InfoTech Spectrum: Iraqi Journal of Data Science (IJDS)
ISSN: 3007-5467, Vol 2(No.2), 61-79 (2025)
DOI: https://doi.org/10.51173/ijds.v1i1.35
ijds.mtu.edu.iq

COMBINING YOLO AND SIFT TO DETECT CONFUSING OBJECTS IN IMAGES

Saba Makki Abed [1*]

[1] *Faculty of Sciences and Arts, Islamic University of Lebanon (IUL), Lebanon*
** Corresponding author E-mail: sabamake6@gmail.com (Saba Makki Abed)*

RESEARCH ARTICLE

ARTICLE INFORMATION

SUBMISSION HISTORY:
Received: 27 May 2025
Revised: 19 June 2025
Accepted: 27 June 2025
Published: 30 June 2025

KEYWORDS:
Object Detection;
YOLOv8;
SIFT;
Computer Vision;
Image Analysis;

ABSTRACT

Object discovery has advanced significantly with the emergence of deep learning models; however, existing algorithms often fail to deliver highly accurate and feature-focused detection, particularly in challenging visual environments. This study addresses the limitation by proposing a novel framework that integrates the high-level detection capabilities of YOLOv8 with the precision-focused characteristics of SURF-based feature extraction (referred to here as "Surzet"). The proposed method leverages YOLOv8 to perform comprehensive object detection while utilizing Surzet to enhance the identification of fine-grained features and local properties, ensuring robustness to scale and rotation. Experimental evaluation on complex image datasets revealed that this hybrid model significantly outperforms YOLOv8 alone, showing higher detection accuracy and a noticeable reduction in false positives. The initial results demonstrate that integrating YOLOv8 with Surzet creates a more reliable and precise object discovery framework. This approach holds great promise for high-level detection and detailed feature recognition applications.

1. INTRODUCTION

Object detection, a fundamental aspect of computer vision, is how digital images and videos are analyzed to identify and locate individual objects. It transcends the capabilities of image classification by determining the presence, location, and type of multiple objects within the scene. The progression of object detection has been dramatically influenced by algorithm advancements, data availability, and computational power [1]. In the early beginnings, traditional object detection techniques were rooted in applying handcrafted features and classifiers. One such technique was the Viola-Jones face detection method introduced in 2011. The algorithm efficiently detected faces using Haar-like features and the AdBoost learning algorithm. It introduced the concept of the integral image and used a cascading process to quickly eliminate non-face regions, making it suitable for real-time face detection. The Shift to Machine Learning and the limitations of handcrafted features led to exploring machine learning techniques to learn these features automatically from data. The HOG (Histogram of Oriented Gradients) descriptor combined with the SVM (Support Vector Machines) classifier became popular for pedestrian detection. This represented an evolution from crafted features towards features learned through machine learning methods. The Revolution in Deep Learning. With the rise of DL, object recognition made a big step forward. CNNs stand for "convolutional neural networks. Inspired by the human visual system's hierarchical structure, researchers have demonstrated significant improvements in image classification tasks3. The success in image classification set the stage for their application to object detection. R-CNN (Regions with CNN features) was among the early methods that applied CNNs to object detection, where region proposals were first generated and then classified using CNNs [2].

While deep learning models like YOLOv8 have advanced object detection, they often fail to detect fine-grained features, especially in complex images with scale and rotation variations. Existing algorithms fail to provide a feature-focused approach for accurate object discovery in challenging visual conditions.

DOI: https://doi.org/10.51173/ijds.v1i1.35

This study aims to develop and evaluate a new object detection framework that integrates the strengths of YOLOv8 with a SURF-based feature extractor (referred to as Surzet), to enhance detection accuracy and reduce false positives in complex image scenarios.

This paper is structured as follows: The introduction provides an overview of the object detection. The related work section reviews existing approaches. The problem formulation section discusses the specific challenges addressed in this study. The proposed method section details the integrated framework that combines YOLOv8 with the Surzet algorithm. The experimental results section presents the evaluation metrics, dataset details, and comparative performance analysis. Finally, the conclusion summarizes the key findings and outlines directions for future research.

1.1 Discovering Challenges and Confusing Things

Object detection has made significant strides over the years, largely thanks to advancements in computational capabilities and machine learning methodologies. However, while detecting distinct and well-defined objects has become increasingly accurate, detecting confusing objects — objects that closely resemble one another or are often obscured or distorted — remains a daunting challenge. This complexity is due to many factors, ranging from inherent ambiguities in the visual features of objects to the limitations of current detection algorithms. This article provides an introduction to some of these challenges [4].

- Visual Ambiguities and Resemblance: Many objects in the real world share similar visual characteristics. For instance, distinguishing between different breeds of dogs or between certain types of fruits like apples and pears from certain viewpoints can be challenging [5].
- Occlusions: Detecting heavily occluded objects, especially when the visible portion is non-distinctive, poses a severe challenge to object detection systems [6].
- Varied Illumination and Shadows: Changes in lighting conditions can alter the appearance of objects. Bright light can cause reflections, and low light can obscure details. Similarly, shadows can create deceptive visual cues that might lead detection algorithms astray [7].
- Deformations and Dynamic Changes: Objects are not always static. For instance, fabric in motion or a bent wire can take on numerous shapes and appearances. Objects with high degrees of flexibility and those that undergo dynamic changes present significant detection difficulties.
- Scale and Perspective Variations: Depending on the distance and angle from the camera, objects can appear vastly different. Detecting objects across various scales and perspectives requires the model to recognize the same object under various visual manifestations.
- Background Clutters and Overlaps: Objects may overlap in cluttered scenes, or the background might contain patterns and colors that confuse the detection algorithm. Distinguishing between the object of interest and visually similar background clutter remains a hurdle in many real-world detection tasks [8].
- Insufficient Training Data for Rare Objects: Machine learning models, intense learning-based ones, require vast amounts of annotated data. For everyday objects, there are ample datasets available. However, insufficient training samples for confusing or rare objects might lead to underfitting.
- Real-time Processing Needs: The algorithms must be accurate and fast, often requiring a trade-off between speed and accuracy.
- Domain Shifts: The subtle differences between the training data and the real-world application can lead to significant detection errors.

Addressing these challenges demands an amalgamation of advanced model architectures, improved training strategies, and richer datasets. While significant progress has been made, the journey to perfecting the detection of confusing objects underscores the intricate interplay between technology and the vast complexities of our visual world.

1.2 Rationale For Combining Yolo V8 And Sift

In the continuously evolving domain of computer vision, there is a relentless pursuit of higher accuracy, robustness, and efficiency in object detection. With the advent of deep learning, models such as YOLO (You Only Look Once) have emerged as frontrunners, offering real-time object detection with impressive accuracy. On the other hand, traditional feature extraction methods, like the Scale-Invariant Feature Transform (SIFT), have maintained their relevance due to their resilience in handling variations in scale, rotation, and viewpoint. The rationale for combining YOLO v8, a cutting-edge version of the YOLO series, with SIFT stems from the pursuit of harnessing the strengths of both deep learning and classical computer vision methods [8].

The YOLO architecture, specifically its latest iterations like v8, offers real-time object detection by viewing the detection problem as a regression problem. Unlike two-stage detectors that propose candidate regions and classify them, YOLO performs detection in a single shot. This streamlined approach significantly boosts processing speed. Additionally, with each subsequent version, YOLO has demonstrated improved accuracy in detecting objects of various sizes and dealing with overlapping objects[9].

SIFT is renowned for its ability to detect and describe local features in images. The keypoints detected by SIFT are invariant to image scale, rotation, and partially invariant to changes in viewpoint and illumination. Such attributes make SIFT exceptionally suitable for tasks like image matching, object recognition, and panorama stitching, especially in scenarios where the objects of interest transform or are viewed under different conditions [10]. While YOLO v8 is incredibly fast and generally accurate, it can sometimes miss out on smaller objects or objects that have undergone significant transformations. Integrating SIFT's feature extraction capabilities can offer a compensatory mechanism. SIFT's robust keypoint detection can potentially aid YOLO in recognizing objects that might otherwise be overlooked [11].

