TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

A Hybrid Faster R-CNN and YOLOv5 Model with Transformer Augmentation for Enhanced Object Detection

Kunal Sahu[a], Khushi Rajput[b], Shweta Sinha[c] and Rinku Raheja[d]

[a,b] Scholar, Department of Computer Science, National P.G. College, Lucknow, India
[c,d] Assistant Professor, Department of Computer Science, National P.G. College, Lucknow, India
ku9alsahu@gmail.com[a],khushirajput13072007@gmail.com[b],shwetasinha.nationalpgcollege@npgc.in[c], rr_141085@yahoo.co.in[d],

KEYWORDS

Object Detection; YOLOv5; Faster R-CNN; Transformer decoder; Hybrid Model;

ABSTRACT

Our proposal includes a three-step model to identify small-scale objects less than 32x32 pixels, e.g., backpacks, handbags, or other discarded items in a security camera image. We initially determine potential boxes with YOLOv5. Then we fine-tune those boxes with Faster R-CNN to achieve more precise results. We now include a small Transformer decoder to detect smaller objects. We will prune the model using weight pruning and INT8 quantization, and will make the size of the model smaller by 20-30%, and targeting 20-30 frames per second on a Jetson Nano to make it executable in real time. Our training will be done on mixed precise on a custom surveillance set which concentrates on small things. We are aiming to make the recall of small objects exceed the 30% baseline by YOLOv5 with obvious benefits in autonomous car, smart security, and farm monitoring applications. The model will subsequently be tested on our set by running the model later, testing it on COCO and KITTI, and testing its ability to work with video streams.

1. Introduction to Object Detection

Object detection refers to a process of detection that involves the detection of objects in an image or a video based on the use of bounding boxes or a rectangle [1]. By the COCO dataset standard, small objects are objects that have their area less than 32x32 pixels [16]. One of the fundamental issues of computer vision is the object detection. Its application can also be found in tracking car vehicles, radiology, detecting crop diseases and manipulation of real-world objects with robots. In this detection process, it is started with preprocessing wherein the image is resized, normalized or converted to grayscale where necessary. Once this has been accomplished, feature extraction follows, which could be edge detection, Histogram of Oriented Gradients (HOG) or deep learning-based CNNs. In the second stage, the region proposal tools define possible location of objects, which are grouped according to the category, which could be cars, people or animals. Regression is used to refine the bounding box values to increase accuracy. Post-processing detectors, such as Non-Maximum Suppression (NMS) are used to eliminate repeated or low-confidence detections. Faster R-CNN, YOLO, and SSD are deep learning frameworks that have greatly enhanced object detection because it is quicker and more precise and can be implemented in autonomous vehicles, security, and medical imaging [1], [2].

The process of object detection has been enhanced by the use of handpicked features to deep learning. Veteran techniques such as Haar Cascades and HOG + SVM employed fixed-rule but were slow and not competitive. Deep learning introduced R-CNN models [4], where objects are found using neural networks. However, they were very

Corresponding Author: Kunal Sahu, Department of Computer Science, National P.G. College, Lucknow, India
Email: ku9alsahu@gmail.com

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

demanding in the sense of computing power. Then, YOLO [7] and SSD [8] accelerated detection to be used in real-time. Recent models such as DETR and Vision Transformers find objects more efficiently without additional mechanisms with the help of attention [3], [14]. Such advances have ensured that object detection has found application in numerous sectors such as security, medical and self-driving vehicles [12].

1.1. Applications of Object Detection

The Object Detection has revolutionized the industries since the machines are able to interpret the data that is captured in form of images or videos. It is an important feature of computer vision. There are numerous applications of this technology in the different industries. The most important applications of Object detection include:

