# Robust Driver Distraction Recognition via Lightweight Body-Part Association and Object Context on NVIDIA Jetson

Frank Zandamela[1 [0000−0003−2201−1985]], Mamodike Sadiki[1 [0000−0002−4984−2992]], Patrick Malatjie[1 [0009−0003−2035−1115]], Teboho Sekopa[1 [0009−0000−3684−1993]], and Moloko Manthata[1 [0009−0002−5379−7263]]

Council for Scientific and Industrial Research, Defence and Security, Pretoria 0001, South Africa
(fzandamela, msadiki, pmalatjie, tsekopa, mmanthata)@csir.co.za
https://www.csir.co.za/defence-and-security

**Abstract.** Driver distraction remains a significant contributor to road accidents; however, existing deep-learning detectors either sacrifice accuracy for speed on resource-constrained hardware or lose generality when confronted with unseen data. This work presents an end-to-end single-stage model that, in one forward pass, jointly identifies the driver, links the driver's body parts, recognises nearby in-cabin objects, and determines whether the driver is distracted. By embedding spatial and semantic relationships directly into its output vector, the model avoids slow post-processing and significantly reduces false alarms caused by objects that merely appear near the driver. Evaluated on five public datasets and an additional real-world collection that were not used for training, the proposed detector boosts mean F1-score by 0.11 (≈20% relative) over a lightweight baseline while maintaining 39 frames per second on an NVIDIA Jetson Xavier edge device—more than three times faster than a comparable two-stage pipeline. These results demonstrate a driver-distraction detector that simultaneously achieves cross-dataset robustness, real-time performance, and efficient deployment on low-power hardware.

**Keywords:** Distracted driver detection · Single-stage detector · Edge AI · Body–part association · Real-time inference

## 1 Introduction

Accurately identifying driver distraction is still a key challenge, and researchers have proposed various deep-learning methods. However, despite recent advancements, there remains a research gap in developing methods with consistent performances across different datasets not used for training and also achieve real-time inference speed on edge devices. There is research focused on developing lightweight distracted driver detection (DDD); for example, Zhao et al. [5] recently proposed a lightweight method that leverages the power of a convolutional neural network and a vision transformer (CoViT). CoViT is a hybrid

model that interleaves multi-scale dilation and depth-wise separable CNN blocks with a low-complexity attention Transformer, capturing both local and global cues efficiently. Despite having only ≈1.24 M parameters, it achieves 159 FPS on a GTX 3090 and achieves an accuracy of 93–98% on three driver distraction datasets, making it ideal for real-time edge deployment. While Liu et al. [6] introduced FRNet, a feature-reorganisation CNN that obtains 97–99% accuracy while running at 59 FPS on an NXP i.MX 8M edge board. It is worth noting that both CoViT and FRNet were only tested on the datasets they were trained on, leaving their cross-dataset robustness uncertain. In addition, the performance of the CoViT method on an edge device remains unknown.

To address both cross-dataset performance and edge deployment shortcomings in the literature, a recent study proposed a lightweight human activity recognition-based distracted driver detection method (HBAR-DDD) [1]. The lightweight HBAR-DDD approach achieved an average F1-score of 0.45 across four datasets and an average inference speed of 21.97 ms or 46 FPS. However, the approach struggled with false detections and misdetections. This was primarily because in the lightweight HBAR-DDD approach, each object or state of the driver's body parts and each distraction object was represented by one box and one class, see Fig. 1. Therefore, the HBAR-DDD approach had two critical issues: **(1)** the model cannot identify which individual in the image is the driver. Therefore, the model cannot link detected body parts to that specific driver, and **(2)** it may falsely flag distraction whenever a cellphone appears near the driver, even if the driver is not using it.

A simple solution is a two-stage approach that first detects the driver's region of interest (ROI), then run the existing HBAR-DDD approach. While the simple two-stage approach significantly improves the accuracy, the improvement comes at the expense of the processing speed - reducing processing speed by a factor of about 3.6 (from 41 FPS to ≈11.3 FPS) during experiments[1]. The HBAR-DDD approach also fails to resolve contextual ambiguity, as object relationships remain unmodeled.

To eliminate this accuracy-speed trade-off while ensuring cross-dataset robustness, we propose BPJ-HBAR, an end-to-end extension of YOLOv5[2]. BPJ-HBAR jointly encodes the driver, associated body parts, and interacting objects in a unified output vector, with an integrated classification head that determines distraction status in a single forward pass. Inspired by Zhou et al.'s extended object representation [7], our design embeds spatial and semantic relationships directly into the detection tensor, thereby eliminating the need for post-hoc association steps. Fig. 1 shows a comparison of the proposed approach with prior HBAR-DDD and driver ROI methods.

[1] Inference speed measured using a $1280 \times 1280$ image size on a Jetson Xavier NX edge device with an 8 GB GPU capable of 21 TOPS.