YOLO's bounding box predictions and SIFT's descriptors can provide richer information about detected objects. While YOLO can rapidly identify and localize objects, SIFT can offer detailed descriptor information about the localized regions, further enhancing recognition [12]. By combining the deep learning process of YOLO v8 with SIFT's traditional feature extraction robustness, the fused system may better generalize across a broader range of scenarios. This is particularly pertinent when training data might not capture all the diverse transformations an object can undergo in real-world scenarios [12].

2. BACKGROUND AND LITERATURE REVIEW

With the evolution of CNN-based detectors, a broad categorization emerged: one-stage and two-stage detectors. While two-stage detectors like R-CNN and its variants first generate region proposals and then classify them, one-stage detectors, such as YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector), perform detection in a single pass. YOLO, for example, divides an image into a grid and predicts bounding boxes and class probabilities simultaneously [3]. The introduction would then delve into the motivation behind combining these two techniques. YOLO is known for its speed and efficiency in detecting objects in real-time, making it a popular choice for applications requiring rapid processing [13].

On the other hand, SIFT excels in identifying and matching objects across different scales and orientations, providing robustness in varied scenarios. Combining these two methods aims to leverage the strengths of both: YOLO's speed and real-time capabilities with the detailed, scale-invariant features captured by SIFT. This hybrid approach is beneficial for identifying objects that are 'confusing' or difficult to detect due to occlusion, varying scales, or complex backgrounds [14]. The introduction would also set the context by discussing the combined method's relevance and applications. It could be particularly beneficial in areas such as autonomous vehicles, surveillance, or robotic vision, where accurately identifying ambiguous objects quickly is crucial Finally the introduction would outline the structure of the review, indicating how the subsequent sections will explore existing literature on YOLO and SIFT individually, the rationale for their integration, the methodology used in the combination, the challenges faced, and the results achieved in comparison to other existing methods. This provides a roadmap for the reader to understand how the review

will dissect and analyze the effectiveness and innovation of combining YOLO and SIFT for detecting confusing objects in images. Object detection is a pivotal field in computer vision, which seeks to locate and classify objects within images [15]. Traditional feature extraction methods, such as SIFT (large-scale invariant feature transform), and deep learning-based methods, such as YOLO (You Only Look Once), have had a singular impact. However, integrating these techniques provides a promising means, especially for detecting confusing image objects. In previous traditional automated processes, YOLO and SIFT may be used separately to achieve specific goals in image processing. Still, there is an attempt to improve detection accuracy in this context. Objects, especially those that are confusing or difficult to distinguish, are identified by the two algorithms' power. Therefore, the addition attempts to take advantage of each algorithm's advantages and overcome their potential limitations when used individually to achieve better results in detecting objects in images [16].

2.1 Traditional Feature Extraction: The Rise Of SIFT

Lowe's introduction of SIFT in 2004 revolutionized the landscape of feature extraction and matching in computer vision. SIFT's design allowed it to detect and describe local features in images invariant to scale changes, rotation, and illumination variations [1]. The algorithm's robustness against these variations made it a preferred choice for many applications, including image stitching, 3D reconstruction, and object recognition. Confusing objects in images—those with similarities in appearance, overlapping structures, or partially occluded objects—pose significant challenges for detection algorithms. While deep learning models like YOLO excel in many scenarios, their performance can be compromised when distinguishing between objects with subtle differences or cluttered scenes [2]. While robust against scale and rotation changes, traditional methods like SIFT may not always provide the semantic understanding needed to differentiate confusing objects.

2.2 Detecting Objects

Object detection plays a crucial role in various computer vision tasks, such as identifying pedestrians, recognizing faces, and detecting objects. Object detection has primarily developed into two distinct categories: traditional machine learning methods and deep learning methods. In traditional machine learning techniques, the initial step involves setting up what's known as feature engineering [3]. This process typically involves manually specifying features and can be broadly categorized into three different methods. One noteworthy approach, dating back to 1981, is the General Hough transform introduced by Ballard D. This method is particularly adept at extracting geometric features from data. Another method, which emerged in 1988, is the Harris corner detector, which focuses on identifying key object features by pinpointing corners in images. To achieve this, the Harris corner detector analyzes two images, extracting corner features, and then computes the degree of correlation between the points to detect objects. The two methods mentioned above are pretty attuned to the specific characteristics of the image. In simpler terms, alterations such as changes in image size, rotation, or grayscale can impact the outcomes they produce.

Lowe introduced SIFT in 2004 [4]. SIFT is an algorithm designed to identify and describe unique features within images. What's neat about it is that it treats each feature as an independent entity, which means that alterations in rotation and scale of the image won't mess with the results it provides. One striking difference emerges when you look at deep learning versus traditional machine learning: deep learning methods often don't rely heavily on intricate feature engineering. Moreover, they outperform traditional machine learning techniques, especially when training on vast datasets. The last decade has seen a remarkable surge in the advancement of deep learning, and this wave has also swept over the realm of object detection. In 2014, a pivotal moment occurred when Girshick and his colleagues introduced the R-CNN model[5]. This marked the first successful application of deep learning to object detection. In 2015, two significant advancements were made in computer vision. Fast R-CNN and Faster R-CNN [6] emerged as improvements upon the original R-CNN, significantly boosting its efficiency in both computation and training. Around the same time in 2015, a groundbreaking model known as YOLO was introduced. This model revolutionized real-time object monitoring, allowing for instantaneous detection and tracking of objects. In 2017,

computer vision was significantly developed with the release of YOLOv2, introduced in [7]. This version aimed to enhance the YOLO model, making it more accurate and efficient. The following year, 2018, two notable advancements were made in the field. First, Mask R-CNN emerged as a refinement of the Faster R-CNN model. Second, YOLO version 3 was introduced to improve YOLO version 2. These developments marked essential milestones in the ongoing evolution of object detection and image segmentation techniques [8]. Fig. 1 compares three tasks in computer vision: classification, localization, and object detection.

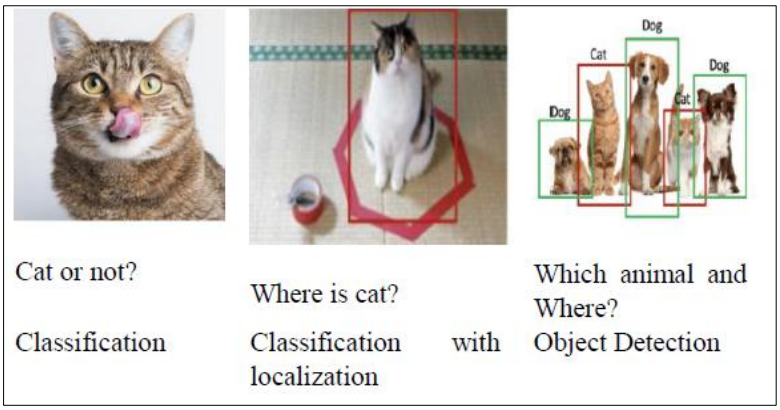

Figure 1: Common computer vision tasks

2.3 Object Detection Techniques

A computer vision challenge entails identifying and locating goods inside an image or video. Object recognition techniques and algorithms can be predominantly categorized into two main approaches: traditional computer vision methods and deep learning techniques [9]. Conventional computer vision techniques depend on manually designed features, heuristics, and algorithms to identify objects in images. Individuals initially evaluate diverse geometric characteristics of things for detection purposes. The object's curvature is ascertained by determining the minimal second moment orientation [10]. The evolution of learning has significantly transformed object detection, resulting in considerable improvements in accuracy and resilience. These methods primarily depend on CNN and have become the prevailing approach in recent years [11]. Prominent DL-based frameworks for object detection are comprised of:

- R-CNN: Geographically focused Convolutional Networks execute an area search and classification to find things in pictures. In 2012, thorough search method replaced the selective search method [12]. Fig. 2 depicts the process of an R-CNN.

