Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.053710

ARTICLE

GDMNet: A Unified Multi-Task Network for Panoptic Driving Perception

Yunxiang Liu, Haili Ma, Jianlin Zhu* and Qiangbo Zhang

School of Computer Science and Information Engineering, Shanghai Institute of Technology, Shanghai, 201418, China

*Corresponding Author: Jianlin Zhu. Email: 226142135@mail.sit.edu.cn

Received: 08 May 2024 Accepted: 08 July 2024 Published: 15 August 2024

ABSTRACT

To enhance the efficiency and accuracy of environmental perception for autonomous vehicles, we propose GDMNet, a unified multi-task perception network for autonomous driving, capable of performing drivable area segmentation, lane detection, and traffic object detection. Firstly, in the encoding stage, features are extracted, and Generalized Efficient Layer Aggregation Network (GELAN) is utilized to enhance feature extraction and gradient flow. Secondly, in the decoding stage, specialized detection heads are designed; the drivable area segmentation head employs DySample to expand feature maps, the lane detection head merges early-stage features and processes the output through the Focal Modulation Network (FMN). Lastly, the Minimum Point Distance IoU (MPDIoU) loss function is employed to compute the matching degree between traffic object detection boxes and predicted boxes, facilitating model training adjustments. Experimental results on the BDD100K dataset demonstrate that the proposed network achieves a drivable area segmentation mean intersection over union (mIoU) of 92.2%, lane detection accuracy and intersection over union (IoU) of 75.3% and 26.4%, respectively, and traffic object detection recall and mAP of 89.7% and 78.2%, respectively. The detection performance surpasses that of other single-task or multi-task algorithm models.

KEYWORDS

Autonomous driving; multitask learning; drivable area segmentation; lane detection; vehicle detection

1 Introduction

Traffic environment perception is one of the key modules in autonomous vehicles, responsible for path planning in drivable areas to guide safe vehicle operation. However, the conditions of drivable areas often change rapidly during vehicle travel. Frequent path planning cannot ensure high-speed vehicle operation, making lane markings a critical element for stable vehicle travel. Typically, lane markings appear as continuous smooth curves, providing ideal guidance for vehicles. However, sometimes the road ahead is occupied by traffic objects, forcing the vehicle to brake intermittently, which not only discomforts passengers but also poses a safety threat to the vehicle. In intelligent traffic environments, typical traffic objects are primarily other vehicles [1]. Therefore, drivable area segmentation, lane detection, and traffic object detection are considered fundamental requirements for perception tasks [2].

In recent years, with the development of deep neural networks and high-performance hardware, deep learning algorithms have shown significant effectiveness in autonomous driving perception tasks. In existing works, drivable area segmentation, lane detection, and object detection are typically treated as three separate tasks, and their respective network models have achieved good results in their respective fields. For example, the YOLO series [3] networks are suitable for object detection, ENet [4] for lane detection, and PSPNet [5] for drivable area segmentation, among others. These models achieve excellent detection results in single-task perception. However, limited by the computational resources of mobile onboard systems, it is difficult to achieve an effective balance between reliability and real-time performance when using multiple models simultaneously for inference tasks.

Researchers have found that the idea of multi-task learning can be utilized to share information between network branches, where different tasks use the same network-extracted features and employ different detection heads for multi-task detection. This approach can improve efficiency while maintaining good performance. For example, MultiNet [6] is used for scene classification, object detection, and semantic segmentation. This architecture shares the same encoder and uses three decoders for the three tasks. HybridNets [7] use a bidirectional feature pyramid network to enhance feature fusion and improve detection accuracy. The YOLOP [2] model, based on an encoder-decoder structure, shares feature information, extracts and fuses features, and then performs detection separately, showing good detection performance in multi-task perception. However, the aforementioned approaches suffer from issues such as insufficient emphasis on image feature extraction by the network, leading to loss of extracted feature information, or imperfect optimization of detection heads, thereby affecting detection results.

