Object Detection is vital field in computer vision that helps detect objects like cars, weapons, and people within Images or videos. This when applied in the CCTV footages it can help detect weapons, suspects, cars and many more real-time to prevent any dangerous incident. This technique recognises, localizes and also detects multiple objects simultaneously. The application of this technology spans across various domains such as health care, security, surveillance, and advanced driver assistance systems. Humans possess a native ability to quickly and precisely detect objects, identifying multiple items in complex scenes with minimal mindful effort. In contrast, computer systems require considerable data, advanced algorithms, and powerful hardware to achieve similar performance. Modern advancements in technology, such as large datasets and high-performance GPUs, have significantly improved computer vision capabilities.

Why YOLO algorithm is used in our research?

YOLO (You Only Look Once) is superior to other object detection algorithms due to its real-time performance, single-stage detection approach, and high accuracy. Unlike traditional methods that use two-stage detectors (e.g., R-CNN), which separate region proposal and classification, YOLO combines these steps into a single network, significantly reducing inference time. This makes YOLO highly efficient, capable of processing images in real-time while maintaining competitive precision and recall. Additionally, YOLO's ability to predict multiple bounding boxes and class probabilities directly from full images, rather than using a sliding window or region proposals, enhances its robustness and versatility in various applications, particularly where speed and accuracy are critical.

Digital Image Processing (DIP)

DIP involves manipulating digital images through computational techniques to enhance or extract information. A digital image is represented as a 2D function f (x,y), where x & y are spatial coordinates, and the function value represents the image's intensity or color at those coordinates.

DIP is categorized into three levels of processing:

Low-Level Processing: Involves in noise reduction, contrast enrichment, and image sharpening with both Inputs & outputs being Images.

Mid-Level Processing: Includes segmentation, edge detection, and object extraction where the Inputs are images and the outputs are properties derived from Images.

High-Level Processing: This task understands and interprets images. Including complete scene and image analysis. This level involves insightful tasks associated to human vision.

Why Image Processing?

Image processing is essential for preparing digital images for viewing and enhancing their appearance. It includes:

- **Image-to-Image Transformations**: Modifying images directly.
- **Image-to-Information Transformations**: Extracting meaningful information from images.
- **Information-to-Image Transformations**: Converting information back into images.

Pixel and Resolution

- **Pixel**: The smallest unit of an image, representing the light intensity at a specific location. In an 8-bit grayscale image, pixel values range from 0 to 255.
- **Resolution**: It includes pixel resolution (total number of pixels), spatial resolution, temporal resolution, and spectral resolution. Higher resolution generally means better image quality but can be costly to achieve. Techniques like Super Resolution can enhance low-resolution images.

Gray Scale and Color Images

- **Gray Scale Image**: Represented by a single intensity function I(x,y), where x and y are spatial coordinates.
- **Color Image**: Represented by three functions for the primary colors (R, G, B). Conversion to digital form involves sampling coordinates and quantizing intensity values.

Techniques in Object Detection

Object detection involves many techniques such as feature based object which rely on manually crafted features like edges, corners, and textures. The Viola-Jones algorithm uses Haar-like features and a cascade of classifiers for real-time object detection, mainly face detection. The SVM Support Vector Machines (SVM) combined with Histogram oriented gradients (HOG) features provide a robust object detection. The Deep learning object detection models like YOLO, Faster R-CNN, and SSD (Single Shot MultiBox Detector) has revolutionized the entire field of object detection. These methods learn features automatically from massive datasets and provide results with high accuracy and efficacy by using a complex Convolutional Neural Networks (CNNs).

Deep Learning

This area of machine learning trains an object detection model using techniques that mimic how the human brain functions. In order to effectively identify, localize, and classify, artificial neural networks (ANNs) need to be trained to study and learn from larger data sets. In these networks, the phrase "Deep" refers to multiple layers of neurons that process information similarly to the human brain. These deep learning models work by sending data across several networked node levels. In order to boost the model's ability to recognize complex patterns, each layer enhances the input data.