[2] No official research paper. Documentation can be found at https://docs.ultralytics.com/yolov5

The contributions of this work are threefold:

- **BPJ-HBAR.** We extend YOLOv5 and YOLOv7 [8] with a unified output vector that simultaneously encodes the driver, associated body parts and in-cabin objects, plus an added head that classifies distraction in one forward pass.
- **Extended object representation for body–part–object context.** Inspired by BPJDet, our design embeds spatial and semantic relations directly in the detection tensor, eliminating post-hoc association while keeping the network lightweight.
- **Real-time, cross-dataset performance.** The model runs at 39 FPS on an NVIDIA Jetson Xavier (≈3.45× faster than a two-stage baseline) and retains state-of-the-art accuracy on three public datasets and a custom dataset not seen during training, demonstrating both edge-readiness and strong generalisation.

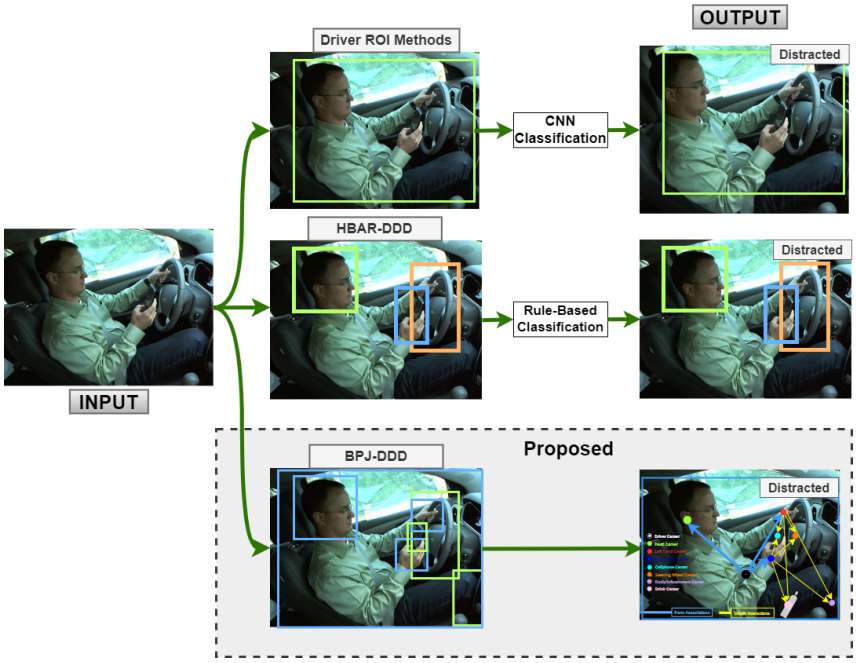


**Fig. 1.** Comparison of the proposed single-stage BPJ-DDD with prior HBAR-DDD [1], and driver ROI methods (e.g., [2–4]). HBAR-DDD and driver-ROI use two-stage pipelines—object detection followed by a separate classification step. In contrast, BPJ-DDD is single-stage: it jointly detects the driver, relevant body parts, and interacting objects and outputs the final distraction class in a single forward pass.

## 2 Related work

Driver distraction remains a leading cause of road-traffic fatalities worldwide. To address this issue, robust distracted driver detection methods that can perform inference in real-time are required. Such methods must run on resource-constrained edge devices (e.g., NVIDIA Jetson series). As a result, this section reviews methods through the lens of the two deployment-critical goals: (i) *robust distracted driver detection via lightweight body-part association and object context*, and (ii) *real-time processing on edge AI devices*.

Early solutions (e.g., [9, 10]) classify the entire image frame with compact CNNs . While simple and often single-stage, these models rely on global appearance and may fail under domain shift. More recent lightweight methods [5, 6, 11] improve speed/accuracy trade-offs, yet their cross-dataset performance remains unknown.

To enhance the robustness of distracted driver detection, researchers have explored the use of object detection to identify specific regions and objects [2–4]. These methods involve two steps: first, the driver ROI is detected, and then classified using a CNN model. For example, the method presented by Sajid et al. [3] utilises EfficientDet [12] to detect distraction objects and the ROI of the driver body parts, and use EfficientNet [13] for classification. The primary disadvantage of two-stage methods is their high computational requirements, which is problematic for embedded GPUs. They also introduce deployment complexity through multi-stage integration.

A recent approach presented by Zandamela et al. [1] (HBAR-DDD) attempted to solve the issue of high computational requirements by using a YOLOv7 object detector to detect driver body parts and classify their states into different activities in one forward pass. The final prediction was obtained by evaluating two conditions: "eyes on the road" and "both hands on the steering wheel." However, this approach struggled with false detections and misdetections because the model cannot identify which individual in the image is the driver. Therefore, the model cannot confidently link detected body parts and objects to the driver.

With the difficulties associated with multi-stage methods and the false detection issues associated with HBAR-DDD, it is critical to utilise lightweight body-part association and object context to detect distracted drivers - a single-stage system that predicts the driver and their parts in one pass (along with relevant objects).