- **Intelligent Surveillance and Security**: Deep learning in these systems is used to recognize faces, license plates, and human activities using camera. These algorithms assist in the detection of suspicious activities, thefts and other tasks that are done through searching and identifying objects [19].
- **Healthcare and Medical Imaging**: Object detection can be used to improve healthcare by making health diagnosis, medical imaging and monitoring patients easier. It identifies the presence of tumour, fractures, early disease diagnosis
- **Farming and Crop Surveillance**: Object detection assists in the detection of diseases in crops, weather detection and finding the soil condition. It also detects growth of plants and animals in the livestock.
- **Gaming and Augmented Reality**: CNN-based object detection is significant in the interaction of the user with the virtual world in the gaming and augmented reality (AR). These systems are based on the vision models to recognize hands, face and full body motions.
- **Performance Tracking and Sports Analytics**: CNN-based algorithm is applied in sports both in performing analysis and in decision-making in real-time. The Hawk-Eye is based on the rapid and precise detection models of objects such as YOLOv4 or SSD. They monitor high-speed balls in a series of frames with a high degree of accuracy and low latency.
- **Robotics and Automation**: Object detection via CNN is a fundamental element in robotics because it makes the machines see and respond to the environment around them. Such models as Mask R-CNN [11], Faster R-CNN [6], and EfficientDet are commonly applied to solve such tasks as item recognition and picking objects. Object detection helps robots in manufacturing industries to put together and pack products.

1.2. Challenges in Object Detection

Detecting partially hidden objects, seeing objects under dark conditions, and distinguishing objects of similar appearances are some of the problems faced by object detection systems. Moreover, the problems such as real-time processing requirements, and biasing of datasets also make sure detection difficult. We discuss below the major issues which affect CNN-based object detectors.

- **Computational Cost**: Object detection makes use of sliding windows that move across an image that adds to the amount of computation required. Even optimized processes such as R-CNN consume large amounts of resources to process every region in an image.
- **Occlusion**: The absence of key parts of object or the concealment of the same confuses CNNs as they might be trained to observe the entire object. In comparison to human beings, CNNs do not guess easily half-seen objects within a picture or real-time.
- **Class imbalance**: As a rule, CNNs tend to work well in case the objects are very visible and fail to perform under unfocused parts of the objects. Class imbalance arises when CNN is trained on high-frequency occurring objects, such as pedestrians, and is so biased against the common objects that they are thus not noticed. Besides, the background in most pictures is majority of an image hence model becomes more accurate in labeling nothing but the object themselves which may result in detection inefficiency.
- **Overlapping Objects**: In a small area where objects are stacked together with a lot of spacing objects can be hidden under the other resulting in the objects being hard to differentiate with. In case the shapes or textures of objects get mixed with one another, it is possible that the model does not provide separate bounding boxes or can fuse multiple instances in a single instance. This is especially an issue in congested

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

areas like crowds, road scenes or shelves in shopping stores. It is especially relevant to the applications such as self-driving cars or drone surveillance where mistakes can be disastrous.

- **Data limitations**: CNNs are also expensive to be trained and need large amounts of labelled data which is not readily accessible. The time and cost involved in developing these datasets is rather time consuming and costly especially in areas that might need expertise. Models in fields where there is limited data are likely to work well in training. examples but to be poor in that of new or unseen data.

2. Research Objectives

This paper introduces a hybrid architecture combining YOLOv5, Faster R-CNN, and a Transformer decoder to improve small-object detection, specifically for surveillance applications. The project is structured around the following key objectives, which outline the necessary stages for development, optimization, and evaluation to meet practical requirements.

- **Design a hybrid system for small-object detection**: A three-stage detection pipeline will be developed. In this system, YOLOv5 generates rapid initial bounding-box predictions, Faster R-CNN refines these for more accurate localization, and a Transformer decoder with four attention heads enhances the overall performance on small objects--defined as those under 32x32 pixels--in complex environments like surveillance footage or autonomous driving scenarios.
- **Optimize for edge device deployment**: The model will be optimized using weight pruning and INT8 quantization. The goal is to reduce the model's size by 20-30% while achieving a processing speed of 20-30 frames per second (FPS) on a Jetson Nano. This ensures the system can run efficiently on resource-constrained hardware such as security cameras and unmanned aerial vehicles (UAVs).
- **Train on a specialized surveillance dataset**: The model will be trained using mixed-precision on a custom dataset containing persons and small artifacts like backpacks. Data augmentation techniques, including zoom and flip, will be employed to increase the model's robustness and adaptability to diverse real-world conditions.
- **Improve small-object recall**: A primary goal is to exceed the 30% baseline recall for small objects (smaller than 32x32 pixels) set by YOLOv5. The focus is on reliably identifying difficult-to-detect objects such as distant pedestrians or traffic signs in crowded or cluttered settings.
- **Conduct testing and outline future work:** The hybrid model's performance will be evaluated against standalone YOLOv5 and Faster R-CNN models using metrics like mean Average Precision (mAP) and recall on both custom and standard datasets (e.g., COCO [16], KITTI [17]). Furthermore, the integration of the model with live video streams will be explored to pave the way for future real-world applications.