Neurons and Layers

Neural networks can be artificial, made to solve AI problems, or biological, made of actual neurons. Weights are used to model the connections between neurons, and these connections can be either positive (excitatory) or negative (inhibitory). An activation function regulates the output, as the network processes the inputs. Deep Neural Networks have several layers between input & output layers, allowing them to model complex relationships.

A neuron or node in deep learning takes in input data, transforms it, and then sends the output to the layer above. This technique is comparable to how neurons in the human brain work. Layers are used to organize neurons. The first layer is responsible for receiving data and converting it into a numerical format that can be processed. The second type of layer is called hidden layers, which serve as intermediary layers and process the input through different calculations. The output layer, which comes in third, generates the finished product using the model's training data.

Each neuron in a layer generates an output by applying a mathematical function to the weighted sum of its inputs. Following layer receives this output after that. Objective here is the reduction of the error between the model's predicted & the actual values by weights adjustment during the training.

Activation Functions

When it comes to deciding whether or not a neuron should fire, activation functions are essential. They give the model non-linearity, which helps it pick up intricate patterns. Typical activation functions consist of:

- **Threshold Function**: A binary function that becomes active when an input value surpasses a predetermined level.
- **Sigmoid Function**: Frequently employed in binary classification applications, this function maps input values to a range between 0 and 1.
- **Hyperbolic Tangent Function**: Offers superior performance in some situations when compared to the sigmoid function, mapping input values to a range between -1 and 1.
- **Rectified Linear Unit**: This frequently used unit outputs zero for negative inputs and the input value itself for positive inputs. It is efficient and simple.

Learning and Training

Deep learning models learn by changing the weights of connections between neurons to minimalize the error in predictions. This process involves three basic stages, first is **Forward Propagation where the** Input data is passed through the network, layer by layer, to produce an output. The second is **Calculating the loss where** the model's output is compared to the actual target value using a loss function. The last is the **Back propagation, here the** error is propagated back through the network, & weights are adjusted to reduce the loss. The model cycles through these steps, continuously refining its weights and improving its performance.

Types of Neural Networks

- **Feed forward Neural Networks:** Data flows in one way from input to output, used for jobs like image classification.

- **Recurrent Neural Networks (RNNs):** Include feedback loops, allowing them to handle sequential data.
- **CNNs:** Specialized for processing grid-like data such as pictures, using convolutional layers to recognize features.

CNNs Architecture

CNNs generally consist of three types of layers:

1. **Convolutional Layers:** It use filters to convolve the input image and extract features. Each neuron inside a feature map is linked to a specific area of the input. The depth, stride, and zero-padding are key hyperparameters that affect the output volume.
2. **Pooling Layers:** It decrease the feature map's spatial resolution and spatial invariance is achieved. Max-pooling is commonly used for retaining the maximum value from every receptive field, reducing dimensionality and computational complexity.
3. **Fully Connected Layers:** These layers interpret features extracted by previous layers & perform high-level reasoning. The final output is usually generated by applying a softmax function or other classifiers.

CNN Training

Training CNNs involves adjusting parameters to minimize error using algorithms like back propagation. Over fitting is a common task, where the model achieves good performance on training data but bad on unknown data. Regularization techniques help moderate this issue.

R-CNN

R-CNN combines region proposals with CNNs to detect objects. It generates potential bounding boxes and classifies each using a CNN.

Faster R-CNN

The newest object identification model, Faster R-CNN, outperforms its predecessors with the integration of a Region Proposal Network into the architecture. A Fast R-CNN detector uses the high-quality region proposals. to categorize and improve item bounding boxes. Faster R-CNN greatly improves speed and accuracy by merging the

RPN with the detection network, enabling real-time object detection in images. It is a well-liked choice for a variety of computer vision jobs because of its effectiveness and performance.