Joint detection-association models from general computer vision offer a solution path. Recent joint detection–association models demonstrate how to bind parts to the correct person in a single pass. PairDETR [14] extends a deformable detection transformer (DETR) with an approximated bipartite matching to jointly detect bodies and faces end-to-end, removing ad-hoc post-processing. PBADet [15] is a one-stage, anchor-free design that uses a single part-to-body centre offset for efficient association. BPJDet augments a one-stage detector with an extended object representation that carries part-offsets, enabling joint body–part prediction without a separate association module.

This work targets robust distracted driver detection by detecting the driver's body and associated body parts. In addition, we associate nearby objects with the driver's hands to provide object spatial context to the model. Furthermore, the HBAR-BPJ approach predicts the state of the driver in a single lightweight pipeline that runs in real-time on edge AI devices. To the best of our knowledge, no such method exists in the literature. Table 1 summarises the key differences between our proposed approach and existing approaches.

**Table 1.** Overview of lightweight DDD approaches and the gaps addressed by this work.

| Approaches | Sources | Single-stage pipeline | Cross-dataset performance tested | Real-time (edge device) | Body-part association |
|---|---|---|---|---|---|
| CNN full-frame | [6, 5, 11] | ✓ | ✗ | ✓ | ✗ |
| Driver ROI detection (two-stage) | [2–4] | ✗ | ✗ | ✗ | ✗ |
| Body parts detection (HBAR-DDD, prior) | [1] | ✗ | ✓ | ✓ | ✗ |
| **Proposed: HBAR-BPJDet (this work)** | | ✓ | ✓ | ✓ | ✓ |

## 3 Proposed method

The BPJ-HBAR framework integrates a YOLOv5 or YOLOv7 backbone with an extended multi-task detection head to jointly detect the driver, link body parts, recognise nearby objects, and classify the driver's global state. During training, feature maps from the backbone feed into the detection head, whose outputs are supervised by a multi-loss function combining bounding box, classification, offset, and state terms. Inference applies non-maximum suppression (NMS) and an association algorithm to yield structured detections that connect drivers with relevant parts and objects, as illustrated in Fig. 2.

We adopt YOLOv5 and YOLOv7 for two pragmatic reasons. First, the reference BPJDet implementation we build upon targets YOLOv5, enabling minimal-change integration of the extended representation. Second, in our prior lightweight HBAR-DDD study, YOLOv7 delivered the best speed–accuracy trade-off across four datasets on a Jetson platform, outperforming more recent YOLO variants under a matched training/evaluation protocol (identical image size and data pipeline). Using both backbones also demonstrates that our head generalises across related one-stage families while isolating our contribution from any single detector.

Our network extends the BPJDet design into a unified multi-task head that jointly predicts three detection streams — *driver bodies*, *body parts*, and *in-cabin objects* — while also regressing two types of association offsets:

– part→body offset $\hat{\mathbf{o}}_{p\to b} \in \mathbb{R}^2$,
– object→hand offset $\hat{\mathbf{o}}_{o\to h} \in \mathbb{R}^2$,

and producing a binary global driver-state logit vector $\mathbf{s}_b \in \mathbb{R}^2$ for each detected driver body, indicating whether the driver is *safe* or *distracted*. Each detection branch outputs classification logits and bounding box parameters $(x, y, w, h)$, following standard one-stage detection practice, but augmented with offset regression and state prediction channels.

*Offset targets.* Let $\mathcal{G}^b, \mathcal{G}^p, \mathcal{G}^o$ be the sets of ground-truth bodies, parts, and objects, each with centre coordinates $\mathbf{c} = (x, y)$ and size $(w, h)$. For a body part $p$ associated with body $b$, the scale-normalised offset target is defined as:

$$\mathbf{t}_{p\to b} = \left( \frac{x_b - x_p}{w_b}, \frac{y_b - y_p}{h_b} \right). \tag{1}$$

For an object $o$ associated with the driver's left or right hand $h$, we define:

$$\mathbf{t}_{o\to h} = \left( \frac{x_h - x_o}{w_h}, \frac{y_h - y_o}{h_h} \right). \tag{2}$$