To address the aforementioned issues, the main research contributions of our paper are as follows:

(1) We propose a unified multi-task network suitable for panoramic driving perception, including drivable area segmentation, lane detection, and traffic object detection.

(2) We employ GELAN during the feature extraction stage to facilitate feature propagation and reuse, thereby enhancing the accuracy and generalization ability of the model in tasks such as object detection.

(3) We analyze and study the characteristics of drivable area segmentation, lane detection, and traffic object detection tasks, and design or optimize the corresponding detection heads accordingly.

2 Related Work

2.1 Semantic Segmentation

Drivable area segmentation and lane detection belong to semantic segmentation, which plays an important role in understanding traffic scenes at the pixel level, providing more precise information for intelligent vehicles. PSPNet [5] proposes the spatial pyramid pooling module to extract feature information at different scales, enhancing drivable area segmentation performance. EdgeNet [8] combines edge detection within the drivable area to achieve better detection results without sacrificing too much inference speed. ENet [4] utilizes techniques like dilated convolutions to accelerate model speed but results in lower lane detection accuracy. ENet-SAD [9] employs self-attention dilation techniques, enabling low-level feature maps to learn content from high-level feature maps, maintaining model lightweightness while improving performance. TSA-LNet [10] utilizes Two-directional Separation Attention (TSA) to infer attention maps along both horizontal and vertical directions, performing adaptive feature refinement by multiplying attention maps with input feature maps to capture both local texture and global contextual information of lanes.

2.2 Object Detection

Object detection is a critical task in computer vision, aiming to accurately identify and localize the boundaries and positions of multiple objects within images or videos. Object detection is mainly divided into two-stage and one-stage detection algorithms. Faster R-CNN [11] utilizes the Region Proposal Network (RPN) as a candidate box generator. The generated candidate boxes are then inputted into classification and localization sub-networks to accomplish the final object classification and localization. While two-stage algorithms achieve good performance in detection accuracy, the two-stage process results in slower speeds, making it unsuitable for practical applications. YOLO [3] algorithms are typical one-stage object detection algorithms, treating the object detection task as a regression task across the entire image. YOLOv3 [12] uses an Feature Pyramid Network (FPN) [13] network to fuse multiscale information and enhance detection performance. YOLOv4 [14] and YOLOv5 [15] enhance the detection performance by improving the network's loss function and utilizing data augmentation techniques. YOLOv9 [16] uses programmable gradient information (PGI) to provide complete input information for target task calculation, obtaining reliable gradient information to update network weights and utilizing the GELAN to improve parameter utilization and model performance. TSF [17] combines point clouds with images to generate enhanced point clouds. Than it utilizes processing results to accomplish non-maximum suppression (NMS) for 3D object detection in autonomous driving.

2.3 Multi-Task Model

Multi-task models aim to simultaneously perform multiple related tasks, typically learning image features with a shared encoder and employing dedicated decoders for each task during prediction. Such structures enable better generalization and information utilization. MultiNet [6], used for scene classification, object detection, and semantic segmentation, shares the same encoder architecture and utilizes three decoders for the respective tasks. However, due to network constraints, the input image size remains fixed. HybridNets [7] enhance feature fusion with a bidirectional feature pyramid network, completing multi-task perception and improving detection accuracy. Yet, the use of lightweight networks in HybridNets results in poorer robustness. The YOLOP [2] model, based on an encoder-decoder structure, shares feature information, extracts and fuses features, and performs detection separately, enabling drivable area segmentation, lane detection, and traffic object detection. However, its network lacks sufficient emphasis on feature extraction, and the simplistic design of detection heads results in issues such as missed detections, false alarms, and segmentation edge errors in scenarios involving local occlusion, adverse weather, or low-light conditions. Overall, the aforementioned models achieve multi-task detection to some extent, but their performance is not satisfactory. Therefore, this paper proposes a unified multi-task perception network to meet the practical requirements of autonomous driving multi-task perception.