SSD

SSD discretizes the output space of bounding boxes into default boxes of various aspect ratios and scales. It predicts object presence and refines box adjustments in a single forward pass through the network.

AlexNet

AlexNet is a pioneering CNN architecture with 5 convolutional layers, 3 fully connected layers, and 1 softmax layer. It was influential in demonstrating the effectiveness of deep learning for image classification.

AlexNet is a pioneering CNN architecture with 5 convolutional layers, 3 fully connected layers, and 1 softmax layer. It was influential in demonstrating the effectiveness of deep learning for image classification.

VGG

VGG is another CNN architecture known for its simplicity and effectiveness in image classification, utilizing deep convolutional networks with small receptive fields.

MobileNets

MobileNets are designed for mobile and edge devices, using depth-wise separable convolutions to reduce computational cost while maintaining accuracy.

TensorFlow

TensorFlow is an open-source library for numerical computation and machine learning, developed by Google. It supports constructing, training, and deploying object detection models with pre-trained models available for datasets like COCO and KITTI.

Caffe Framework

Overview

Caffe is a deep learning framework designed for speed, modularity, and openness. It supports various models and optimizations, making it suitable for tasks like object detection, semantic segmentation, and more. Caffe allows easy switching between CPU and GPU, which accelerates training times.

Caffe Models

- **OpenPose:** Detects human body, hand, and facial keypoints.
- **Fully Convolutional Networks (FCNs):** Used for semantic segmentation.
- **CNN-vis:** Generates images using CNNs.
- **Speech Recognition:** Implements speech recognition with Caffe.
- **DeconvNet:** For semantic segmentation.
- **Coupled GAN (CoGAN):** For generating paired images.
- **SegNet:** Real-time semantic segmentation architecture.
- **Deep Hand and DeepYeast:** Pre-trained models for specific tasks.

Python vs. Other Languages for Object Detection

Python is preferred for object detection due to its readable code, support for libraries like NumPy, and ease of use. It offers multiple libraries for machine learning and computer vision, making it a practical choice for implementing and testing algorithms.

OpenCV

A powerful, open-source library for computer vision and machine learning applications is called OpenCV (Open Source Computer Vision Library). It aims to speed machine perception in commercial products and provide a standardized infrastructure for computer vision applications. Businesses can simply utilize and change the code thanks to its BSD license. It was formally introduced by Intel Research in 1999 with the goal of advancing CPU-intensive applications. OpenCV offers an extensive collection of features for real-time computer vision applications, supporting multiple platforms and offering interfaces for MATLAB, C++, Python, and Java. With support for MMX and SSE instructions, the library is specifically designed for real-time vision applications and runs on Windows, Linux, Android, and macOS.

Haar Cascade Classifier

It is used for object detection in images. It works as follows:

- **Training**: Requires positive (face images) and negative (non-face images) samples. Features are extracted using Haar-like features, that compute the difference in pixel intensities.
- **Feature Calculation**: Uses integral images to speed up the computation. Each feature involves a simple sum of pixel intensities within rectangular areas.
- **Classifier Cascade**: Features are organized in stages, discarding non-face windows early in the process to improve efficiency. The final classifier combines weak classifiers into a strong one to improve detection accuracy.

YOLO

SSD, R-CNN and Fast R-CNN are some of the several object detection techniques. In 2015 Joseph Redmon, along with his colleagues introduced YOLO which detects different objects in an image. In 2016 it was officially discovered and it reframed the object detection with single regression. This unified model is able to predict numerous bounding boxes and class probabilities simultaneously.

Since its release, YOLO algorithm has outperformed many ongoing algorithms in terms of speed and accuracy to detect objects. The YOLO method can be utilised for a variety of Computer Vision (CV) activities, such as those involving animals, Ariel images, the military, autonomous vehicles, sports, hospitals, and others etc.