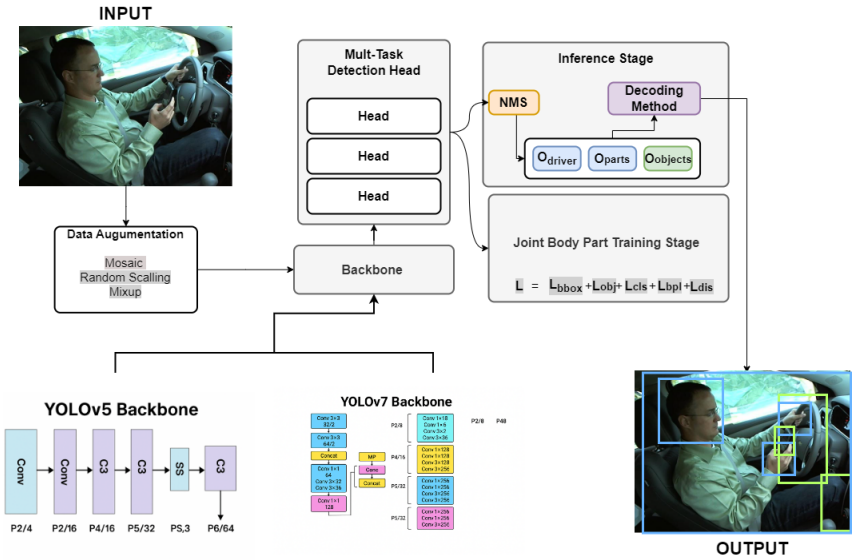


**Fig. 2.** BPJ-HBAR architecture (backbone, head, training/inference).

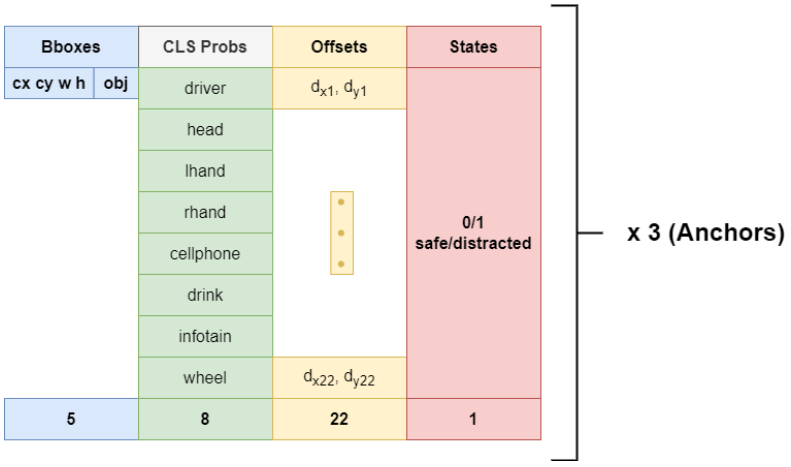


**Fig. 3.** YOLOv5/7 BPJ-HBAR detection head outputs (per grid cell x 3 anchors). Normal YOLO models only have $5 + n_c$ outputs; the BPJ-HBAR approach extends the detection head with offsets and the global driver state class.

*Loss function.* We train with the standard YOLO-style bounding box regression ($L_{\text{box}}$), objectness ($L_{\text{obj}}$), and classification ($L_{\text{cls}}$) losses across all detection streams, extended with: (i) a body–part offset regression loss ($L_{\text{bpl}}$), and (ii) a binary global driver-state classification loss ($L_{\text{dis}}$), where the driver state is *safe* or *distracted*:

$$\mathcal{L}_{\text{total}} = L_{\text{box}} + L_{\text{obj}} + L_{\text{cls}} + L_{\text{bpl}} + L_{\text{dis}}. \tag{3}$$

Here, $L_{\text{box}}$ is computed as the CIoU loss between predicted and target boxes; $L_{\text{obj}}$ and $L_{\text{cls}}$ are binary cross-entropy (BCE) losses for objectness and class probabilities, respectively; $L_{\text{bpl}}$ is a smooth $L_1$ loss on the predicted part–body and object–hand offsets, normalised by the associated bounding box size; and $L_{\text{dis}}$ is a BCE-with-logits loss applied only to driver detections (class 0), using a dynamic positive-weight factor to address class imbalance.

*Derivation of the association loss $L_{bpl}$.* Following BPJDet/BPJDetPlus, we supervise association offsets in grid space with a visibility mask on positive cells [7]. Let $G_s$ be positives on level $s$ and $\phi(\cdot) \in \{0, 1\}$ indicate visibility. For each positive cell $i \in G_s$ we regress (i) $k_p$ part→body offsets $\hat{\mathbf{d}'}^{p}_{i,j}$ to targets $\mathbf{t}'^{p}_{i,j}$ and (ii) $k_o$ object→hand offsets $\hat{\mathbf{d}'}^{o}_{i,h}$ to targets $\mathbf{t}'^{o}_{i,h}$, where $\mathbf{d}'$ are displacements normalized

by the feature-map stride. We minimise a Smooth-$L_1$ (Huber) penalty:

$$L_{\text{bpl}} = \sum_{s} \frac{1}{|G_s|} \sum_{i \in G_s} \Bigg[ \sum_{j=1}^{k_p} \phi(v_{i,j}^{p}{>}0)\, \text{SmoothL}_1\big(\hat{\mathbf{d}'}_{i,j}^{\,p} - \mathbf{t}'^{\,p}_{i,j}\big) \tag{4}$$

$$+ \lambda_{o\to h} \sum_{h=1}^{k_o} \phi(v_{i,h}^{o}{>}0)\, \text{SmoothL}_1\big(\hat{\mathbf{d}'}_{i,h}^{\,o} - \mathbf{t}'^{\,o}_{i,h}\big) \Bigg], \tag{5}$$

where we use the elementwise Huber with $\beta$=2.0. This departs from BPJDet's MSE choice but keeps the same supervision space; empirically it improves robustness to occasional annotation noise.