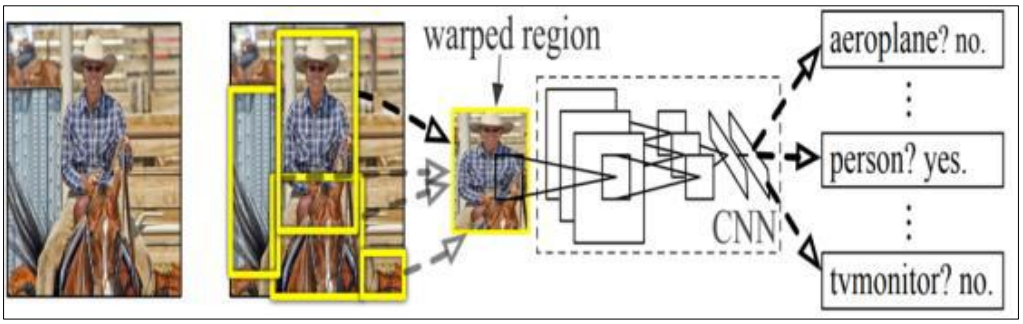

Figure 2: Region-based (R-CNN) Framework.

- Fast R-CNN: Fast Region-based Convolutional Network (Fast R-CNN) was established in 2015 by R. It utilizes the complete image as input for the Convolutional Neural Network (CNN), in contrast to R-CNN, which necessitates individual evaluations for each region proposal [13].
- Mask R-CNN, a Mask Region-based Convolutional Network, changed the field of computer vision when it came out in 2017. It could find bounding boxes and guess object masks, which was a big deal. In COCO challenges, this new method did better than others in identifying key points, bounding boxes, object instance identification, and object segmentation. R-CNN

Faster framework and the Mask R-CNN can be used together to make a mask for objects [14]: boundary box distances and category names.

- Faster R-CNN substitutes a RoIAlign layer on top of the main RoIPool layer, thereby improving localization precision. The second branch incorporates two convolutional layers, while the third branch is designated for object mask detection [15]. This procedure greatly enhances the accuracy of object localization and classification.

2.4 YOLO (You Only Look Once)

The YOLO method detects objects in real time using a single evaluation model and a neural network. Using an SxS grid, it predicts the image's bounding boxes and their associated probability [16]. YOLO distinguished itself with its innovative methodology in object redefined, making it possible to identify in just one forward network hop, significantly enhancing efficiency with its YOLO design, which has undergone numerous iterations and resilience. The following are the versions of YOLO models:

- The YOLOv1 series originated in 2016 when Joseph Redmon and colleagues introduced the groundbreaking YOLO methodology [8]. YOLO conceptualizes the two-stage process (area recommendation followed by categorization) as a regression problem by rejecting conventional techniques that saw object identification.
- YOLOv2, acknowledging the constraints of the initial architecture, the subsequent version, YOLOv2, sometimes referred to as "YOLO9000," implemented substantial enhancements.
- YOLOv3, a further evolution, resulted in YOLOv3, which integrated nuanced enhancements and effectively addressed real-world difficulties.
- YOLOv4 represents a significant advancement in the YOLO series, combining improved detection accuracy with exceptional speed.
- YOLOv8 in January 2023, Ultralytics introduced YOLOv8, marking a substantial advancement in deep learning and computer vision. This is the most recent addition to the esteemed YOLO line of object identification algorithms. Ultralytics, renowned for its association with the YOLO series through the launch of YOLO-v5, has persistently advanced the frontiers of object identification.

Since its debut, YOLO has continually led in integrating speed with precision. Every iteration has sought to enhance its forerunner, pursuing real-time detection while maintaining the integrity of forecasts. The advent of YOLO-v8 has generated significant enthusiasm within the research community, as we anticipate the forthcoming study that will elucidate the advances and architectural modifications implemented. The architectural modifications will enhance the algorithm's scalability and adaptability [23]. Edge devices, like smartphones, smart cameras, and IoT gadgets, exhibit processing power and memory capacity limitations. Implementing deep learning models on these devices presents distinct problems. The model must be lightweight, efficient, and simultaneously effective. YOLO-v5 represented a substantial advancement in that regard. Preliminary evidence from Ultralights is pushing this limit even further, as shown by YOLO-v8. YOLO-v8 could change applications that need to respond instantly to object detection by focusing on fast inference speed on limited hardware. These applications could include security systems that detect threats in real time and smartphone applications that use augmented reality contexts.

3. THE PROPOSED SYSTEM

Our system utilises deep learning methodologies, notably, there are two algorithms: YOLO and SIFT. Regarding real-time object detection, YOLO is widely praised for its effectiveness. In contrast, SIFT is frequently used for the extraction of features and matching. The integration of various methodologies possesses the capacity for precise and effective item classification and detection. The proposed system includes many phases and procedures, indicating a sequence of operations that data experiences. Fig. 3 represents the system's workflow.

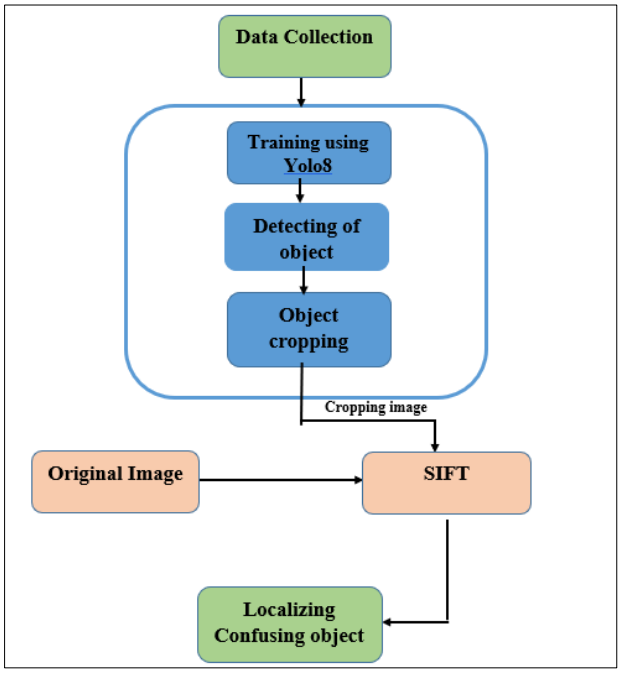

Figure 3: General block diagram of the proposed.

3.1. Data Collection

Pascal VOC 2012 is a widely adopted benchmark initially introduced for the Pascal (VOC) and builds upon its predecessors. Pascal VOC 2007 and 2010 comprise 20 prevalent object categories, including aeroplanes, bikes, birds, boats, bottles, automobiles, cars, dining tables, cats, and so on [28]. Moreover, specific object categories are accompanied by pixel-level segmentation masks, enabling more detailed object classification assignments .The dataset is thoughtfully divided into groups for testing and training, rendering it suitable for training object detection models and evaluating their performance. The training subset includes many images for model training, while the testing subset is designated for assessing model accuracy. Notably, the images contained within the Pascal VOC 2012 dataset encompass a wide range of scenes, backgrounds, ambient light, and the direction of the objects, presenting a formidable challenge for object detection algorithms.

The dataset consists of 17,112 photos, divided into test, validation, and training sets. Training and object detection system evaluation find their foundation in these image datasets. Training has used various pictures depicting twenty common object groupings with different forms. YOLO provides a mosaic zoom technique that amalgamates random segments from four photographs into a cohesive composition. YOLOv8, a YOLO model that uses a close mosaic, terminates this strategy. Providing the model with continually unrealistically impossible scenarios is not the goal.

3.2 Model Architecture

3.2.1 Convolutional Neural Network (CNN)

CNNs are a variant of deep feed-forward artificial neural networks frequently employed in computer vision tasks, including image classification. Convolutional Neural Networks (CNNs) differ from conventional multilayer perceptron (MLP) networks by incorporating convolutional layers, pooling operations, and nonlinear activation functions, such as the tanh, sigmoid, and ReLU types. The three-dimensional arrangement of neuron layers in CNN architectures—input and hidden layers—makes them ideal for processing large amounts of multi-channel picture data, making CNNs suitable for image acquisition. To reduce the amount of weights that need to be calculated and their redundancy, neurons in a layer are connected to only a brief segment of the layer before convolution, which is the spatial action on small parts of the input image. One type of neural network is the convolutional neural network, which uses feedforward architectures that excel in processing large datasets due to their convolutional function, which enables neurons to encompass peripheral units within the convolution kernel. A convolutional neural network comprises a fully connected layer integrated with a pooling layer and one or more convolutional layers. Convolutional neural

networks yield exceptional results in audio and image recognition. It requires fewer parameters than other deep neural networks. Convolutional neural networks are frequently employed deep learning models owing to their advantages. Presented herein is a succinct elucidation of the foundational architecture of convolutional neural networks.