Numerous additional variations of YOLO have been created over time, including YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, and YOLOv8. Prior studies have shown that speed and accuracy of YOLOv5 is better than earlier versions. The latest version of YOLO, YOLOv8, is very efficient.

The base YOLO model processes pictures in real-time at 45 FPS. It's smaller version: Fast YOLO, processes an astounding 155 FPS while still achieving double the mAP of other real-time detectors. Compared to state-of-the-art detection systems, YOLO makes more localization errors but is less likely to predict false positives on background (Redmon et al., 2016).

A novel method for object detection called YOLO was introduced by Redmon et al. (2016). Complete training and real-time speeds are made possible by the YOLO, which maintains a high average precision. The input image is divided into a S × S grid by the system. An object's center falls into a grid cell, and that grid cell is in charge of detecting it. B bounding boxes and confidence scores for those boxes are predicted for each grid cell. The model's level of confidence that the box contains an object and its estimate of the box's accuracy in terms of predictions are both reflected in these confidence ratings. The difference between the expected and ground truth objects is the measure of confidence. In the event that there is nothing in that cell, the confidence ratings ought to be 0.

If not, the intersection over union (IOU) between the predicted box and the ground truth should be equal to the confidence score. Five predictions make up each bounding box: x, y, w, h, and confidence. The centroid of the box with relation to the grid cell's boundaries is represented by the (x, y) coordinates. The projected width and height in relation to the entire image are represented by the letters w and h. Ultimately, the IOU between the projected box and the ground truth box is represented by the confidence prediction.

The GoogLeNet model for image categorization served as the model for the network design. There are two completely linked layers in the network after 24 convolutional layers. 1 x 1 reduction layers are employed with 3 × 3 convolutional layers in place of the inception modules used by GoogLeNet. In Figure 4.1, the YOLO architecture is displayed.

Background of YOLO

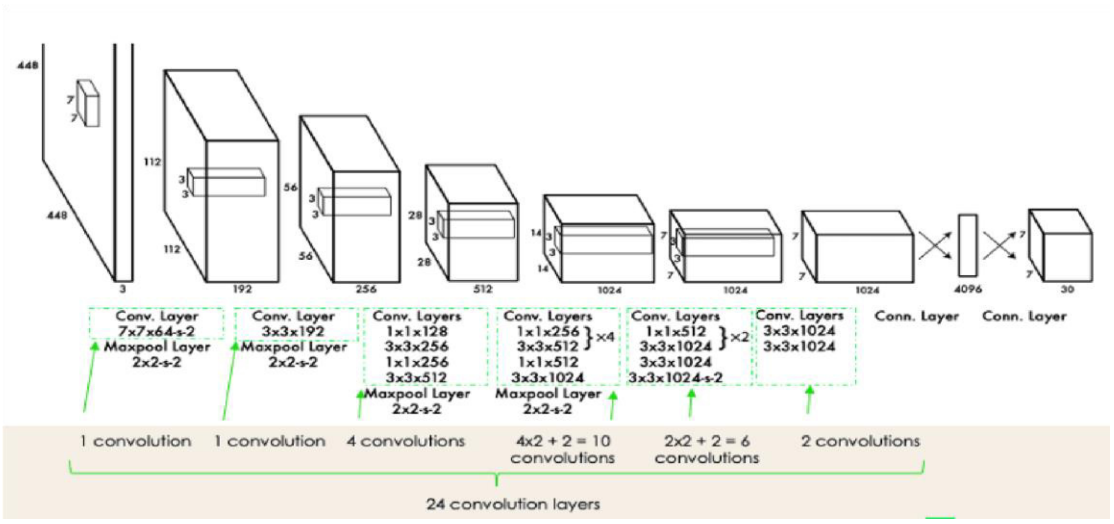


Fig. 4.1: YOLO Architecture (Redmon et al., 2016)

As further research was conducted to improve detections, the YOLO architecture was further improved by the inclusion of techniques and procedures to increase accuracy, decrease network size, and provide faster detections. These enhancements are enumerated in Table 4.1.