*Derivation of the driver-state loss $L_{dis}$.* For each positive driver detection $b$ (i.e., a matched detection whose target class is the *driver*, class 0), the head outputs a single logit $s_b \in \mathbb{R}$ for the binary label $y_b \in \{0, 1\}$ indicating *distracted*. We use BCE-with-logits with a dynamic positive-class weight:

$$w^{+} \;=\; \lambda_{\text{drv}}\, \frac{N + \varepsilon}{P + \varepsilon},$$

where $P = \sum_b y_b$ and $N = |\mathcal{B}| - P$ are the numbers of positive/negative driver-state labels within the current (layer) batch, $\varepsilon$>0 avoids division by zero, $\lambda_{\text{drv}}$ is a hyperparameter ("drv_pw"), and $\mathcal{B}$ is the set of positive driver detections considered. The loss is

$$L_{\text{dis}} \;=\; \frac{1}{|\mathcal{B}|} \sum_{b \in \mathcal{B}} \Big( -\; w^{+}\, y_b \log \sigma(s_b) - \;(1 - y_b) \log\big(1 - \sigma(s_b)\big) \Big), \tag{6}$$

with $\sigma(\cdot)$ the sigmoid. In code we use `binary_cross_entropy_with_logits` with `pos_weight`= $w^{+}$ and `reduction='mean'`. The final driver-state term $L_{\text{dis}}$ is summed across FPN levels and scaled by the coefficient $w_{\text{dis}}$ ("f_cls_w") in the total loss.

*Association at inference.* After non-maximum suppression (NMS) on the *body* and *part/object* streams, we associate parts/objects to bodies using the anchor points predicted by the body head. For each detected body $b$ with box $B_b = (x_1^b, y_1^b, x_2^b, y_2^b)$, the head outputs $K$ per-class anchor points $\hat{\mathbf{c}}_{b\to j} = (\hat{x}_{b\to j}, \hat{y}_{b\to j})$ in image pixels (obtained by mapping grid-space offsets through the stride and the inverse letterbox transform). For each part/object detection $d$ of class $j$, with centre $\mathbf{c}_d = (x_d, y_d)$, box $\mathbf{b}_d = (x_1^d, y_1^d, x_2^d, y_2^d)$, and score $s_d$, we select the nearest body by

$$\hat{b} = \arg\min_{b} \; \big\|\mathbf{c}_d - \hat{\mathbf{c}}_{b\to j}\big\|_2,$$

subject to a containment check

$$\text{insideIoU}(B_{\hat{b}}, \mathbf{b}_d) = \frac{\text{area}(B_{\hat{b}} \cap \mathbf{b}_d)}{\text{area}(\mathbf{b}_d)} > \tau_{\text{in}},$$

with $\tau_{\text{in}} = 0.6$. If multiple detections of the same class compete for the same body, we keep the one with the higher $s_d$ and update that body's stored entry for class $j$ to $\left[x_d, y_d, s_d, x_1^d, y_1^d, x_2^d, y_2^d\right]$. For datasets without left/right labels, a second pass attaches the remaining unmatched hand to the other hand anchor using the same rule. Finally, for each associated body we output the driver state using a single distracted logit $s_b$; the driver is flagged *distracted* if $\sigma(s_b) \geq \tau_d$ (default $\tau_d = 0.5$), else *safe*.

## 4 Experimental Results

This section presents experimental results aimed at evaluating the impact of incorporating body part association into the detection framework. Specifically, we seek to answer the following research questions:

- Does the addition of body part association improve driver distraction detection performance?
- Does the proposed extension maintain real-time processing speed comparable to the baseline model without body part association?

To address these questions, we compare the BPJ-HBAR model against its baseline counterpart under identical training and evaluation protocols. Performance metrics, inference times, and association accuracy are reported to quantify the trade-offs between accuracy gains and computational cost.

### 4.1 Datasets and setup

All models were trained on a subset of the State Farm Distracted Driver (STF) [16] dataset that we manually re-annotated for object detection with eight classes: *driver*, *head*, *left hand*, *right hand*, *cell phone*, *drink/cup*, *infotainment*, and *wheel*. The training split contains 2345 images, while the validation split contains 350 images. A computer with an Nvidia Quadro RTX 5000 GPU and 16GB of memory will be used.