3. Related Work

3.1. YOLOv5

The YOLO (You Only Look Once) family of detectors has gained a massive popularity as one of the leading real time object detectors. Its defining feature is that it uses a single pass over the image, thus localization is performed in a single forward pass, and hence does not require the bulky region proposal step that afflicts two-step detectors like R-CNN [4] or Faster R-CNN [6]. YOLOv5 is designed to be highly inferred, with a CSPDarknet53 backbone that is a compromise between speed and accuracy [10]. It usually reaches 30-50 frames per second, with a mean average precision of about 45.4% on the COCO val2017 dataset [9]. The hierarchical breakdown of the image offered by the backbone allows the image to be processed very quickly, which makes the model suitable to be deployed in real time, e.g. drone surveillance or a security camera feed. However, it is the performance of YOLOv5 on smaller objects which are considered by the COCO standard as less than 32x32 pixels that is weak because it finds about 30% of such objects [10]. The working of YOLOv5 is as follows:

1) The processing of the input image is first performed by CSPDarknet53 backbone of YOLOv5, which provides very detailed representations as features.

2) These resultant feature maps are then divided in an NxN grid. The duty of each single cell in this grid is to identify any object in which the geometric center is located within the geometrical boundaries of the cell.

3) The YOLOv5 detection head within a grid cell provides a predetermined number of bounding-box predictions, B. Each box of prediction is emitted by the model with the spatial coordinates (x, y, w, h) and a scalar

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

of confidence which indicates the probability of an object to be there and a vector of class-probability estimates in the C possible categories. All these predictions are obtained in one forward pass through the network.

4) The algorithm uses Non-Maximum Suppression (NMS) to eliminate unnecessary hypotheses. This is done to eliminate overlapping bounding boxes and only the boxes with best confidence scores are retained, to give a concise set of detections.

5) The remaining bounding boxes are then ultimately projected back over to the coordinate system of the original input image. The output that results is the following boxes with their respective class labels and confidence values.

In our system, YOLOv5 is the first proposal generator because it is incredibly fast and does not require much time to create a rough proposal of bounding boxes. Being a single-stage detector, it provides the initial stage of localization of the object to the subsequent modules, hence ensuring a high frame rate which is important in real-time application. Its grid-based scheme and reduced recall on fine objects, approximately 30% recall of objects smaller than 32x32 pixels, however, require additional refinements steps. As a result, the output of the initial boxes in YOLOv5 is fed into other modules that optimize and expand the propositions hence maintaining the speed of the entire hybrid system but improving the quality of detection. We have chosen YOLOv5 due to speed with which it produces first hypotheses which is essential in applications like autonomous vehicle perception. The low computational requirements of YOLOv5 make it suitable to edge devices that will be the focus of our application, and the responsiveness of the model is useful when it comes to completing rapid prototyping on datasets like KITTI [17].

3.2. Faster R-CNN

Faster R-CNN replaces the slow selective search of R-CNN and Fast R-CNN with a Regional Proposal Network (RPN) making it even faster [5], [6]. RPN is a small neural network that slides over the feature map and determines how likely an object exists there. It is almost 10 times faster than Fast R-CNN [5]. It uses Anchor boxes which predefined bounding boxes that serve as references for predicting where object is located. It is a two-stage detector. Faster R-CNN is great at making sharp object boxes. It runs a ResNet-50 backbone with a Region Proposal Network to hit about 38% mAP@0.5:0.95 on COCO val2017 [12]. It scans images to pick out regions then fine-tunes boxes and guesses object types. It's great for precision but slow at 5-10 FPS. This makes it tough for real-time tasks like self-driving cars. Earlier models like R-CNN were way slower and less accurate [4]. Faster R-CNN's strength is nailing tough objects like small signs in KITTI's driving scenes [12]. It handles complex images well but needs a speed boost to fit our setup. The pre-trained weights make it easy to use in our model. The working of Faster R-CNN is as follows:

1) The complete image gets processed by the base CNN which is typically VGG-16 or ResNet.