3 Methods

3.1 Network Structure

As illustrated in Fig. 1, the GDMNet is based on an encoder-decoder structure comprising a backbone network and a neck network in the encoder, which are responsible for feature extraction and feature fusion, respectively. The decoder consists of specialized detection heads for drivable area segmentation, lane detection, and traffic object detection. All three tasks utilize features extracted during the encoding stage. The quality of these features impacts the detection results, and the combined influence of these three tasks affects the model's optimization. The input image first undergoes feature

extraction by the backbone network, yielding feature maps B_2, B_3, B_4, and B_5 at different scales, with sizes of 1/4, 1/8, 1/16, and 1/32 of the original image, respectively. These feature maps then progress to the neck network, where feature maps B_2, B_3, and B_4 are generated by progressively expanding and merging with B_3, B_4, and B_5 in a top-down manner. Subsequently, the resulting feature maps F_3, F_4, and F_5 are merged with N_3, N_4, and N_5 in a bottom-up manner, and feature information at different scales is outputted to the traffic object detection head. N_4 undergoes three DySample expansions to obtain the drivable area segmentation result. For lane detection, N_4 is first upsampled once and fused with B_3, and the resulting feature map is processed by the FMN, followed by two additional upsampling operations to obtain the lane detection result. Finally, the outputs of object detection and semantic segmentation constitute the output of autonomous panoramic perception for driving.

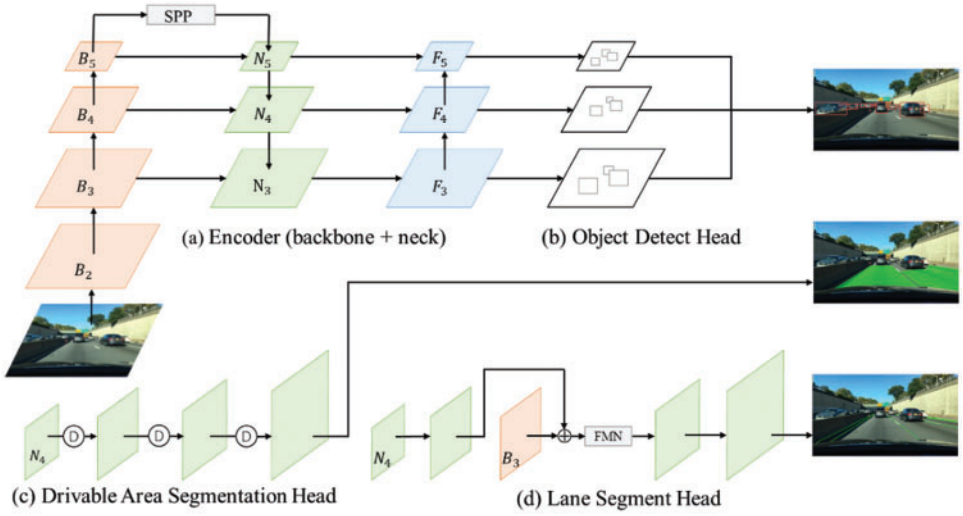

Figure 1: Schematic of network structure: (a) encoder (backbone + neck), (b) object detect head, (c) drivable area segmentation head, (d) lane segment head

3.2 Encoder

The three task features are associated and shared primarily during the encoding stage. Drivable areas, lane markings, and traffic objects in images are typically adjacent to each other without overlapping. Therefore, distinguishing and extracting features, particularly the edge features of objects, during the encoding stage is crucial for subsequent detection. The encoder mainly consists of a backbone network and a neck network. The backbone network is CSPDarknet [14], which addresses the problem of gradient vanishing during feature extraction and supports feature propagation and reuse, thereby reducing parameters and computational complexity. In the neck network section, it comprises the Spatial Pyramid Pooling (SPP) [18] module and the FPN [12]. SPP generates and merges features of different scales, while FPN merges features from different semantic levels in a top-down manner, ensuring that the generated features contain multi-scale and multi-semantic information. SPP consists of two convolutional layers with kernel sizes of 1×1, stride of 1, and max-pooling layers with fixed kernel sizes of 5, 9, and 13, respectively, concatenated in a cascade structure. CSPDarknet and FPN use CSPNet [19] to enhance feature extraction ability and address the gradient vanishing problem as a residual structure. However, during feature fusion, due to the sequential use of multiple residual structures, as the model depth increases, the gradient path also elongates, leading to gradient information loss and ineffective improvement in model accuracy. To address this, we replace CSPNet

with GELAN. In the backbone network, the feature extraction process at each layer consists of convolution followed by GELAN. The structure of CSPNet and GELAN is illustrated in Fig. 2.