3.2.2 Primer on CNN's Architecture

Fig. 4 illustrates the fundamental architecture of the CNN system. Every output component in the fully connected layer is linked to the input.

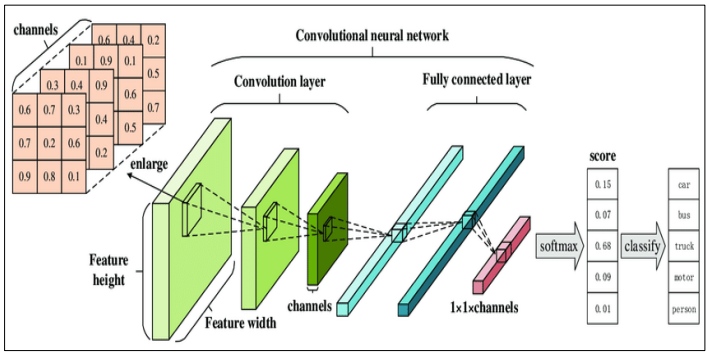

Figure 4: The architecture of CNN.

Utilizing a small square matrix, convolution is a way to pull out details from an input picture. A feature map is the x-y table that traverses the original image with the filter, where each neuron possesses identical weight parameters. Various filters applied to the same image can produce distinct feature maps, resulting in effects such as edge detection, sharpening, and blurring through the modification of filter parameters. Fig. 5 shows the result of applying a convolutional kernel, also known as a filter, on a CNN source layer. Convolution preserves spatial information from the input data, as seen in the example that computes a value of 5 at the corresponding location of the destination layer for a single kernel point.

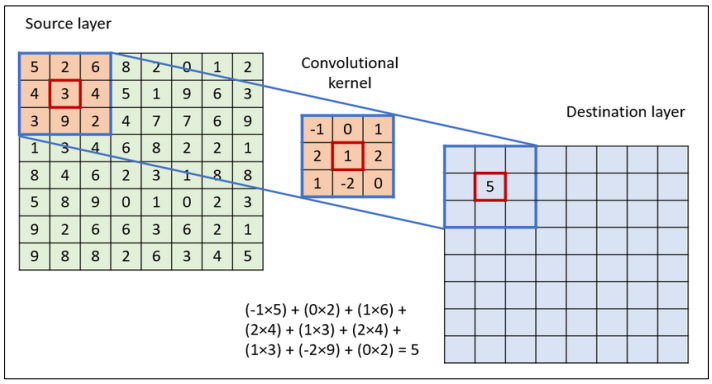

Figure 5: a process known as convolution [29].

In CNN, this layer converts aggregated feature representations into more utilitarian forms, preserving critical information while eliminating superfluous features. It guarantees uniformity despite positional or lighting fluctuations, provides resilience against clutter, and yields a concise, interpretable representation. Feature maps have less detail because the pooled layer consolidates outputs from neighboring neuron groups, increasing their resilience to input distortions.

$$Y_{kij} = \max_{(p,q)\in R_{ij}} X_{kpq} \quad \ldots (1)$$

In this context, ykij denotes the output of the pooling operator associated with the kth feature map. In contrast, xkpq refers to the element located at (p,q) within the pooling region Rij. A limited area surrounding the coordinates (i,j). The average pooling method utilises the arithmetic mean of the elements within each pooling region, as demonstrated in the equation. 2. This layer sorts the data by using traits found by earlier layers and filters to put them into groups. It connects every neuron in one layer to every neuron in the next layer, most of the time using a softmax activation function to give chances ranging from 0 to 1.

$$Y_{kij} = \frac{1}{|R_{ij}|} \sum (p,q) \in R_{ij} \; X_{kpq} \; \ldots (2)$$

In neural networks, transfer functions compute the weighted total of inputs and biases, hence defining the activation of a neuron. They alter data by gradient processing, typically via descending a gradient, and make output based on the factors fed in. AFs change outputs in various uses, such as recognizing objects, categorization, and machine learning. The output layer generates the final predictions using information from the preceding phases. This layer generally utilises soft ax activation to transform the raw outputs from the previous layer into probability scores, reflecting the likelihood of each class in tasks such as picture classification.

3.2.2 YOLO Algorithm

It uses a single review and a single neural network model to find bounding boxes and guess how likely they will happen. It's easier to use because it can make predictions in real time. The YOLO method takes the whole picture as input and breaks it into an SxS grid of cells. Bounding boxes and confidence numbers are in each cell of this grid. To get the confidence score, multiply the chance of finding an item by the value of the intersection over union (IoU) between the real boxes and the projected boxes. The Google Net convolutional neural network, which includes the inception module, is used [17]. In the YOLO system, there are 24 convolutional layers in this module, followed by two fully connected layers. A 3x3 convolutional layer is added instead of the inception module. It comes after a dimensionality reduction layer and one of three filter sizes (1x1, 2x2, or 3x3). There are three versions of YOLO: v1, v2, and v3. The fastest one is v3, which has nine convolutional layers and a set number of filters.

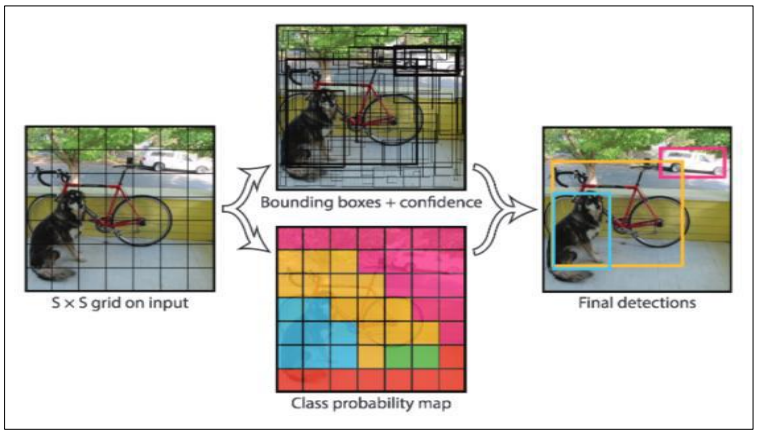

Figure 6: YOLO framework.

Along with the plan, the RoIAlign layer makes translation-equivariance and scale-equivariance easier. ResNet is used in this method and works on the picture input over 101 layers. The RoIAlign levels take care of the Regions of Interest found. This network has a fully connected layer that is added through one of the three branches. These branches look at the bounding box coordinates and predict the probability. In YOLO, the last layer that predicts each grid cell gives back S*S*(C+B*5), where S*S is the overall grid size, C is the expected probability for each class, and B is the number of anchor boxes per cell that are linked to confidence scores and four coordinates, Fig. 6.

The ImageNet dataset is utilised for classification, having been pre-trained with almost half of the Convolutional layers. In contrast to previous methodologies, YOLO frequently produces many bounding box forecasts that lack objects. The Non-Maximum Suppression (NMS) technique is employed after the network to resolve this issue. NMS consolidates overlapping bounding boxes into a singular box for identical objects; yet, instances of false positives in detection may still occur.

3.2.3 SIFT Algorithm

It is a method developed by David Lowe in 1999. It has widespread applications in image processing and computer vision, particularly feature identification and alignment. SIFT is distinguished for its capacity to preserve precision despite variations in scale, rotation, illumination, and perspective. Consequently, it demonstrates significant use in object recognition, image mixing,

and reconstructing three-dimensional scenes. Visual data can be used to get highly unique and consistent features using the SIFT method. SIFT aspects stay the same when scale, rotation, lighting, and viewpoint change [18]. Using SIFT-extracted features enables reliable matching of the same objects across different photos. When implementing SIFT on images, the fundamental computational procedures for creating a set of image features are as follows:

- The preliminary SIFT phase entails a comprehensive search across several scales and image locations [19], effectively executed by the Difference of Gaussian (DoG) technique as depicted in Figure 7.