2) A detailed feature map is created from the image which serves as shared foundation for both regional proposal generation and object detection.

3) The RPN slides over the feature map to evaluate multiple anchor points at each specific position. Probability of containing an object is predicted for each anchor point.

4) After selecting top region proposals, the system applies ROI Pooling to extract fixed size feature maps.

5) The system makes its final determination by passed these pooled features through fully connected layers which classify objects and adjust the box coordinates for optimal alignment with the actual object.

To address the localization inaccuracies inherent in the first-stage proposals, we integrate Faster R-CNN as a second-stage refiner. Its two-stage architecture, featuring a Region Proposal Network (RPN) and dedicated box-regression heads, is renowned for high detection accuracy and precise bounding box regression [6]. While traditionally computationally expensive (5-10 FPS), its workload is significantly reduced in our framework by operating on the pre-filtered proposals from YOLOv5. This synergistic combination allows us to capitalize on Faster R-CNN's superior accuracy for bounding box refinement without incurring its full computational cost, thereby enhancing the precision of our detections, particularly for overlapping and complex objects. Such hybrid approaches have proven effective in real-world surveillance tasks [18], [19]. We choose Faster R-CNN to fix YOLOv5's rough guesses. Its high accuracy is key for spotting small stuff like pedestrians or signs in COCO [16] and KITTI [17]. The RPN makes boxes very precise which is perfect for our hybrid model's goal. But its slow speed isn't good for drones or cars. That's why we use YOLOv5's fast boxes to lighten its load. The ResNet-50 backbone pulls out details from busy scenes. This helps our model work better in real-world tasks like surveillance [13]. The pre-trained

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

weights let us plug it in without starting from scratch. This precision is crucial for catching tiny objects in tough settings like city roads.

3.3. Transformer Decoder

Transformer decoders are great for seeing the whole picture. They use attention to focus on all parts of an image at once [14]. This helps spot tiny objects that other models miss. They're newer in object detection but work well with CNNs like YOLOv5. Our model uses a light version with 4 heads and 2 layers. It's built to catch small objects under 32x32 pixels in COCO [16] or KITTI [17] scenes. Older models like YOLOv5 struggle with global context so Transformers help a lot. The final component of our architecture is a lightweight Transformer decoder, which we introduce to capture global contextual information that CNNs often miss. Convolutional networks are inherently local, which can limit their ability to detect small objects that rely on broader scene context. The multi-head self-attention mechanism within the Transformer allows the model to weigh features across the entire image, effectively aggregating dispersed clues that signify the presence of a small object. We employ a compact design with four attention heads and two layers to minimize computational overhead. This module is tasked specifically with boosting the recall of sub-32x32 pixel objects by re-scoring and fine-tuning the refined proposals from the previous stage based on global feature relationships.

The Transformer decoder is a key choice for our hybrid model, enhancing small-object detection with its multi-head self-attention, which captures global image context for items like backpacks, discarded packages, or traffic signs [14]. Its lightweight design with four heads and two layers ensures minimal overhead, supporting real-time needs. Pre-trained on ViT [20], it adapts quickly to our dataset, complementing YOLOv5 and Faster R-CNN by refining tiny targets. Its low compute needs enable Jetson Nano deployment, boosting recall for surveillance tasks [15].

3.4. Optimization Techniques

The methods of optimization like weight pruning and INT8 quantization are used to optimize the model size without affecting the performance of inferencing [15]. Such approaches are inevitable to implement the models in low-powered systems like the Jetson Nano. The resulting models can run at real-time inference rates of 20-30 FPS using fixed-point arithmetic, which then removes redundant parameters and allows high-performance applications like autonomous driving and surveillance. As such, this has made our detection system practically viable and scalable in any context of operation.