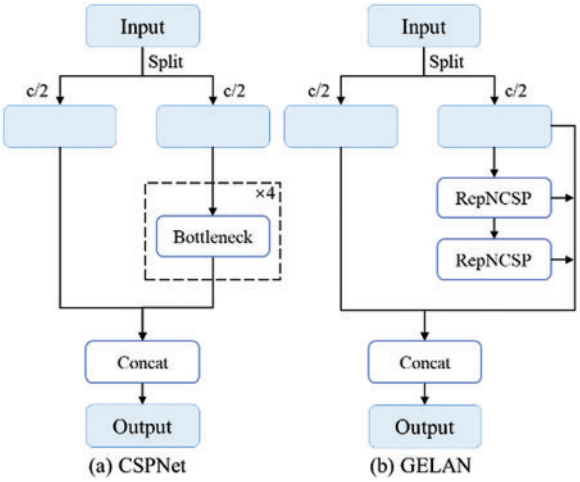

Figure 2: Structure diagram of CSPNet and GELAN: (a) CSPNet, (b) GELAN

The GELAN module inherits the idea of feature extraction and branching from CSPNet. First, the feature map is split into two parts after feature extraction: the main path and the branch path, facilitating the separation and processing of features at different scales and levels. Next, the feature map of the main path undergoes two RepNCSP operations to increase nonlinearity, while aggregating feature information during feature propagation and finally concatenating. Lastly, the feature map of the branch path is concatenated with that of the main path, which is performed in the channel dimension, aggregating feature information layer by layer to retain both low-level and high-level information, obtaining more gradient information, and enhancing model accuracy. There are differences between the Bottleneck in CSPNet and RepNCSP in GELAN: the features are split into two parts in Bottleneck, one part undergoes two convolution operations and is then merged with the other part, while in RepNCSP, one part undergoes convolution operation and the other part undergoes Bottleneck operation, and after merging, undergoes convolution processing again. RepNCSP explicitly fuses features at different stages to enhance information flow, helping to avoid gradient vanishing and information bottleneck problems.

3.3 Decoder

3.3.1 Drivable Area Segmentation Head

In YOLOP, the result of the drivable area head is obtained by upsampling N_4 from the neck network three times consecutively. The upsampling method utilizes nearest-neighbor interpolation to enlarge the feature map size, where nearest-neighbor interpolation only considers information from adjacent pixels. It fills the pixels in the target feature map directly using the original feature map. This method has low computational complexity but may result in discontinuous grayscale values in the interpolated image, causing noticeable aliasing at image edges and significantly affecting the drivable area segmentation result. Therefore, we adopt DySample [20] as the upsampling method for the detection head, as illustrated in Fig. 1c. DySample mitigates the issues associated with nearest-neighbor interpolation, ensuring smoother transitions between pixels and improving the drivable area segmentation result. The structure of DySample is illustrated in Fig. 3.