Improvements on YOLO Versions

From the introduction of the YOLO, there have been various changes and improvements which resulted in several versions of YOLO from YOLOv1 to YOLOv7. The findings on the YOLO versions are summarized in Table 4.1.

Table 4.1: Summary of Improvements on YOLO[1]

Sr. No.	YOLO Variant	Improvement	Results
1	YOLOv1 (Redmon et al., 2016)	SSD solves the problem of drawing boundary boxes and class identification.	Higher accuracy and speed compared to Faster R-CNN
2	YOLOv2 (Redmon & Farhadi, 2018)	Iterative improvements on Batch Normalization, higher resolution detection and use of anchor boxes	Architecture reduction, quicker and more accurate identification, and improved high-resolution image detection
3	YOLOv3 (Redmon & Farhadi, 2018)	Bounding box predictions now include an objectless score, additional backbone network layer connections, and predictions at three different granularities.	Improves detection of smaller objects
4	YOLOv4 (Alexey et al., 2020)	Enhanced feature aggregations, utilization of mish activation, and a bag of	Achieved improved accuracy and ease of training, high quality

[1] Olorunshola, O. E., Irhebhude, M. E., & Evwiekpaefe, A. E. (2023). A comparative study of YOLOv5 and YOLOv7 object detection algorithms. *Journal of Computing and Social Informatics*,

Sr. No.	YOLO Variant	Improvement	Results
		freebies with mosaic augmentations	performance and accessibility
5	YOLOv5 (Nepal & Eslamiat, 2022)	Reduced network parameters, use of Cross Stage Partial Network (CSPNet) for the head, PANet for the neck of the architecture, residual structure and auto-anchor. It also utilizes mosaic augmentations.	Extremely easy to train, inference on individual, batch images, video feed and webcam ports. Ease of transfer and use of weights. Faster and more lightweight than previous YOLO.
6	YOLOv6 (Chuyi et al., 2022)	Redesigned network backbone and neck to EfficientRep Backbone and Rep-PAN Neck. The Network head is decoupled separating different features from the final head	Improvement in detecting small objects, anchor free training of model. Less stable and flexible as compared to YOLOv5.
7	YOLOv7 (Wang et al., 2022)	E-ELAN-based layer aggregation, trainable goodie bag, and 35% fewer network parameters. Model scaling for models based on concatenation.	Increase in speed and accuracy, ease of training and inference.
8	YOLOv8 (Bochkovskiy et al., 2023)	Enhanced model architecture with a focus on reducing computational cost while maintaining accuracy. Integration of advanced data augmentation techniques.	Significant reduction in computational resources needed, with comparable or improved accuracy over previous versions. Improved real-time performance for embedded systems.

The primary distinctions between the architectures of YOLOv1, YOLOv2, YOLOv3, YOLOv4, and YOLOv5, according to Nepal and Eslamiat (2022), are that YOLOv1 employs the softmax function, whereas YOLOv2 has a better resolution classifier, higher accuracy, and higher efficiency than YOLOv1. It's because the CNN of YOLOv2 has a batch normalizing layer added to it. To fetch features from input image with more efficiency & detection performance, YOLOv3 employs Darknet53 as its fundamental backbone. Multi-object classification is available in YOLOv3, meaning that an object may simultaneously fall under more than one category.

To ascertain the likelihood that an input image corresponds to a certain label, YOLOv3 substitutes an independent logistics function for the softmax function. Additionally, YOLOv3 employs the 2-class entropy loss for every category, which reduces the computational complexity caused by softmax functions.