To be able to run our applications in real-time on resource-constrained edge platforms, like the Jetson Nano, we use are a set of post-training optimization steps. Weight pruning eliminates redundant parameters selectively which reduces the size of the model as well as the inference latency. In the postprocessing step, the INT8 quantization is applied to reduce the network size and speed up calculation with fixed-point arithmetic [15]. The measures are mandatory to achieving our desired 20-30 FPS goal to ensure the practical feasibility of the hybrid architecture with the cost of just non-negligible loss to accuracy.

Optimization techniques play a critical role in the implementation of our model to edge devices, and pruning, and INT8 quantization result in a reduction of model size up to 30% on the Jetson Nano [15]. Such strategies maintain a 20-30 FPS throughput which is able to match the high accuracy of Transformer-based detectors with the speed constraints of real-time applications, e.g. autonomous driving. They enable scalable training on both COCO [16] and KITTI [17] datasets by reducing computational needs, enabling effective deployment in aerial robotics among other uses, and their scalability will enable future experiments on the databases. Based on RT-DETR, such techniques ensure practicality as well as high effectiveness in object recognition processes.

4. Proposed Methodology

4.1. Model Architecture

The hybrid model consists of three stages which takes advantage of the strengths of YOLOv5, Faster R-CNN, and Transformer components to address limitations in classic hybrids. The complete architectural flow is illustrated in Fig.1.

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

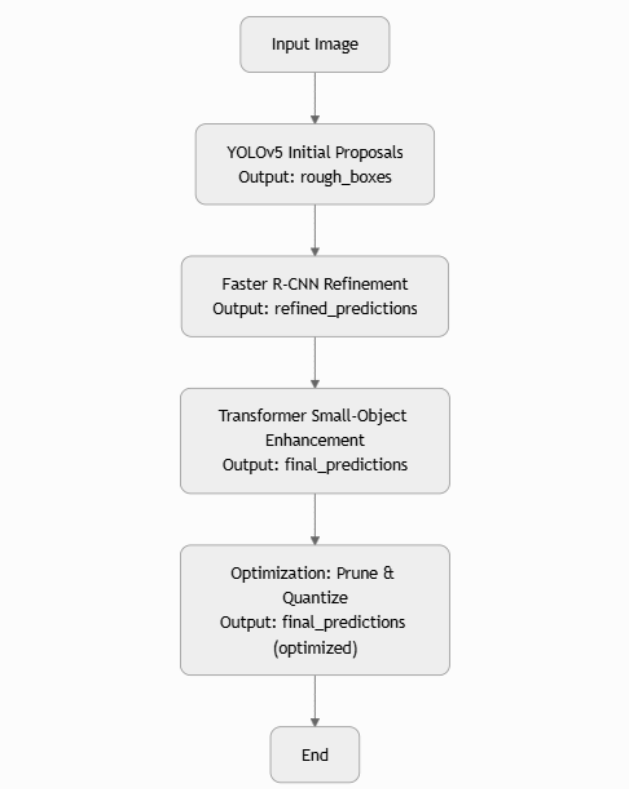

Figure 1: Process flow of the hybrid detection model, illustrating the sequential pipeline from YOLOv5 proposal generation to final optimization.

The stages are specifically selected to overcome the deficiencies of the previous ones forming a synergistic pipeline as follows:

- **YOLOv5 for Initial Object Detection**: The medium variant of YOLOv5 with a CSPDarknet53 backbone is selected. It achieves ~45% mAP on COCO val2017 for the medium variant [9]. This stage conducts fast, grid-based detection to generate initial bounding box proposals and class probabilities. It supports real-time processing. Coarse bounding boxes with confidence scores are produced. These are sent to the next stages for refinement. YOLOv5's low latency suits real-time needs. Its limited small-object detection requires further enhancement.
- **Faster R-CNN for Enhanced Bounding Box Precision**: Faster R-CNN employs a ResNet-50 backbone and a Region Proposal Network (RPN) for high accuracy. It achieves ~37–42% mAP on COCO val2017 with ResNet-50 [12]. This stage refines YOLOv5's initial boxes using the RPN. It re-evaluates feature maps for accurate bounding box regression and classification. Improved bounding boxes with better localization are generated. This enhances the quality of initial proposals. Faster R-CNN's accuracy is strong but slow (5–10 FPS). Using YOLOv5's outputs reduce the RPN's workload, improving efficiency.
- **Transformer Decoder for Small-Object Detection**: A lightweight Transformer decoder with 4 attention heads and 2 layers is designed, inspired by hybrid architectures [14]. This stage refines Faster R-CNN's boxes with multi-head self-attention. It captures global context to improve small-object detection under 32x32 pixels. Final bounding boxes and class predictions are produced. Accuracy for small objects is enhanced. The decoder addresses CNN limitations in global context. It aims to boost small-object recall beyond YOLOv5's 30% baseline on the custom dataset.
- **Optimization Strategies**: Weight pruning and quantization, adapted from efficient models, are applied. This maintains competitive accuracy (e.g., within 2–3% of baseline mAP) on COCO after pruning and quantization [15]. The approach enables deployment on low-resource edge devices like Jetson Nano. Optimized performance supports 20–30 FPS while maintaining accuracy.

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

4.2. Algorithm

The input image is taken as I and the result of this algorithm are bounding boxes B. Following steps are taken to obtain the final bounding boxes:

Step 1: In the first phase, one creates feature maps from the input image I by using the YOLOv5 CSPDarknet53 backbone.

Step 2: After that, the YOLOv5 head generates the coarse bounding boxes as well as their respective class labels and confidence scores.

Step 3: The coarse boxes are then Non-Maximum Suppression using Intersection-Over-Union (IoU) cutoff of 0.5 and confidence cutoff of 0.3 is applied to the coarse boxes to remove the redundant detections.

Step 4: The result of the filtering of the coarse bounding boxes is then sent on to the next step.

Step 5: A Faster R-CNN framework is initialized in the system, which uses a ResNet-50 backbone along with a Region Proposal Network (RPN) which is pre-trained on the COCO dataset.

Step 6: The ResNet-50 architecture is then used to process the image I to compute feature maps.

Step 7: Initial proposals are improved using the RPN on the feature maps along with the previously obtained coarse bounding boxes.

Step 8: Faster R-CNN head then predicts the fine tuned bounding boxes using the pooled regional features and then Non-Maximum Suppression (IoU and confidence threshold 0.5) is applied again.

Step 9: The predictions are refined and are output to the next module.

Step 10: Pipeline continues by setting up a Transformer decoder configuration, which consists of 4 attention heads and 2 layers, with parameters pre-trained on a Vision Transformer (ViT).

Step 11: The refined predictions, this Transformer decoder, where the multi-head self-attention mechanisms are used to calculate contextually-aware token representations.

Step 12: A final linear layer is used to map these representations of tokens to definitive bounding box coordinates and a stringent Non-Maximum Suppression step (IoU of 0.5, confidence threshold of 0.6) is used to remove overlaps.

Step 13: The output of this transformer based refinement is the final predictions.

Step 14: Post training optimizations are done that involve pruning the redundant model weights and quantizing the rest of the parameters to 8-bit integer (INT8) format to improve inference speed.

Step 15: Finally, the model outputs the model optimized predictions

4.3. Training

- **Procedure**: In order to balance speed and accuracy, the training program will make use of mixed-precision computation. This plan makes use of 16-bit floating operations that are lightweight on most layers but maintain full 32-bit accuracy in the important parts where analytical accuracy is of paramount importance. The goal of such a design is to achieve high computational efficiency at the expense of detection fidelity, which is required when edge deployments have limited processing resources.
- **Dataset**: The training data will be a custom corpus which is curated by surveillance images. The collection contains human subject scenes and small objects like handbags, backpacks, or other personal items, most of which have been rendered at resolutions lower than 32x32 pixels. The dataset spans over a variety of environments and a variety of lighting conditions, and it is designed in a manner that fosters robustness in practice. Particle focus has been on improving the recall of the transformer module on small objects- a huddle in traditional detection models.
- **Loss Function**: The constituents of the hybrid architecture will use customized optimization methods. YOLOv5 will use the Complete IoU loss to make localization predictions in the early stages; Faster R-CNN will combine cross-entropy with Smooth-L1 to make predictions in the late stages; and the transformer unit will add an auxiliary loss that is specifically aimed at enhancing the recognizability of small-scale targets. All these methodologies are expected to stabilise the learning dynamics and enhance the overall accuracy.