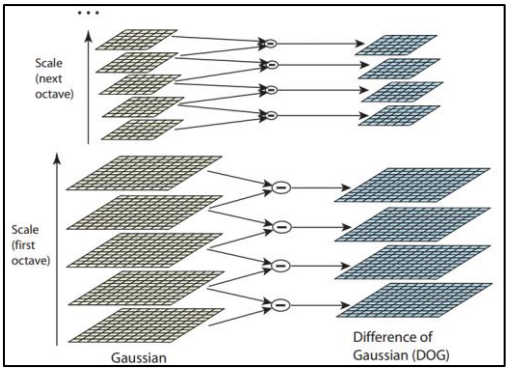

Figure 7: DoG space [37].

A comprehensive model is applied at each prospective site to ascertain the exact placement and magnitude[20]. Key points undergo a contrast and edge assessment to preserve the stable ones [21] Fig. 8.

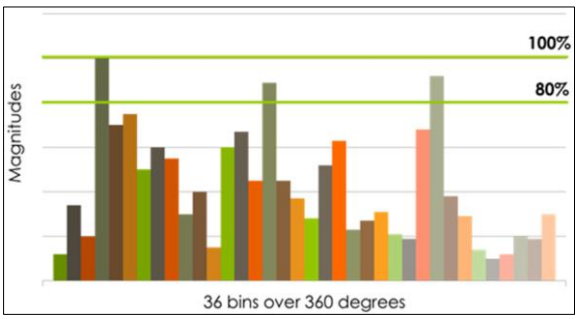

Figure 8: Histogram of dominant orientations.

A descriptor is generated to signify each key point based on the previously assigned orientation. The descriptor, which relies on the gradient histogram in the image, is computed as shown in Fig. 9.

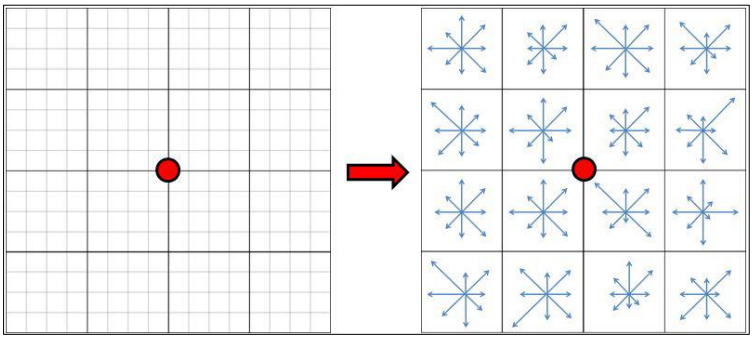

Figure 9: Key point descriptor [39].

3.3 Evaluation Metrics

Including both precision and recall in this metric is essential for judging how well object detectors work because it gives a complete picture of how well the model can spot things in images across multiple categories.

- **Average Precision:** is often used to check how accurate a model's forecasts are. It gives a number showing how well the model can find and identify items in an image. It usually judges how well a model balances accuracy and memory.
- **Precision:** quantifies the proportion of projected positive cases that are true positives. It computes the proportion of accurate forecasts**.**

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \qquad \ldots (3)$$

- **Recall:** Conversely, assesses the model's efficacy in identifying all genuine positive events within the dataset. It **computes** the proportion of true positives accurately detected by the model.

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \qquad \ldots (4)$$

- **Average Precision (AP):** The measure is established by looking at the precision-recall curve for a particular class, which means figuring out the precision at different recall levels: thresholds (often from 0 to 1) and the area under this curve. AP thoroughly evaluates the model's capability to accurately recognize things within a specified category.

$$\text{AP} = \frac{1}{n}\sum_{i=1}^{n}(Ri - Ri - 1).pi \qquad \ldots (5)$$

Where: AP Average Precision is denoted as AP, and n is the number of points on the curve for accuracy and recall. Ri is the recall at point i, Pi is the precision at point i, and Ri-1 is the recall at point i-1 that came before.

- **Map:** An object identification model's overall performance over several classes is evaluated statistically by mean average precision. You construct mAP by independently computing the AP for every class, then averaging these AP values. This offers one value that captures the general performance of the model over all object types.

4. RESULTS OF EXPERIMENTS

The range from inadequate to exceptional is contingent upon the degree of correspondence between the red-labeled anticipated boundary box and the blue-labeled actual one, where iou is the ratio of the overlapped area to the combined area of the two boxes, Fig. 10.

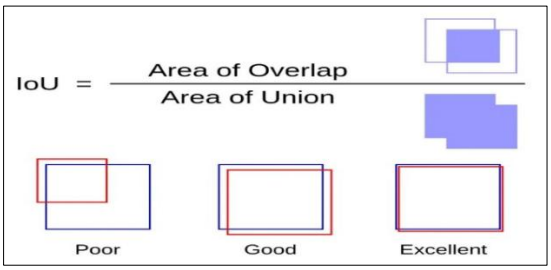

Figure 10: Compute the Intersection Over Union (IoU).

The training step involves calculating the error in a loss function, with the primary metrics being box_loss and cls_loss:

- Box_loss quantifies the mistake in bounding box detection.
- Cls_loss measures the error in identifying object classes.

This differentiation facilitates a more nuanced comprehension of model efficacy. The model may successfully determine the exact location of the square surrounding an item whose category is wrongly defined. If the model learns from the data efficiently, you should see a gradual decrease in these values over time. Both box_loss and cls_loss decreased, as we observed. After training the model, the validation stage uses the validation dataset to evaluate the model's quality. Mean Average Precision (mAP50-95) is the most essential quality metric. Accuracy should increase from one iteration forward if the model improves and learning in the subsequent one. We observed a progressive enhancement.

4.1 Training

The training and validation phases are the two main components of a training cycle. During the training phase, the "train" technique is used. It uses the training dataset to get a random selection of images, allowing for the specification of batch size. The model subsequently processes the photos, yielding boundaries for each object found and labeled with its class. The output is subsequently passed into the loss function, which evaluates it against the precise results extracted from the image annotation files. This function quantifies the degree of error. The loss function's output is subsequently transmitted to the optimiser, which appropriately modifies the model's weights to reduce mistakes in the following iteration. The default optimiser is SGD (Stochastic Gradient Descent); however, alternative options like Adam may be explored for potential enhancements. During the validation phase, the following operations are carried out by the train function the bounding boxes for each of these images are identified after the model processes them. The model's accuracy can be calculated by looking at the differences between the expected and actual results, which are acquired by comparing the generated results with the exact values from annotated text files. The model's learning and development from one epoch to the next is achieved by looking at the progress and results for each phase inside each epoch.

4.2 Evaluation and Results Metric

The model's generated images and accompanying predicted bounding boxes show satisfactory performance, according to the evaluation measures. Prediction plots show that F1 scores for all classes are between 0.68 and 0.40, precision is between 1.00 and 0.99 across all classes, recall is always 0.70, and the recall curve is 0.88 as the model successfully identified all images of persons, cars, and dogs. Nonetheless, there are accuracy concerns, as the program occasionally produced duplicate predictions for the same three individuals or the identical vehicles, and the cause may be the lateral perspective of individuals, vehicles, or canines. The model achieves a commendable mAP score, a commonly utilised statistic for object identification models. Based on using the physical truth notes that come with the validating pictures, the model guesses the size of the box that shows a significant intersection over the union time required to prepare each image in the given dimensions (1, 3, 480, 640) is 45.7 ms, inference is 207.4 ms, and post-processing is 7.7 ms.

4.3 Confusion Matrix

Making a Matrix of Confusion: Both models will be used to make predictions for each test picture YOLOv8 will give you predicted SIFT may provide you with feature descriptors and occasionally, object classes, in contrast to bounding boxes and object classifications Determine each model's similarity score or confidence score result should be Because these limits can be changed the balance between precision and recall can be adequately managed Setting up correspondences It is necessary to line up YOLOv8's predictions with SIFT's forecasts for each expected item to make a confusion matrix The matching could be based on how the items are classified where the bounding boxes meet or a mix of the two. Constructing the confusion matrix produces:

- True Positives: Both YOLOv8 and SIFT accurately identify the object as ambiguous.
- False Positives: YOLOv8 identifies the item as ambiguous; however, SIFT does not.
- False Negatives: YOLOv8 fails to identify the item; however, SIFT recognises it as ambiguous.
- True Negatives: YOLOv8 and SIFT accurately ascertain that the object is unambiguous.