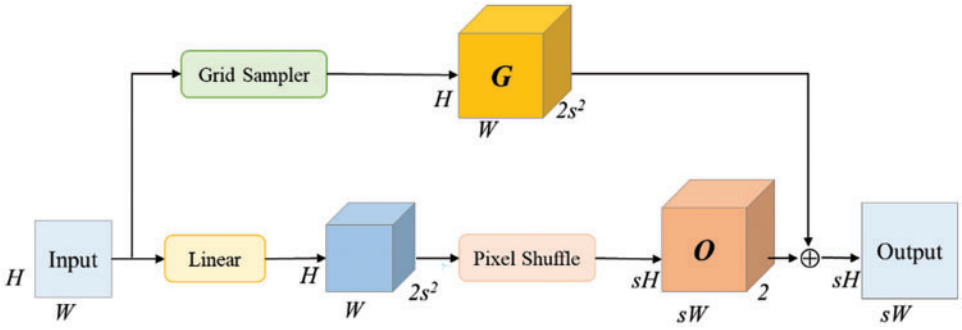

Figure 3: Structure diagram of DySample

Here, s represents the dilation rate, G denotes the coordinates of grid sampling points, and O represents the position compensation generated by the dynamic sampling point generator. DySample first determines the initial sampling points for each upsampling position using bilinear interpolation. For each upsampling position, DySample learns to generate a set of offset values. These offsets reflect the fine-tuning requirements relative to the initial sampling points, aiming to better fit the content and boundary information of the input features during upsampling. By applying the generated offsets, each initial sampling point is moved to a new position. The sampling points adjusted by the offsets correspond to new feature values on the continuous feature map. Through the grid sampling function, DySample can obtain precise feature values for these points from the continuous feature map, thereby achieving refined filling of the upsampling positions on the original input feature map. Throughout the upsampling process, DySample effectively breaks the fixed interpolation rules of conventional upsampling methods by learning to generate content-related offsets, enabling dynamic, flexible, and semantically meaningful upsampling of the input feature map. This process not only improves the quality of the upsampling results but also enhances accuracy, especially in edge processing tasks.

3.3.2 Lane Detection Head

Drivable area segmentation and lane detection are tasks related to semantic segmentation. Unlike drivable areas, lane markings are thin and elongated, making it easy for the encoder to lose original detailed information after multiple feature map analyses. If simple upsampling is directly applied to the feature map to restore it to the original image size, the detection performance may be unsatisfactory in complex scenarios, such as long-distance lane markings or intersections. To address this issue, we have redesigned the lane detection head, as shown in Fig. 1d. The feature map N_4 is upsampled once to a size of $\frac{w}{4} \times \frac{h}{4}$, and then fused with the high-resolution feature map B_3. This fusion captures contextual information at different scales but fails to establish long-distance contextual dependencies and hinders the propagation of information over long distances or across occlusions. To resolve the aforementioned challenges, we adopt the FMN [21] to process the merged feature map. Subsequently, after two upsampling operations, the feature map is restored to the original image size to obtain the segmentation result. The structure of FMN is illustrated in Fig. 4.

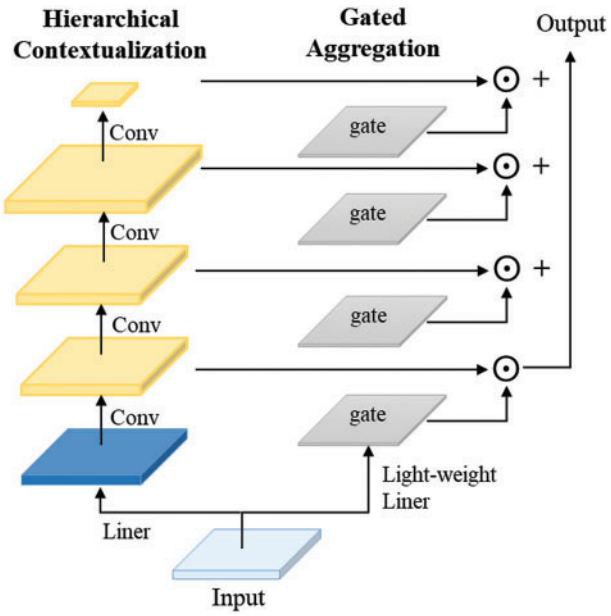

Figure 4: Structure diagram of FMN

First, the given image feature maps are processed hierarchically to generate multi-level feature maps covering different granularity of contexts from short-range to long-range. Next, using spatial and hierarchical-sensitive gating mechanisms, the feature maps at each level are weighted and merged to form a single feature map consistent with the input size, allowing the model to dynamically adjust the attention to different hierarchical contexts based on query content. Finally, a linear layer generates modulators and implements element-wise modulation, multiplying the modulators with the original features of query tokens to generate the final refined feature representations. This series of operations not only preserves the fine-grained information of query tokens but also successfully incorporates contextual information from local to global, effectively handling the fused features from earlier stages, thereby significantly improving the detection performance of the model.