4.4. Implementation

- **Framework**: The framework will be implemented in the PyTorch ecosystem, as it is a modular framework with an active community. Repurposed pre-existing weights of YOLOv5 and Faster R-CNN will be obtained to speed up the initialisation step by the authoritative repositories. By utilizing these baselines, it is

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

straightforward to attain a smooth integration of the fast region proposal generation of YOLOv5 with the accurate analysis of Faster R-CNN without the necessity of training all the layers.

- **Transformer Integration**: A Vision Transformer decoder will be integrated to enhance the system to be more focused on tiny objects. Using the multi-head attention, the decoder will view the whole picture and compress minute details that would have gone unnoticed by traditional convolution-based detectors. The decoder will be tailored to the conditions that include human occupants, personal objects, and other small objects which are frequently seen in the video footage of the surveillance.
- **Hardware**: The entire experiment will be performed on an NVIDIA RTX 3080 graphic card, which will be able to support the high training loads. A model will be deployed and tested on Jetson Nano edge devices once convergence is reached so as to determine real-time performance with limited power budgets. The aim is to maintain a constant frame rate of about 20-30 FPS, therefore, pointing out the appropriateness of use in live surveillance [22].
- **Pre-training**: The modules in the framework will start with pre-trained weights based on the custom surveillance data. The transformer decoder will be further refined to be able to better identify smaller targets like distant pedestrians, backpacks, or tools. This measure should simplify the transition to downstream applications, which are automated monitoring systems and driver-assistance systems [21].

5. Expected Outcome

The hybrid model suggested should be able to perform better than YOLOv5 and YOLO-Faster R-CNN hybrids due to lightweight Transformer decoder and optimisation methods. The accuracy of detection can be improved, and it may have a high precision, recall, and F1-score, and the IoU can be higher, therefore, signifying a small improvement compared to the benchmark on a custom surveillance dataset centered around persons and small items. The Transformer decoder has the potential to improve the recall of small-objects, with the goal of better detection of objects smaller than 32x32 pixels, which can improve the miss rates in deploying the system to tasks like traffic sign and pedestrian detection. We want our hybrid configuration to reach 20-30 FPS on the Jetson Nano, reaching higher accuracy on small objects and with high-endurance real-time capabilities in areas like self-driving or surveillance. The size of the model can be minimised by optimisation, and possibly maintain accuracy on edge devices. The model might hence be applicable in the autonomous driving, intelligent surveillance, and agricultural surveillance application, and may play a role in enhancing the detection of small objects.

Conclusion and Future Scope

This proposal outlines a three-phase hybrid design that is meant to enhance object detection of small objects that are less than 32x32 pixels. The framework uses YOLOv5 to make initial bounding-box hypotheses, which are subsequently refined by Faster R-CNN to attain a higher precision and finally, it uses the Transformer decoder to add global contextual information to the proposals, which enhances recalling small targets. Through this, the method will aim at addressing existing limitations in applications like traffic-sign recognition and identification of pedestrians. Pruning and quantisation complementary methods are harnessed to make the system real-time executable on the limited edge devices like the NVIDIA Jetson Nano, which is a viable approach to areas such as autonomous driving, smart surveillance, and high-precision agriculture. The direction of future work will focus on testing the hypothesised advantages by undertaking a systematic test on a customised surveillance data set; future studies can be directed at iterative pruning schemes to further expedite the inference at the expense of fidelity. In order to challenge the portability of the framework, we suggest to multi-dataset-benchmark it with well-known datasets like COCO [16] and KITTI [17], as well as to be integrated into live video pipelines to evaluate dynamic performance. Furthermore, with the evolution of the realm of real-time vision, potential improvements can be made by replacing the existing YOLOv5 component with more recent designs of the state-of-the-art detectors- e.g., YOLOv8 or YOLOv10 [9]- in such a way that the architectural design will be able to utilise the state-of-the-art design principles and, accordingly, achieve an even higher degree of efficacy in the localisation of small objects without losing the benefits associated with the multi-stage refinement pipeline.