To determine how well the combined YOLOv8 and SIFT method works, take the numbers from the conflation grid and use them to figure out several assessment measures, such as accuracy, precision, recall, and the F1-score. For your reference, the reason behind the confusing matrix is shown below:

- Accuracy: assesses the overall accuracy of the predictions.
- Precision: assesses the precision of affirmative forecasts
- Recall: assesses the capacity to identify all good events.

- F1-Score: The F1-score provides a happy medium between recall and precision using a harmonic mean calculation.

The confusion matrix shows that "Actual Confusing" items are hard to understand, while "Actual Not Confusing" items are easy to understand. The model's predictions are used to fill in the matrix cells.

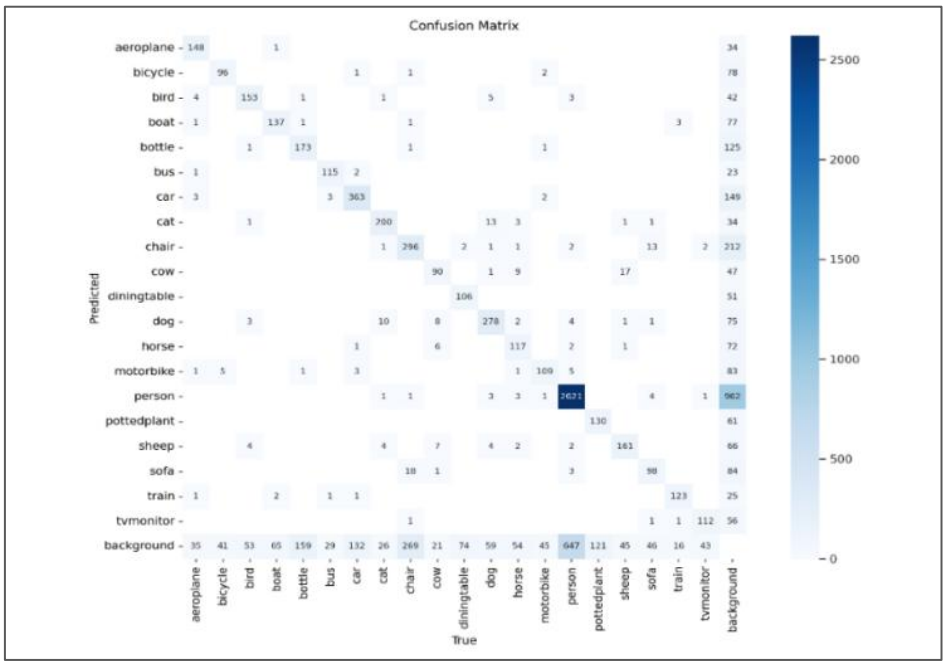

Figure 11: Confusion matrix for all classes.

Fig. 11 shows a confusion matrix. It shows how well the model has predicted different classes by comparing the accurate labels (along the horizontal axis) with the predictions made by the model (along the vertical axis). The diagonal numbers represent correct predictions, while off-diagonal numbers indicate misclassifications. The larger numbers on the diagonal, the better the model's performance. Table 1 shows a performance table for an object detection model, listing various classes of objects and the corresponding metrics to evaluate the model's predictions: Accuracy, Precision, Recall, and F1-Score. Each metric is given as a decimal, which can be interpreted as a percentage (e.g., 0.99 is 99%). Based on the supplied data, the average percentages for the metrics are as follows: Precision: 93.45%, Recall: 93.60%, F1-Score: 93.55%. The model's overall accuracy is 96.09%.

Table 1: The performance table for an object detection model

	precision	**recall**	**f1-score**	**support**
aeroplane	1	1	1	148
bicycle	1	1	1	99
bird	1	1	1	133
bus	1	1	1	115
car	1	1	1	363
cat	1	1	1	700
chair	1	1	1	296
motorbike	1	1	1	763
person	1	1	1	2403
background	1	1	1	937
accuracy	1	1	1	1
macro avg	1	1	1	5957
weighted avg	1	1	1	5957

Fig. 12 shows person1, person2, car1,car2, and dog object detection that was trained for only 25 epochs, and then no error in weight and stopped the pre-trained Yolov8n model.

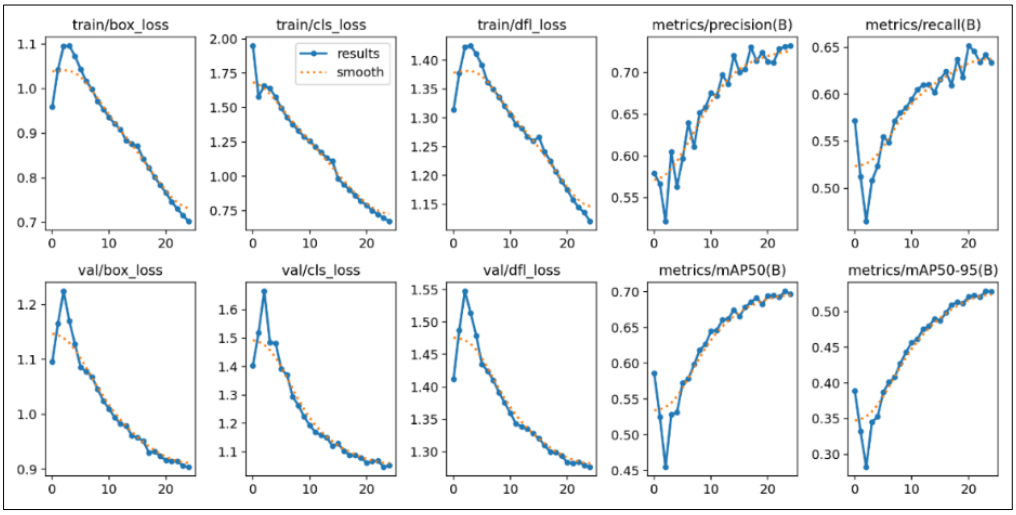

Figure 12: Trained stage for only 25 epochs.

Figure 13: Instances of object detection in images utilizing the proposed technique. train_batch_0

4.4 Validation

Fig. 13 and Fig. 14 depict the system's outputs for detecting objects. They give you pictures with boxes around the objects you pointed out. Every container is designated with a numerical or categorical identification, potentially signifying the detected object type (e.g., "14" for individuals or "7" for vehicles), and select boxes feature confidence scores, representing the model's assurance over the detection. These photos illustrate the model's capacity to identify and discern various items inside an individual image or throughout a sequence of images.

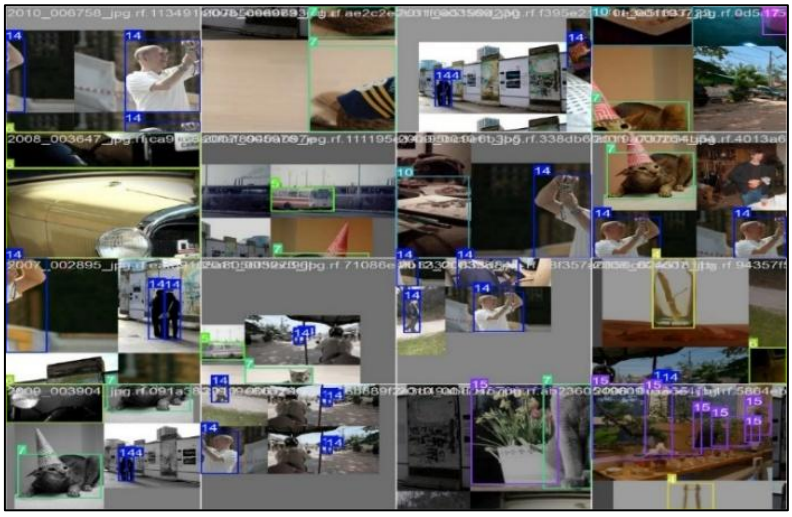

Figure 14: Instances of object detection in images utilizing the proposed technique. train_batch1.

