web scraping scripts with computer vision (CV) models for preprocessing, deploying the pipeline on AWS infrastructure, and setting up notifications for monitoring. It encompassed technologies such as Python, ChromeDriver, YOLOv8 for image processing, and AWS services like EC2, S3, SNS, and Lambda.

2. Description of Tasking

The primary task was to scrape and preprocess images from five major search engines:

- Bing
- Google
- Yahoo
- Yandex
- DuckDuckGo

The workflow included:

- Scraping images using a Python script with ChromeDriver for automation.
- Preprocessing the images with a CV model to filter out unwanted data, such as collages or photos containing humans.
- Uploading the cleaned images to an S3 bucket.
- Sending Telegram alerts for manual review via AWS SNS and Lambda integration.

The system processed over 100,000 images during the project, with weekly batches averaging 15,000–20,000 photos.

3. Difficulties of Tasking

Structural Problems

- Search Engine Limitations: Handling rate limits and CAPTCHA mechanisms on multiple search engines.
- Variability in Image Content: A significant variety in image quality and relevance increased preprocessing complexity.

Resource Constraints

- Infrastructure Scaling: Balancing EC2 instance usage with storage and processing demands on S3.
- Automation Limits: Even with automated filtering, final validation still required manual checking.

4. System Implementation

The implementation was divided into the following steps:

Web Scraping

Developed Python scripts using Selenium with ChromeDriver to automate image downloads from each search engine.

Applied query-specific filters to retrieve diverse datasets.

Preprocessing with YOLOv8

Implemented a computer vision model to analyze and discard non-relevant images (e.g., collages, human photos).

Achieved a batch size reduction of 10–20% on average.

AWS Deployment

Deployed the system on an AWS EC2 instance for continuous operation. Uploaded filtered images to an S3 bucket for secure storage.

Alert System

Used AWS SNS and Lambda to send Telegram alerts for every batch of 30 images processed, notifying for manual review.

5. Challenges and Solutions

Search Engine CAPTCHAs

- Addressed using dynamic wait times and proxy rotation in the scraping scripts.

Model Accuracy

- Tuned YOLOv8 thresholds to minimize false positives while maintaining high precision in filtering.

Scalability

- Implemented parallel processing on the EC2 instance to handle larger batches efficiently.

6. Recommendations for Future Work

Improved Automation

- Train custom CV models to refine preprocessing, reducing dependency on manual review.
- Explore integrating multilingual scraping to expand dataset diversity.

Enhanced Efficiency

- Use distributed scraping with multiple EC2 instances to scale the operation further. Implement incremental scraping to avoid duplicate downloads. Data Labeling and Analytics

Develop a labeling pipeline for downstream AI model training purposes. Create dashboards for analyzing dataset statistics and filtering performance.

7. Summary

The project successfully developed a robust system for automated image scraping and preprocessing, achieving significant efficiency in handling large-scale datasets. Over 100,000 images were processed with the pipeline, demonstrating its capability to integrate web scraping, computer vision, and cloud infrastructure. Future iterations of the system could improve automation and scalability, paving the way for broader applications in AI and data analysis.

8. Acknowledgements

Special thanks to:

- Alfred and Jia Jun for guidance and feedback.
- My Team Members for their collaboration and support during implementation.
- The AWS and Python development communities for providing resources and tools to streamline the project.