The backbone of the YOLOv4 design is CSPDarknet53, a hybrid of the CSP and Darknet53 networks. YOLOv4 has less hardware requirements and is more accurate and efficient at detecting objects. CSPDarknet53 serves as the backbone of the Focus structure used by YOLOv5. In YOLOv5, the Focus layer first appeared. The YOLOv3 algorithm's first three layers are swapped with the Focus layer. Using a Focus layer has the advantages of requiring less memory for CUDA, having a smaller layer, and having more forward & backward propagation.

YOLOv5

A researcher Glenn and his team produced YOLOv5, a new version of the YOLO family, a month after YOLOv4 was made available. Compared to YOLOv4, YOLOv5 is about 90% lighter and faster. Using RepVGG Style structure, EfficientRep Backbone, Rep-PAN Anchor-free paradigm, SimOTA algorithm, and SIoU bounding box regression loss function, YOLOv6 has many improvements in its backbone, neck, head, and training strategies. On the other hand, YOLOv7 outperforms all other known object detectors in terms of speed and accuracy in the range of 5 FPS to 160 FPS, and has the highest accuracy (56.8% AP) of all known real-time object detectors with 30 FPS or higher on GPU V100.

With a 50% reduction in computation and 40% reduction in parameters compared to the state-of-the-art real-time object detector, YOLOv7 significantly increased real-time object detection accuracy without raising the cost of inference. It also boasts a faster inference speed and higher detection accuracy.

Nepal & Eslamiat (2022) state that YOLOv5 differs from earlier releases in that PyTorch is used in place of Darknet. It makes use of CSPDarknet53 as its mainframe. To improve the information flow, it employs the Path Aggregation Network (PANet) as the neck. PANet uses a novel feature pyramid network (FPN) with many top-down and bottom-up layers. This enhances the model's low-level feature propagation. PANet increases the localization accuracy of the item by improving the localization in lower layers.

Furthermore, YOLOv5 shares the same head as YOLOv4 and YOLOv3, which provide three distinct feature map outputs in order to accomplish multi-scale prediction. The following is a summary of the YOLOv5 model: Support structure: CSP network, focus structure, Neck: PANet, SPP block, Head: GIoU-loss YOLOv3 head. The use of CSPDarknet53, which addressed the issue of repeating gradient information present in YOLOv4 and YOLOv3, allowed YOLOv5 to outperform YOLOv4 in terms of accuracy while decreasing the network's parameters and speed of inference. Figure 4.2 displays the YOLOv5 architecture.