To evaluate distracted driver detection (DDD) accuracy and inference speed, each BPJ-HBAR approach will be tested on a Jetson Xavier NX edge device with an 8 GB GPU capable of 21 TOPS, configured using NVIDIA JetPack v5.1.3. The evaluation will be conducted on five test datasets (Table 2). In addition to the STF test set, four external datasets (EZZ2021 [17], AUC2 [9], CSIR, and 100-driver [18]) were included to assess the cross-dataset generalisation performance of the algorithms. To ensure consistency across datasets with varying numbers of classes, a subset of 2210 images from the 100-driver dataset containing only the shared classes with the other datasets (STF, EZZ2021, AUC2, CSIR) was randomly selected for evaluation. The cross-dataset evaluation aims to measure the robustness and generalisability of the models when applied to unseen data.

**Table 2.** Characteristics of the distracted driver detection test datasets that will be used.[§] Dataset not yet released to the public, it will be released on request.

| Dataset | Year | Environment | Type of distractions | Image samples |
|---|---|---|---|---|
| STF [16] | 2016 | Real | 1 safe driving, 9 distracted activities | 2247 |
| EZZ2021 [17] | 2021 | Real | 1 safe driving, 9 distracted activities | 3716 |
| AUC2 [9] | 2019 | Real | 1 safe driving, 9 distracted activities | 1074 |
| CSIR[§] | 2022 | Real | 1 safe driving, 9 distracted activities | 512 |
| 100-driver [18] | 2023 | Real | 1 safe driving, 21 distracted activities | 2210 |

### 4.2 Baselines

The lightweight HBAR-DDD algorithm trained on the modified STF training dataset will serve as a baseline. The algorithm was trained for 150 epochs with an input image size of 704 x 704 and an IoU threshold value of 0.65. The other hyperparameters were set to their defaults. The training and validation performance metrics of the algorithm are summarised in Table 4.2. The trained model obtained an mAP@0.5 and mAP@0.95 of 0.864 and 0.66, respectively. These performance metrics are above the minimum mAP target of 0.5, suggesting that the model was able to learn features from the annotated STF dataset.

**Table 3.** Training performance metrics of the standard YOLOv7.

| Model | Parameters (M) | GFLOPS | $\mathbf{mAP}_{\mathbf{val}_{50-95}}$ | Precision | Recall |
|---|---|---|---|---|---|
| HBAR-YOLOv7 | 37.2 | 105.3 | 0.66 | 0.94 | 0.85 |

### 4.3 Configurations

In this work, three configurations are compared:

- **YOLOv7t_HBAR-DDD (Baseline):** prior HBAR-style detector without explicit part–object association.
- **Two-stage HBAR-BPJDet:** detects the driver region and then applies the HBAR-DDD approach.
- **Single-stage HBAR-BPJDet (proposed):** extends BPJDet with *object* predictions and *object→hand* offsets and a *global driver-state head*, performing detection, association, and final state in one pass.

Details about the training and testing environments are summarised in Table 4. The training performance of the models is summarised in Table 5. The best model was chosen based on the lowest logarithmic average mismatch rate (mMR) - a metric that is used to assess the quality of the association of BPJDet.

**Table 4.** Training and testing environment and hyperparameter configuration for the proposed approaches.

| #**Config** | **Details** |
|---|---|
| Training platform | Nvidia Quadro RTX 5000 GPU × 2 + 16GB RAM |
| Testing platform | NVIDIA Jetson Xavier NX (8GB GPU, 21 TOPS) |
| Optimizer | SGD (momentum = 0.937) |
| Framework | PyTorch 2.5.1 |
| Python | 3.10.4 |
| Epochs | 150 |
| Batch size | 8 |
| Initial learning rate (LR) | 0.01 |
| Final learning rate (LR) | 0.01 |
| Loss functions | YOLO loss, body part loss, driver state classification loss |
| Image size | 1280 |
| Body part loss weight ($body\_part\_w$) | 0.015 |
| Driver state loss weight ($cls\_final\_w$) | 0.015 |

**Table 5.** Performance comparison between YOLOv5 BPJ-HBAR and YOLOv7 BPJ-HBAR on the proposed driver monitoring task. R = Recall, P = Precision, mMR = miss-matching rate.

| **Model** | **R** | **P** | **mAP.05** | **mAP.5:.95** | **mMR** |
|---|---|---|---|---|---|
| YOLOv5 BPJ-HBAR | 0.9900 | 0.7900 | 0.9800 | 0.7000 | 0.00098531 |
| YOLOv7 BPJ-HBAR | 0.9990 | 0.79394 | 0.98962 | 0.70153 | 0.00098531 |

### 4.4 Results

Table 6 and Table 6 summarise the F1-score and the accuracy performance of the algorithms evaluated across the five test sets. The average f1-score and accuracy metrics were calculated using only the EZZ2021, AUC2, CSIR, and 100-driver test sets because it is expected that the algorithms will perform well on the STF.