TEJAS Journal of Technologies and Humanitarian Science
ISSN-2583-5599
Vol.04, I.04 (2025)
https://www.tejasjournals.com/
https://doi.org/10.63920/tjths.44009

References

[1] Z. Zou, Z. Shi, Y. Guo, and J. Ye, “Object detection in 20 years: A survey,” Proc. IEEE, vol. 111, no. 3, pp. 257–276, Mar. 2023.
[2] J. Huang et al., “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Vancouver, Canada, Jun. 2023, pp. 7464–7475.
[3] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable DETR: Deformable transformers for end to-end object detection,” in Proc. Int. Conf. Learn. Represent., Virtual, May 2021.
[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Columbus, OH, USA, Jun. 2014, pp. 580–587.
[5] R. Girshick, “Fast R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis., Santiago, Chile, Dec. 2015, pp. 1440 1448.
[6] 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.
[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Las Vegas, NV, USA, Jun. 2016, pp. 779–788.
[8] W. Liu et al., “SSD: Single shot multibox detector,” in Proc. Eur. Conf. Comput. Vis., Amsterdam, The Netherlands, Oct. 2016, pp. 21–37.
[9] M. Hussain, “YOLOv5, YOLOv8 and YOLOv10: The go-to detectors for real-time vision,” arXiv preprint arXiv:2407.02988, Jul. 2024.
[10] R. Khanam and M. Hussain, “What is YOLOv5: A deep look into the internal features of the popular object detector,” arXiv preprint arXiv:2407.20892, Jul. 2024.
[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in Proc. IEEE Int. Conf. Comput. Vis., Venice, Italy, Oct. 2017, pp. 2980–2988.
[12] Z. Chen, H. Wang, Z. Li, and Q. Yan, “A survey of deep learning-based object detection methods in autonomous driving,” IEEE Trans. Intell. Transp. Syst., vol. 24, no. 11, pp. 12345–12367, Nov. 2023.
[13] M. Hussain, “A comprehensive survey of deep learning techniques for object detection in surveillance systems,” arXiv preprint arXiv:2408.01567, Aug. 2024.
[14] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in Proc. Eur. Conf. Comput. Vis., Glasgow, U.K., Aug. 2020, pp. 213–229.
[15] Y. Li, Y. Wang, Z. Liu, and J. Sun, “Efficient object detection for edge devices: A survey,” IEEE Access, vol. 11, pp. 123456–123478, Oct. 2023.
[16] T.-Y. Lin et al., "Microsoft COCO: Common Objects in Context," in Proc. Eur. Conf. Comput. Vis., Zurich, Switzerland, Sep. 2014, pp. 740-755.
[17] Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Providence, RI, USA, Jun. 2012, pp. 3354-3361.
[18] Aliu, M. A. Mabayoje, and A. E. Onuiri, "Real Time Detection of Hand Carried Weapons for Kidnapping Mitigation in Nigeria: A YOLOv5–Faster R-CNN Hybrid Approach," Int. J. Comput. Appl., vol. 184, no. 14, pp. 14-21, Sep. 2022.
[19] Ayush Kashyap et al., Design and Implementation of an Intelligent Loan Eligibility System Using Machine Learning Techniques, TEJAS Journal of Technologies and Humanitarian Science, ISSN-2583-5599, Vol.04, I.02 (2025), https://doi.org/10.63920/tjths.42002
[20] Esha Srivastava et al., AI-Driven Predictive Analytics with the Help of IoT for Organizational Change Management, TEJAS Journal of Technologies and Humanitarian Science, ISSN : 2583-5599, V. 04, I.03, July-2025, https://doi.org/10.63920/tjths.43001
[21] H. Singh and N. Singh, "Real-time smart surveillance using YOLO-Faster R-CNN hybrid approach," Int. J. Comput. Program. Database Manag., vol. 6, no. 2, pp. 112-120, 2023.
[22] Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Represent., Virtual, May 2021.