Department of Electrical and Computer Engineering
North South University

Senior Design Project -CSE499A

Fire and Smoke Detection Using YoloV8

Authors

Abu Mukaddim Rahi (2022027042)
Abdullah Almoon (1922223642)
Amrina Afroz (171259042)
Abu Sufiun (1831274642)

Faculty Advisor:

Dr. Mohammad Ashrafuzzaman Khan
Associate Professor

Project Design :

The design of our real-time fire and smoke detection system encompasses a holistic approach, involving hardware specifications, software architecture, and the integration of critical components. This section delineates the key elements of the system, outlining their functionalities and interactions.

Hardware Specifications

GPU: RTX 3050

The system leverages the computational power of an RTX 3050 GPU. This GPU is selected for its robust processing capabilities, specifically tailored to handle the demands of real-time object detection tasks. The parallel processing architecture of the RTX 3050 is instrumental in accelerating the inference speed of the YOLOv8 model.

Processor and Memory

Complementing the GPU, a high-performance processor and ample memory are integral components of the hardware setup. This ensures seamless coordination between the GPU, memory, and processor, optimizing the overall system performance.

Software Architecture

Operating System: Linux Ubuntu

The system operates on a Linux Ubuntu environment, chosen for its stability, compatibility with deep learning frameworks, and robust support for GPU acceleration. This operating system provides an ideal foundation for deploying and executing the YOLOv8 model.

Deep Learning Framework: PyTorch

PyTorch, a powerful and versatile deep learning framework, is employed for the implementation of the YOLOv8 model. Its dynamic computation graph and extensive library of pre-built modules facilitate seamless model development, training, and integration.

Programming Language: Python

Python serves as the primary programming language for the system. Its rich ecosystem of libraries, particularly those dedicated to machine learning and computer vision, enhances the efficiency and flexibility of system development.

System Components and Interactions

YOLOv8 Model Integration

The core of the system lies in the integration of the YOLOv8 model. This pre-trained model, fine-tuned on our annotated dataset, is seamlessly incorporated into the real-time detection pipeline. Its ability to process images in a single pass is pivotal in achieving real-time performance.

Video Input

The system receives a continuous stream of video frames from the selected video source. This input stream serves as the primary data source for the YOLOv8 model, enabling it to perform real-time object detection.

Frame Processing Module

The frame processing module is responsible for preprocessing each incoming frame before it is fed into the YOLOv8 model. This module applies necessary transformations, including resizing, normalization, and format conversion, to ensure compatibility with the model's input requirements.

Object Detection and Bounding Box Generation

Upon receiving a preprocessed frame, the YOLOv8 model performs object detection, predicting bounding boxes for instances of fire and smoke. These bounding box coordinates, along with associated class probabilities, are generated by the model and passed to the subsequent stages of the pipeline.