Based on the observed results, three trends stand out:

*(1) Robustness gains from association.* Both BPJDet-based variants substantially outperform the baseline across non-STF datasets, confirming the value of explicit part–object binding. On challenging sets, the proposed single-stage improves F1 over the baseline by +**0.12** (AUC2: 0.67 vs. 0.55), +**0.18** (CSIR: 0.68 vs. 0.50), and +**0.22** (EZZ2021: 0.86 vs. 0.64). Mean F1 improves from **0.54** to **0.65** (+**0.11 absolute**). However, the baseline model performs better than the proposed approach on the challenging 100-driver dataset, producing +**0.07** (100-driver: 0.39 vs. 0.46).

*(2) Accuracy–latency trade-off.* The two-stage variant yields the best mean F1 (**0.88**) and dominates per-dataset peaks (e.g., 0.94 on 100-driver, 0.96 on EZZ2021), but runs at **11.3 FPS**. The proposed single-stage reaches up to **39 FPS** (**3.45× faster** than two-stage) while retaining strong mean F1 (**0.65**). The baseline is slightly faster than YOLOv7 (**41 FPS**) but less robust (mean F1 **0.54**). In short, the proposed single-stage meets real-time requirements while offering commendable robustness gains over the baseline, whereas the two-stage is accuracy-strong but speed-limited.

*(3) Dataset-specific behaviour.* Gaps are largest on *EZZ2021* and *CSIR*, which feature higher intra-class variation, different camera positions, and challenging illumination; here, association mitigates false positives (e.g., detecting objects near passengers) and improves recall under partial occlusion. All methods perform best on *STF* (close to the training distribution). The *100-driver* set remains challenging due to its camera position variations, with the two-stage variant's dedicated person detector providing a measurable advantage.

**Table 6.** Cross-dataset F1 comparison (higher is better). Mean F1 excludes STF.

| Approach | STF | EZZ2021 | AUC2 | CSIR | 100-driver | Mean F1 | FPS |
|---|---|---|---|---|---|---|---|
| HBAR-DDD (Baseline) | 0.86 | 0.64 | 0.55 | 0.50 | 0.46 | **0.54** | 41.0 |
| Two-stage HBAR-DDD | 0.98 | 0.96 | 0.88 | 0.72 | 0.94 | **0.88** | 11.3 |
| HBAR-BPJDet YOLOv5 (proposed) | 0.90 | 0.87 | 0.72 | 0.65 | 0.32 | **0.64** | 27.9 |
| HBAR-BPJDet YOLOv7 (proposed) | 0.95 | 0.86 | 0.67 | 0.68 | 0.39 | **0.65** | 39.0 |

**Table 7.** Cross-dataset accuracy (%) comparison. Mean Accuracy excludes STF.

| Approach | STF | EZZ2021 | AUC2 | CSIR | 100-driver | Mean Acc |
|---|---|---|---|---|---|---|
| HBAR-DDD (Baseline) | 97 | 93 | 80 | 65 | 90 | **82.0** |
| Two-stage HBAR-DDD | 90 | 92 | 85 | 64 | 89 | **82.5** |
| HBAR-BPJDet YOLOv5 (proposed) | 92 | 95 | 81 | 71 | 87 | **83.5** |
| HBAR-BPJDet YOLOv7 (proposed) | 93 | 94 | 78 | 65 | 84 | **80.3** |

### 4.5 Discussion

Overall, the results show a clear trade-off between cross-dataset accuracy/F1 and runtime on the Jetson Xavier NX. The **Two-stage HBAR-DDD** achieves the highest cross-dataset **Mean F1 (0.88)** by first localising the driver with a YOLOv7-tiny detector and then applying the legacy HBAR-DDD recogniser. However, this cascaded design incurs a significant runtime penalty, dropping to

**11.3 FPS**, which is below real-time and confirms the expected latency overhead of two sequential networks.

Both **HBAR-BPJDet (YOLOv5/YOLOv7)** integrate part association in a single pass and hence run substantially faster (≈28–39 FPS). Their **Mean F1** values (**0.64**–**0.65**) are notably higher than the baseline HBAR-DDD (**0.54**), indicating that adding body-part/object association improves cross-dataset discrimination relative to the legacy single-stream baseline without sacrificing real-time performance. Between the proposed single-stage variants, **YOLOv7** attains the highest throughput (**39 FPS**) with slightly better mean F1 than YOLOv5, while **YOLOv5** delivers the best **Mean Accuracy (83.5%)** among the single-stage models.

In short, if maximum accuracy is the sole objective, the two-stage pipeline is strongest but slow; if **real-time deployment** is the priority, the **single-stage HBAR-BPJDet** variants provide a better balance—substantially faster while still outperforming the baseline in cross-dataset F1. These findings support the core claim that **explicit body-part association improves generalisation** and can be realised in a single pass to preserve practical speed on embedded GPUs.