3.3.3 Object Detection Head

As shown in Fig. 1b, for traffic object detection, we adopt an anchor-based multi-scale detection scheme. Utilizing Path Aggregation Network (PAN) [22] in a bottom-up propagation manner, we employ 3×3 convolutional kernels with a stride of 2 to downsample feature maps, corresponding to different stages of feature maps in FPN. This fusion of low-level features with high-level semantic features results in multi-scale feature maps. Each grid of the feature maps is assigned three anchor boxes with different aspect ratios. The detection head predicts the position offsets, width-height scaling, and confidence scores of traffic objects, thereby obtaining the traffic object detection results.

3.4 Loss Function

The network comprises three decoders, and the loss for multi-tasking consists of three components. The object detection loss L_{det} is the weighted sum of classification loss, object loss and bounding box loss, as shown in Eq. (1), where a_1, a_2, and a_3 are weights: $a_1 = 0.5$, $a_2 = 1.0$, and $a_3 = 0.05$, emphasizing the significance of object presence detection.

$$L_{\text{det}} = a_1 L_{class} + a_2 L_{obj} + a_3 L_{box} \tag{1}$$

The classification loss and object loss employ focal loss [23], aimed at reducing the impact of correctly classified samples, thereby forcing the network to focus on hard-to-classify samples. The bounding box loss calculation uses MPDIoU [24], as shown in Eq. (2):

$$MPDIoU = IoU - \frac{d_1^2}{w^2 + h^2} - \frac{d_2^2}{w^2 + h^2} \tag{2}$$

$$d_1^2 = \left(x_1^{Bgt} - x_1^{Bprd}\right)^2 + \left(y_1^{Bgt} - y_1^{Bprd}\right)^2 \tag{3}$$

$$d_2^2 = \left(x_2^{Bgt} - x_2^{Bprd}\right)^2 + \left(y_2^{Bgt} - y_2^{Bprd}\right)^2 \tag{4}$$

where $\left(x_1^{Bgt}, y_1^{Bgt}\right)$ and $\left(x_2^{Bgt}, y_2^{Bgt}\right)$ are the coordinates of the top-left and bottom-right points of the ground truth bounding box, and $\left(x_1^{Bgt}, y_1^{Bgt}\right)$ and $\left(x_2^{Bgt}, y_2^{Bgt}\right)$ are the coordinates of the top-left and bottom-right points of the predicted bounding box, w and h are the width and height of the input image. MPDIoU considers the geometric features of both the predicted and ground truth bounding boxes by computing the absolute position of the midpoint between them. It includes factors such as the overlap or non-overlap region, distance between the center points, and width and height deviations, all of which can be calculated from the coordinates of the top-left and bottom-right points of the ground truth and predicted boxes. When the ground truth and predicted boxes are of the same proportion but differ in actual size, the size difference is inversely proportional to the value of MPDIoU. By minimizing the difference between the predicted and annotated boxes, the accuracy of object detection is improved, and the convergence of network training is accelerated. The calculation process of MPDIoU is simplified, requiring fewer complex computations during training, thereby reducing training time and improving training efficiency.

The drivable area segmentation loss $L_{da\text{-}seg}$ adopts cross entropy loss (L_{ce}) [25], which aims to minimize the classification error between the network output pixels and the target. The lane detection loss $L_{ll\text{-}seg}$ employs L_{ce} and intersection over union loss (L_{IoU}), $L_{IoU} = 1 - TP/(TP + FP + FN)$, which is particularly effective for predicting lane lines in sparse categories.