4.5 Testing

The goal is to use the YOLO model to recognize objects and make predictions, usually using a model that has already been trained. We set a confidence level for identification at 0.25. Objects with confidence scores below that level may be thrown away. After that, we gave the picture file we wanted to recognize, "V8.jpeg." Parts of the identified objects are also cropped from the image, with the chosen objects enclosed by bounding boxes. Combining the "Image" and "Display" functions allows you to show YOLO detection data that includes images with edges. After that, we list folders inside the "detect_dir" directory. Each one could represent a different prediction or YOLO execution. Based on its timestamp, the subfolder that has been changed the most recently is chosen, and the directory with the newest predictions is found. After that, we make the file path to the "V8.jpeg" picture in this newest subfolder, representing the most recent YOLO forecast. The results are finally shown using the Image and View functions. They show the results of YOLO object identification with boxes around the identified items. The "height = 600" argument sets the height of the image to 600 pixels. Using YOLO to find objects in an image called "V8.jpeg," saving the results, and then showing the image with the found objects outlined by boxes that highlight the most recent YOLO forecasts is what the process involves.

4.6 Testing Results of The Sift Algorithm

Upon identifying things inside the image, it subsequently displays it featuring these recognized objects. Furthermore, it uses the SIFT (Scale-Invariant Feature Transform) technique to identify key points between two photos. The method entails the examination of a singular image titled "V8.jpeg," identifying various items, including person1, person2, person3, car1, car2, and a dog. Furthermore, it offers insights into the duration allocated to several phases, including preceding, inferring, and subsequent processing.

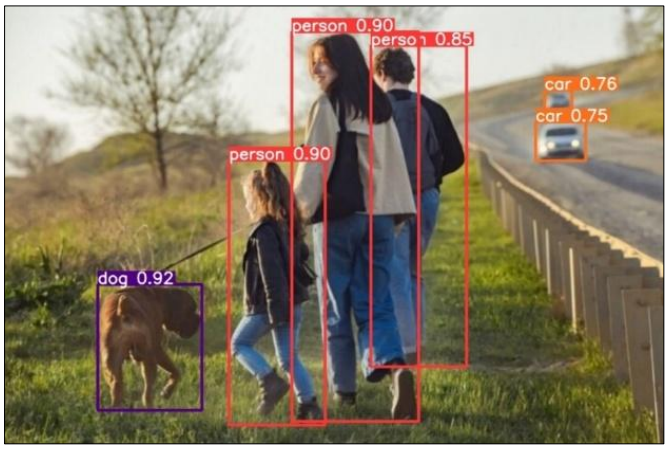

Figure 15: A Single image detects person1, person2, person3, car1, car2, and dog.

Fig. 15 shows the test outcomes of an artificial intelligence model for finding objects:

- **Person:** The person was identified with a confidence of 0.85. This means the model is 85% sure that the object is a person.
- **Person:** The person was identified with a confidence of 0.90. This means the model is 90% sure that the object is a person.
- **Person:** The person was identified with a confidence of 0.90. This means the model is 90% sure that the object is a person.
- **Car**: The car was identified with a confidence of 0.75. This means the model is 75% sure that the object is a car.
- **Car:** The person was identified with a confidence of 0.76. This means the model is 76% sure that the object is a car.
- **Dog:** The dog was identified with a confidence of 0.92. This means the model is 92% sure that the object is a dog.

The findings reveal that the model comprises 168 layers, totaling 3,151,904 parameters, with

no gradients (indicative of the inference phase rather than training). In the photograph, the model identifies one person1, one person2, one person3, one car1, one car2, and one dog. The model offers a detailed breakdown of the time allocation throughout the object detection process:

Preprocessing consumes 3.0 ms for image preparation. The core object detection process, known as inference, demands 114.1 ms. Post-processing, entailing result refinement and storage, requires 89.1 milliseconds. These timings pertain to the input image, characterized by the format (1, 3, 480, 640), representing a single image with three RGB color channels and a resolution of 480x640. The procedure initiates by loading two images, 'V8.jpeg' and 'V82.jpeg,' through OpenCV's 'cv2.imread' function. It proceeds to exhibit the 'V8.jpeg' image using 'cv2_imshow' and showcases the image "V82.jpeg" using the same method.

Figure 16: Use the key points and descriptions of two pictures to find matches between them.

Fig. 16 and Fig. 17 show an image of the discovered universe, which is cropped from the original image so that we get a cropped image of the game. Subsequently, it generates a SIFT object using OpenCV's 'cv2.xfeatures2d. SIFT is an algorithm designed for detecting and describing features, primarily used in image matching and object recognition.

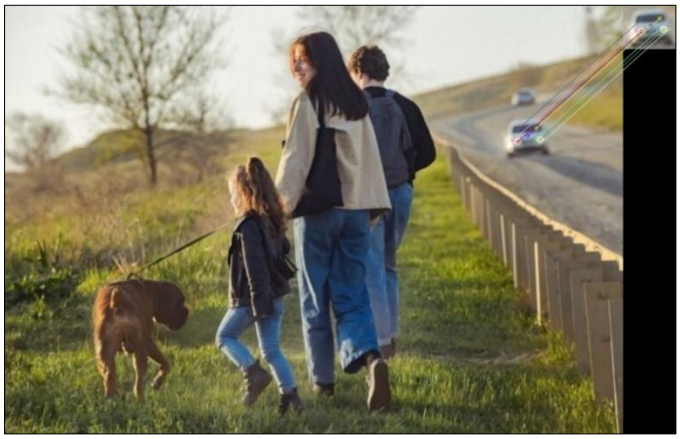

Figure 17: Put the resultant picture up as the img3 with the lines drawn between the important

SIFT is employed on both the 'V8.jpeg' image and the 'V82.jpeg' template image to obtain key points ('kp1' and 'kp2') and descriptors ('des1' and 'des2') for each image. These key points and descriptors are pivotal for locating matches between the two images. The next step involves the creation of a Brute-Force Matcher ('bf') using OpenCV's 'cv2.BFMatcher This matcher is utilized to identify the top matches between the descriptors of the two images using the 'knnMatch' function, with a parameter 'k=2' signifying the return of the two best matches for each keypoint. A ratio test is applied to the matches to filter and retain the most reliable matches. This test checks whether the distance of the best match is less than 0.5 times the distance of the second-best match. If this condition is met, the match is categorized as "good" and stored in the "good" list. Finally, using cv2.imshow, the model displays the findings on the V8.jpeg picture, generating a visualization of these good matches. The final image, with all the lines drawn to indicate the essential spots, is saved as the variable img3 spots.

5. CONCLUSION

Integrating YOLO and SIFT techniques represents an innovative stride in object detection within images, especially with elements that are difficult to distinguish. Merging YOLO's rapid detection framework with the detailed feature analysis of SIFT creates a robust system for identifying objects under challenging conditions. However, this combination introduces computational complexity, highlighting the need for optimization to ensure real-world applicability. Despite its promise, this fusion may not be universally optimal. Identifying scenarios where this combined approach yields the most significant advantages is an active area of research. Ultimately, the synergy between YOLO's deep learning prowess and SIFT's precise feature extraction exemplifies the dynamic advancements occurring in computer vision, as researchers seek to refine and adapt these methods to diverse and complex visual tasks.

CONFLICT OF INTEREST

The authors declare that there is *no* conflict *of interest* regarding the publication of this paper.

REFERENCES

[1] S. Choudhary, S. Pareyani, and S. Kourav, "A Real-Time Local Binary Pattern-based Face Recognition System on Open CV using Boosted Cascade of Simple Features," Proceedings - 2024 13th IEEE International Conference on Communication Systems and Network Technologies, CSNT 2024, pp. 856–861, 2024, doi: 10.1109/CSNT60213.2024.10545885.

[2] M. M. Zahoor et al., "Brain Tumor MRI Classification Using a Novel Deep Residual and Regional CNN," Biomedicines 2024, Vol. 12, No. 7, 1395, Jun. 2024, doi: 10.3390/biomedicines12071395.

[3] C. R. Edwin Selva Rex, J. Annrose, and J. Jenifer Jose, "Comparative analysis of deep convolution neural network models on small scale datasets," Optik (Stuttg), vol. 271, p. 170238, Dec. 2022, doi: 10.1016/J.IJLEO.2022.170238.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 580–587, Sep. 2014, doi: 10.1109/cvpr.2014.81.

[5] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness." Accessed: Jul. 24, 2025. [Online]. Available: https://github.com/rgeirhos/texture-vs-shape

[6] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, no. 07, pp. 12993–13000, Apr. 2020, doi: 10.1609/aaai.v34i07.6999.