## 5 Conclusions

We presented a lightweight approach to robust distracted driver detection for edge devices. Building on BPJDet, our model jointly predicts bodies, parts, and distraction objects and introduces object→hand offsets to bind evidence to the correct driver, alongside a global driver–state head that preserves a single-stage pipeline. Trained on a re-annotated STF subset (8 classes), the proposed single-stage HBAR-BPJDet improves cross-dataset mean F1 from 0.545 (YOLOv7t_HBAR-DDD) to 0.65 at 39 FPS, offering a practical accuracy–latency trade-off for embedded deployment; a two-stage variant attains higher mean F1 (0.88) but runs at 11 FPS. The explicit association reduces near-object false positives and mitigates occlusion-related misses. Limitations include dataset scale/diversity and reliance on fine-grained hand–object labels. Future work will investigate semi-supervised labelling, lightweight temporal cues, and hardware-aware quantisation/distillation to further narrow the accuracy gap to multi-stage systems while maintaining real-time throughput.

## References


1. Zandamela, F., Kunene, D., Skosana, V., Stoltz, G.: Lightweight YOLO for distracted-driver detection on edge devices. MATEC Web Conf. **406**, 10001 (2024)
2. Dong, B.T., Lin, H.Y.: An on-board monitoring system for driving-fatigue and distraction detection. In: 2021 22nd IEEE Int. Conf. Industrial Technology (ICIT), vol. 1, pp. 850–855. IEEE, Valencia, Spain (2021)
3. Sajid, F., Javed, A.R., Basharat, A., Kryvinska, N., Afzal, A., Rizwan, M.: An efficient deep-learning framework for distracted-driver detection. IEEE Access **9**, 169270–169280 (2021)

4. Wang, J., Wu, Z., Li, F., Zhang, J.: A data-augmentation approach to distracted-driving detection. Future Internet **13**(1), 1 (2020)
5. Li, Z., Zhao, X., Wu, F., Chen, D., Wang, C.: A lightweight and efficient distracted-driver detection model fusing convolutional neural network and vision transformer. IEEE Trans. Intell. Transp. Syst. **25**(12), 19962–19978 (2024). https://doi.org/10.1109/TITS.2024.3447041
6. Duan, C., Gong, Y., Liao, J., Zhang, M., Cao, L.: FRNet: DCNN for real-time distracted-driving detection toward embedded deployment. IEEE Trans. Intell. Transp. Syst. **24**(9), 9835–9848 (2023). https://doi.org/10.1109/TITS.2023.3270879
7. Zhou, H., Jiang, F., Si, J., Ding, Y., Lu, H.: BPJDet: extended object representation for generic body-part joint detection. IEEE Trans. Pattern Anal. Mach. Intell. **46**(6), 4314–4330 (2024). https://doi.org/10.1109/TPAMI.2024.3354962
8. Wang, C.Y., Bochkovskiy, A., Liao, H.Y.M.: YOLOv7: trainable bag-of-freebies sets new state of the art for real-time object detectors. In: Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition, pp. 7464–7475. IEEE, Vancouver, Canada (2023)
9. Eraqi, H.M., Abouelnaga, Y., Saad, M.H., Moustafa, M.N.: Driver-distraction identification with an ensemble of convolutional neural networks. J. Adv. Transp. **2019**, (2019)
10. Yan, C., Coenen, F., Zhang, B.: Driving-posture recognition by convolutional neural networks. IET Comput. Vision **10**(2), 103–114 (2016)
11. Liu, D., Yamasaki, T., Wang, Y., Mase, K., Kato, J.: Toward extremely lightweight distracted-driver recognition with distillation-based neural architecture search and knowledge transfer. IEEE Trans. Intell. Transp. Syst. **24**(1), 764–777 (2023). https://doi.org/10.1109/TITS.2022.3217342
12. Tan, M., Pang, R., Le, Q.V.: EfficientDet: Scalable and efficient object detection. In: Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition, pp. 10781–10790 (2020)
13. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114. PMLR (2019)
14. Ali, A., Gaikov, G., Rybalchenko, D., Chigorin, A., Laptev, I., Zagoruyko, S.: PairDETR: joint detection and association of human bodies and faces. In: Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp. 423–432 (2024). https://doi.org/10.1109/CVPR52733.2024.00048
15. Gao, Z., Zhou, H., Sharma, A., Zheng, M., Planche, B., Chen, T., Wu, Z.: PBADet: a one-stage anchor-free approach for part–body association. arXiv preprint arXiv:2402.07814 (2024)
16. Montoya, A., Holman, D., Smith, T., Kan, W.: State Farm distracted driver detection. https://kaggle.com/competitions/state-farm-distracted-driver-detection (2016). Accessed 28 Mar 2022
17. Ezzouhri, A., Charouh, Z., Ghogho, M., Guennoun, Z.: Robust deep-learning-based driver-distraction detection and classification. IEEE Access **9**, 168080–168092 (2021)
18. Wang, J., Li, W., Li, F., Zhang, J., Wu, Z., Zhong, Z., Sebe, N.: 100-Driver: a large-scale, diverse dataset for distracted driver classification. Manuscript under review (2022)