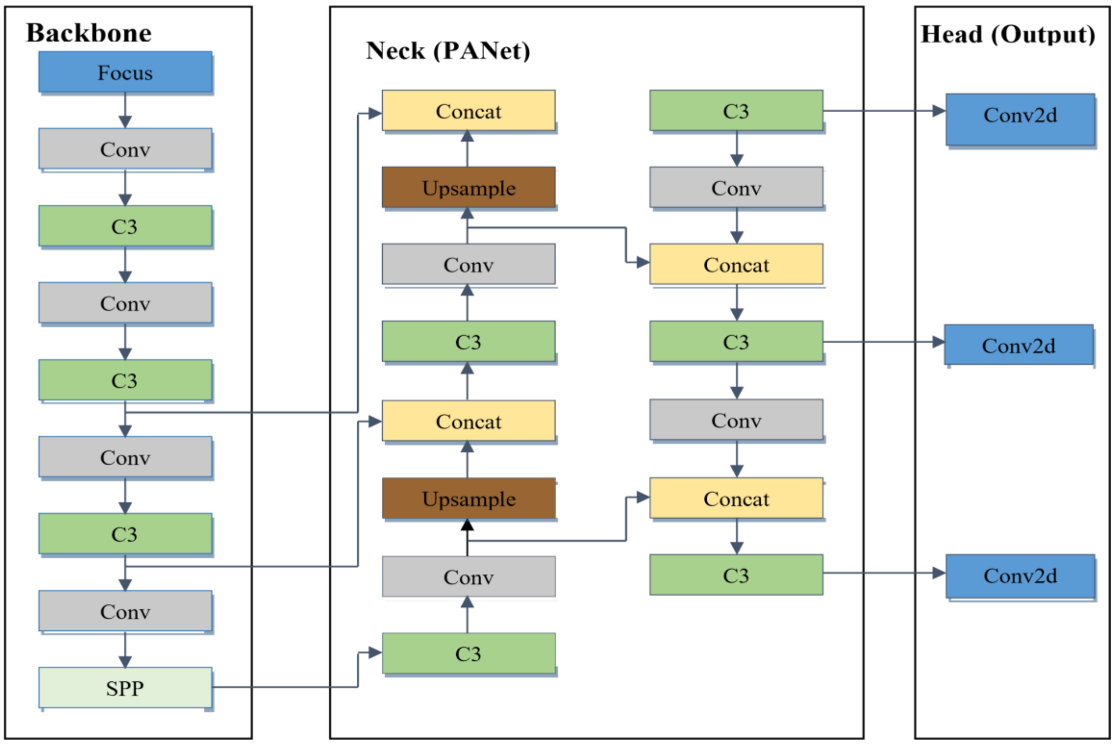


Fig. 4.2: YOLOv5 Architecture (Nepal & Eslamiat, 2022)

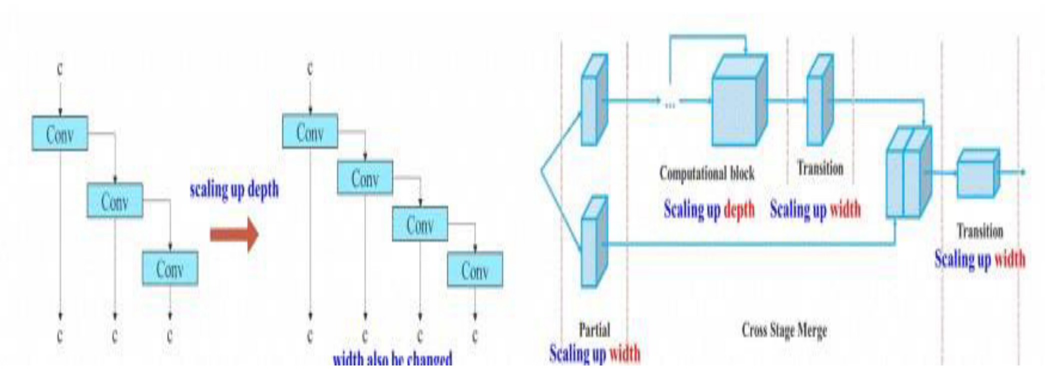
YOLOv7

YOLOv7 with its amazing features, YOLOv7 is a real-time object detector that is now changing the CV business. Without the use of any other datasets or pre-trained weights, YOLOv7 was only trained from scratch on the MS COCO dataset (Wang et al., 2022). According to Wang et al. (2022), YOLOv7 has the best accuracy at 56.8% AP among all known real-time object detectors with 30 FPS or above on Graphics Processing Units (GPU) V100. It also outperforms all other known object detectors in the range of 5 FPS to 160 FPS.

With around 40% fewer parameters and 50% less processing than the state-of-the-art real-time object detector, YOLOv7 significantly increased real-time object detection accuracy without increasing inference costs. It also boasts faster inference speed and higher detection accuracy.

Extended efficient layer aggregation networks (E-ELAN) are a feature of YOLOv7. To continuously improve the network's capacity for learning without erasing the initial gradient path, E-ELAN employs the expand, shuffle, and merge cardinality techniques (Wang et al. 2022). The architecture of the transition layer remains unaltered, while E-ELAN solely modifies the architecture of the computing block. Apart from preserving the original E-LAN design architecture, E-ELAN also helps various computational block groups to acquire a wider range of characteristics.