Post-Processing and Visualization

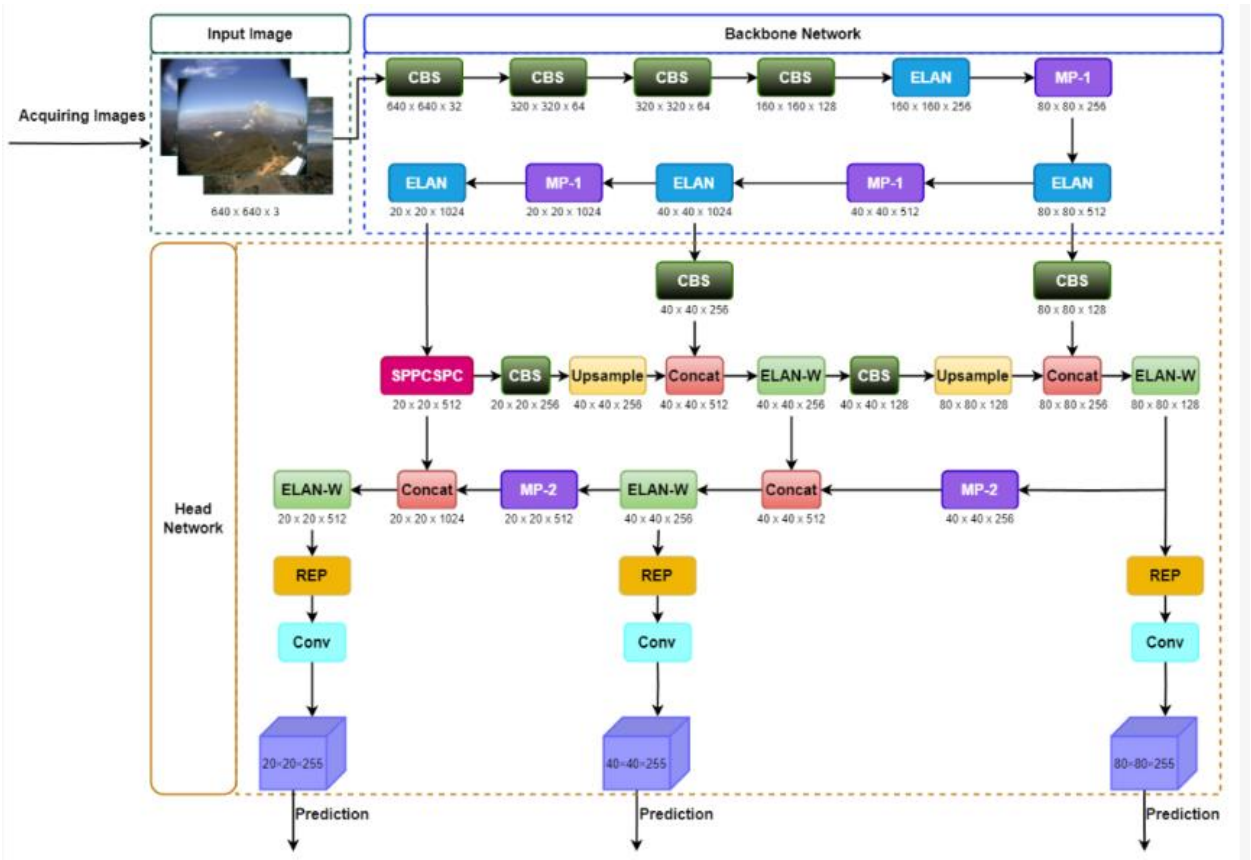
The post-processing module refines the raw detections generated by the YOLOv8 model. Non-maximum suppression is applied to eliminate redundant bounding boxes and retain only the most confident predictions. The final set of bounding boxes, along with corresponding class labels and confidence scores, are then visualized on the output frame.

Real-Time Display

The processed frames with annotated bounding boxes are presented in real-time on a graphical user interface (GUI). This GUI provides a visual representation of the detection system's performance, allowing for immediate feedback and monitoring.

System Workflow

The system workflow is orchestrated as a sequential pipeline. Video frames are continuously fed into the frame processing module, which prepares them for input into the YOLOv8 model. The model, in turn, performs object detection and generates bounding box predictions. These predictions undergo post-processing to refine the results, which are then visualized in real-time on the GUI.


Input Image
Backbone Network
Acquiring Images
640 x 640 x 3
CBS
640 x 640 x 32
CBS
320 x 320 x 64
CBS
320 x 320 x 64
CBS
160 x 160 x 128
ELAN
160 x 160 x 256
MP-1
80 x 80 x 256
ELAN
80 x 80 x 512
MP-1
40 x 40 x 512
ELAN
40 x 40 x 1024
MP-1
20 x 20 x 1024
ELAN
20 x 20 x 1024
CBS
40 x 40 x 256
CBS
80 x 80 x 128
Head Network
SPPCSPC
20 x 20 x 512
CBS
20 x 20 x 256
Upsample
40 x 40 x 256
Concat
40 x 40 x 512
ELAN-W
40 x 40 x 256
CBS
40 x 40 x 128
Upsample
80 x 80 x 128
Concat
80 x 80 x 256
ELAN-W
80 x 80 x 128
ELAN-W
20 x 20 x 512
Concat
20 x 20 x 1024
MP-2
20 x 20 x 512
ELAN-W
40 x 40 x 256
Concat
40 x 40 x 512
MP-2
40 x 40 x 256
REP
Conv
20×20×255
REP
Conv
40×40×255
REP
Conv
80×80×255
Prediction
Prediction
Prediction