The final loss function of the model is the weighted sum of three parts of loss, as shown in Eq. (5), where r_1, r_2, and r_3 are the weights, specifically $r_1 = 0.2$, $r_2 = 0.2$, and $r_3 = 0.2$, ensuring a balanced consideration of each task's importance within the model.

$$L_{all} = r_1 L_{\text{det}} + r_2 L_{da-seg} + r_3 L_{ll-seg} \tag{5}$$

4 Results and Analysis

4.1 Dataset

The dataset used in the experiment is the BDD100K [26] dataset, created by the Berkley Deep Drive project team at the University of California, Berkeley, based on real road scenes. This dataset provides a large number of high-quality images for research and development of autonomous driving systems, covering various scenarios such as urban, suburban, and highway environments, as well as different weather conditions (sunny, rainy, snowy, etc.) and lighting conditions. The BDD100K dataset comprises 70,000 training set images, 10,000 validation set images, and 20,000 test set images. Since the labels for the test set are not publicly available, the validation set is used to evaluate algorithm models.

4.2 Implementation Details

In the experiment, the algorithm models were trained on the Ubuntu 20.04 operating system. The programming language used was Python 3.9, and the deep learning framework PyTorch version was 1.13.1, with CUDA version 11.7. The GPU used for model training and testing was the NVIDIA GeForce RTX 3090.

To incorporate more prior knowledge of objects in traffic scenes, the k-means clustering algorithm was employed to obtain prior anchor boxes from the dataset. Adam was selected as the optimizer for training the model, with β_1 and β_2 set to 0.937 and 0.999, respectively. The initial learning rate and final learning rate were set to 0.001 and 0.002, with a weight decay coefficient of 0.0005. The α and γ in the focus loss are set to 0.2 and 2, respectively. The training consisted of 100 epochs, with a batch size of 28 during training and 1 during testing. The input image size was adjusted from 1280×720 to 640×384. Data augmentation techniques were utilized to enhance the model's robustness in various environments, including adjusting the image's hue and saturation, as well as applying random rotation, scaling, translation, shearing, and horizontal flipping to introduce geometric distortions to the images.

4.3 Evaluation Metrics

In this article, the drivable area segmentation subtask in the segmentation task employs mIoU as the evaluation metric, calculated by averaging IoU between the feasible driving area and the non-feasible driving area. The lane detection subtask is evaluated using two metrics: accuracy (Acc) and IoU, which are calculated using Eqs. (6) and (7), respectively:

$$\text{Acc} = \frac{TP + TN}{N} \tag{6}$$

$$IoU = \frac{B_{gt} \cap B_{prd}}{B_{gt} \cup B_{prd}} \tag{7}$$

where TP and TN represent the numbers of true positive and true negative samples predicted correctly, respectively, and N is the total number of samples. B_{gt} and B_{prd} respectively detect the real area and the predicted area.

The traffic object detection subtask utilizes recall and the mean average precision (mAP50) for each class with an IoU threshold of 0.5 as evaluation metrics, calculated by Eqs. (8) and (9), respectively:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$mAP50 = \frac{1}{n_{class}} \int_0^1 P(R)\ dR \tag{9}$$

where FN represents the number of false negatives, N_{class} represents the number of classes, P represents precision, and R represents recall. In the model inference speed section, the metric is frames per second (FPS).