Model scaling for concatenation-based models is also available in YOLOv7. Model scaling is mostly used to modify certain model features and produce models at various scales in order to accommodate varying inference speeds. The suggested compound scaling approach can preserve both the ideal structure and the characteristics that the model possessed at the time of its original design. The model scaling for YOLOv7 concatenation-based models is shown in Figure 4.3.



(a) concatenation-based model (b) scaled-up concatenation-based model (c) compound scaled up depth and width for concatenation-based model

Fig. 4.3: Model Scaling of YOLOv7 (Wang et al., 2022)

From (a) to (b), it can be shown that the output width of a computational block likewise grows when depth scaling is applied to concatenation-based models. The next transmission layer's input width will rise as a result of this phenomena. Thus, (c) is suggested: for concatenation-based models, model scaling entails scaling only the depth of a computing block and scaling the appropriate breadth of the remaining transmission layer.

When compared to YOLOv5 and YOLOv7, YOLOv6 offers even greater advances in terms of detection, but it is less scalable and requires more effort to train. Additionally, YOLOv6 outperforms YOLOv5 and YOLOv7 in terms of accuracy when utilized for single image inference as opposed to multiple image inference (Banerjee, 2022). Because they are suitable for multiple item detection and make it simple to customize the training and inference process, YOLOv5 and YOLOv7 were used in the experiment.

YOLOv8

YOLOv8 is the latest advancement in the YOLO (You Only Look Once) series of real-time object detectors, building on the strengths of its predecessors to achieve higher accuracy and faster inference speeds. It features an optimized backbone and neck architecture that enhances the detection of objects of varying sizes and shapes, particularly improving the accuracy for small and densely packed objects. YOLOv8 also introduces improved model scaling, allowing it to adjust depth, width, and resolution to balance speed and accuracy depending on the hardware available, while maintaining the model’s structural integrity and feature representation. One of the key innovations in YOLOv8 is dynamic label assignment, which dynamically adjusts ground-truth labels during training based on the network's predictions, improving the model's focus on hard-to-detect objects. Additionally, advanced training techniques, including enhanced data augmentation and improved loss functions, contribute to better generalization and robustness against overfitting. YOLOv8 continues to optimize inference speed, making it well-suited for real-time applications, even on resource-constrained devices. Incorporating features like Cross-Stage Partial Networks (CSPNet), YOLOv8 reduces model size and computational complexity while retaining high representational capacity. Designed to surpass the performance benchmarks of YOLOv7 and other contemporary detectors, YOLOv8 represents a significant leap forward in the field of real-time object detection, maintaining a strong balance between speed and accuracy.

Applications of Object Detection

Facial Recognition

Facial recognition systems, such as "Deep Face" by Facebook and Google's facial recognition in Photos, identify and verify human faces using deep learning techniques. They analyze facial features like eyes, nose, and mouth for accurate recognition.

People Counting

Object detection aids in counting individuals for purposes such as security, store performance analysis, and crowd management. Challenges include handling fast-moving individuals and varying crowd densities.

Industrial Quality Check

In industrial settings, object detection improves processes like sorting, inventory management, and quality control by automating the identification and classification of products.

Self-Driving Cars

Self-driving cars use object detection to perceive their surroundings, integrating radar, LIDAR, GPS, and computer vision to navigate and avoid obstacles autonomously.

Security

Object detection enhances security through applications like facial recognition, retina scans, and real-time monitoring, aiding in criminal identification and surveillance.

Object Detection in Video Surveillance

In video surveillance, object detection is crucial for monitoring and analyzing video feeds. The process typically involves:

- **Pre-Processing**: Enhancing video quality and reducing noise.
- **Segmentation**: Dividing frames into meaningful regions to isolate objects.
- **Foreground and Background Extraction**: Identifying moving objects against a static background.
- **Feature Extraction**: Analyzing object characteristics for classification and tracking.

Effective video surveillance relies on robust object detection to handle challenges like varying lighting conditions and occlusions.