[7] K. D. Toennies, "Feature Extraction by Convolutional Neural Network," An Introduction to Image Classification, pp. 169–186, 2024, doi: 10.1007/978-981-99-7882-3_8.

[8] J. Wang, Q. M. Jonathan Wu, and N. Zhang, "You Only Look at Once for Real-Time and Generic Multi-Task," IEEE Trans Veh Technol, vol. 73, no. 9, pp. 12625–12637, 2024, doi: 10.1109/tvt.2024.3394350.

[9] T. Lindeberg, "Scale Selection," Computer Vision, pp. 1–14, 2021, doi: 10.1007/978-3-030-03243-2_242-1.

[10] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2017-January, pp. 936–944, Jul. 2017, doi: 10.1109/cvpr.2017.106.

[11] J. Zhao, "Sports Motion Feature Extraction and Recognition Based on a Modified Histogram of Oriented Gradients with Speeded Up Robust Features," J Comput (Taipei), vol. 33, no. 1, pp. 63–70, 2022, doi: 10.53106/199115992022023301007.

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Trans Pattern Anal Mach Intell, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/tpami.2016.2577031.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN.," IEEE Trans Pattern Anal Mach Intell, vol. 42, no. 2, pp. 386–397, Jun. 2018, doi: 10.1109/tpami.2018.2844175.

[14] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, vol. 2017-January, pp. 6517–6525, Nov. 2017, doi: 10.1109/cvpr.2017.690.

[15] M. Fu et al., "Image Stitching Techniques Applied to Plane or 3-D Models: A Review," IEEE Sens J, vol. 23, no. 8, pp. 8060–8079, Apr. 2023, doi: 10.1109/jsen.2023.3251661.

[16] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: challenges, architectural successors, datasets and applications," Multimed Tools Appl, vol. 82, no. 6, pp. 9243–9275, Mar. 2023, doi: 10.1007/s11042-022-13644-y/tables/7.

[17] Y. Amit, P. Felzenszwalb, and R. Girshick, "Object Detection," Computer Vision, pp. 875–883, 2021, doi: 10.1007/978-3-030-63416-2_660.

[18] M. Bansal, M. Kumar, and M. Kumar, "2D object recognition: a comparative analysis of SIFT, SURF and ORB feature descriptors," Multimed Tools Appl, vol. 80, no. 12, pp. 18839–18857, May 2021, doi: 10.1007/s11042-021-10646-0/tables/10.

[19] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for Object Detection," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3500–3509, Oct. 2021, doi: 10.1109/iccv48922.2021.00350.

[20] R. Gavrilescu, C. Zet, C. Fosalau, M. Skoczylas, and D. Cotovanu, "Faster R-CNN:an Approach to Real-Time Object Detection," EPE 2018 - Proceedings of the 2018 10th International Conference and Expositions on Electrical And Power Engineering, pp. 165–168, Dec. 2018, doi: 10.1109/icepe.2018.8559776.

[21] "Comparative Study of Some Deep Learning Object Detection Algorithms: R-CNN, FAST R-CNN, FASTER R-CNN, SSD, and YOLO | Nile Journal of Engineering and Applied Science." Accessed: Jul. 24, 2025. [Online]. Available: https://www.nilejeas.com/index.php?mno=150264

[22] P. Adarsh, P. Rathi, and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," 2020 6th International Conference on Advanced Computing and Communication Systems, ICACCS 2020, pp. 687–694, Mar. 2020, doi: 10.1109/icaccs48705.2020.9074315.

[23] K. Li and L. Cao, "A review of object detection techniques," Proceedings - 2020 5th International Conference on Electromechanical Control Technology and Transportation, ICECTT 2020, pp. 385–390, May 2020, doi: 10.1109/icectt50890.2020.00091.

[24] N. O'Mahony et al., "Deep Learning vs. Traditional Computer Vision," Advances in Intelligent Systems and Computing, vol. 943, pp. 128–144, 2020, doi: 10.1007/978-3-030-17795-9_10.

[25] G. Neela, K. Babu, and V. J. Peter, "SKIN CANCER DETECTION USING SUPPORT VECTOR MACHINE WITH HISTOGRAM OF ORIENTED GRADIENTS FEATURES," ICTACT JOURNAL ON SOFT COMPUTING, p. 2, 2021, doi: 10.21917/ijsc.2021.0329.

[26] D. K. Larasati, I. Setyawan, and A. A. Febrianto, "Automatic Stop Line Violations Detection using Histogram of Oriented Gradients and Support Vector Machine," ICOIACT 2022 - 5th International Conference on Information and Communications Technology: A New Way to Make AI Useful for Everyone in the New Normal Era, Proceeding, pp. 361–366, 2022, doi: 10.1109/icoiact55506.2022.9972237.

[27] Z. Alshingiti, R. Alaqel, J. Al-Muhtadi, Q. E. U. Haq, K. Saleem, and M. H. Faheem, "A Deep Learning-Based Phishing Detection System Using CNN, LSTM, and LSTM-CNN," Electronics 2023, Vol. 12, Page 232, vol. 12, no. 1, p. 232, Jan. 2023, doi: 10.3390/electronics12010232.

[28] C. Cao et al., "An Improved Faster R-CNN for Small Object Detection," IEEE Access, vol. 7, pp. 106838–106846, 2019, doi: 10.1109/access.2019.2932731.

[29] Y. Su, D. Li, and X. Chen, "Lung Nodule Detection based on Faster R-CNN Framework," Comput Methods Programs Biomed, vol. 200, p. 105866, Mar. 2021, doi: 10.1016/j.cmpb.2020.105866.

[30] G. Ahmed Salman, A. Ghalib Ahmed, and H. Moaiad Hussen, "Medical Image Compression Utilizing The Serial Differences and Coding Techniques," InfoTech Spectrum: Iraqi Journal of Data Science , vol. 2, no. 1, pp. 26–36, Jan. 2025, doi: 10.51173/ijds.v2i1.13.

[31] D. Yi, J. Su, and W. H. Chen, "Probabilistic faster R-CNN with stochastic region proposing: Towards object detection and recognition in remote sensing imagery," Neurocomputing, vol. 459, pp. 290–301, Oct. 2021, doi: 10.1016/j.neucom.2021.06.072.

[32] H. Xiong, J. Wu, Q. Liu, and Y. Cai, "Research on abnormal object detection in specific region based on Mask R-CNN," Int J Adv Robot Syst, vol. 17, no. 3, May 2020, doi: 10.1177/1729881420925287.

[33] T. Bai et al., "An Optimized Faster R-CNN Method Based on DRNet and RoI Align for Building Detection in Remote Sensing Images," Remote Sensing 2020, Vol. 12, Page 762, vol. 12, no. 5, p. 762, Feb. 2020, doi: 10.3390/rs12050762.

[34] Z. Dahirou and M. Zheng, "Motion Detection and Object Detection: Yolo (You only Look Once)," Proceedings - 2021 7th Annual International Conference on Network and Information Systems for Computers, ICNISC 2021, pp. 250–257, 2021, doi: 10.1109/icnisc54316.2021.00053.

[35] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," Procedia Comput Sci, vol. 199, pp. 1066–1073, Jan. 2022, doi: 10.1016/j.procs.2022.01.135.

[36] N. Sasikala, V. Swathipriya, M. Ashwini, V. Preethi, A. Pranavi, and M. Ranjith, "Feature Extraction of Real-Time Image Using SIFT Algorithm," European Journal of Electrical Engineering and Computer Science, vol. 4, no. 3, May 2020, doi: 10.24018/ejece.2020.4.3.206.

[37] S. Cho et al., "Dog Noseprint Identification Algorithm," International Conference on Information Networking, vol. 2021-January, pp. 798–800, Jan. 2021, doi: 10.1109/icoin50884.2021.9333973.

[38] Z. Hossein-Nejad, H. Agahi, and A. Mahmoodzadeh, "Image matching based on the adaptive redundant keypoint elimination method in the SIFT algorithm," Pattern Analysis and Applications, vol. 24, no. 2, pp. 669–683, May 2021, doi: 10.1007/s10044-020-00938-w/tables/6.

[39] W. Burger and M. J. Burge, "Scale-Invariant Feature Transform (SIFT)," pp. 709–763, 2022, doi: 10.1007/978-3-031-05744-1_25.