4.4 Ablation Experiments

To validate the effectiveness of the proposed method in this paper, we conducted ablation experiments using YOLOP as the baseline model, and the results are shown in Table 1. Firstly, by employing GELAN in the encoder, the network better extracts features and propagates gradients during the

encoding phase. Experimental results indicate that GELAN leads to significant improvements in all three tasks: drivable area segmentation, lane detection, and traffic object detection. Secondly, incorporating a detection head designed based on the characteristics of drivable area segmentation and lane detection into the network resulted in a 0.4% increase in mIoU for drivable area segmentation, a 2.8% and 0.4% increase in Acc and IoU for lane detection, respectively, demonstrating a notable improvement. Lastly, using MPDIoU in the loss function for traffic object detection to accurately assess the matching degree of detection boxes resulted in a 0.5% increase in Recall and a 1% increase in mAP50. Compared to the baseline model, our model achieved improvements of 1.1%, 6.8%, and 1.6% in mIoU for drivable area, Acc for lane detection, and mAP50 for traffic object detection, respectively. Experimental results demonstrate that compared to YOLOP, GDMNet uses GELAN in the feature extraction stage to promote feature propagation and reuse, thereby enhancing the model's accuracy and generalization capability in detection tasks. Additionally, the detection heads are redesigned or improved based on the characteristics of each task, enabling the model to achieve more precise detection results.

Table 1: Results of ablation experiments

Model	GELAN	Detect head	MPDIoU	mIoU/%	Acc/%	IoU/%	Recall/%	mAP50/%	FPS
A	×	×	×	91.1	68.5	26.4	89.2	76.6	**84.7**
B	√	×	×	91.9	71.6	26.1	89.4	77.4	62.1
C	√	√	×	**92.3**	74.4	**26.5**	89.2	77.2	56.1
D	√	√	√	92.2	**75.3**	26.4	**89.7**	**78.2**	56.5

To validate the effectiveness of the GELAN in the encoder, we designed experiments to compare the impact of different cross-stage networks, including CSPNet [19], C3 [15], C2f [27], C3HB [28], and GELAN [16], on the model's performance. The experimental results are presented in Table 2. C3 adopts the concept of inheriting gradient flow, utilizing three consecutive convolutions in the Bottleneck. While it leads to limited improvements in detection results, its impact is moderate. C2f employs serial concatenation operations in the Bottleneck. It exhibits significant enhancement in drivable area segmentation and lane detection accuracy, but its effectiveness in traffic object detection is limited. C3HB combines gradient flow with recursive gate convolution to efficiently incorporate high-order spatial interactions among features. However, its overall performance is inferior to GELAN. Benefiting from its superior design, GELAN ensures effective feature extraction and gradient propagation during the encoding phase, resulting in consistently good detection performance across all metrics.

Table 2: Comparative experimental results of cross stage partial networks

Model	mIoU	Acc/%	IoU/%	Recall/%	mAP50/%
CSPNet [19]	91.1	68.5	26.4	89.2	76.6
C3 [15]	91.5	69.6	26.4	89.3	7.7
C2f [27]	91.6	70.7	**26.7**	88.9	76.6
C3HB [28]	91.9	69.9	26.3	89.1	76.7
GELAN [16]	**91.9**	**71.6**	26.1	**89.4**	**77.4**

To validate the effectiveness of our designed drivable area segmentation detection head (DASH), we conducted experiments to compare it with the baseline model. The experimental results are shown in Table 3. Our designed detection head achieved improvements of 0.4%, 1.8%, and 1.1% in Acc, IoU, and mIoU for drivable area, respectively. The results demonstrate that employing DySample as the upsampling method for feature maps in drivable area segmentation benefits feature map recovery, leading to enhanced detection accuracy.

Table 3: Comparison experiment of drivable area segmentation detection heads

Model	Acc/%	IoU/%	mIoU/%
Baseline	97.2	85.5	91.1
DASH	**97.6**	**87.3**	**92.2**

The comparison results between our designed lane detection head (LDH) and the baseline model are presented in Table 4. The Acc and IoU metrics improved by 6.6% and 0.2%, respectively. The experimental results indicate that integrating high-resolution feature maps from the early stages in the lane detection head, followed by processing through FMN, enables the network architecture to adapt to the characteristics of thin or intersecting lane lines, thereby significantly improving lane detection accuracy.

Table 4: Comparative experiment of lane detection heads

Model	Acc/%	IoU/%	mIoU/%
Baseline	68.5	26.4	**62.4**
LDH	**75.1**	**26.6**	**62.4**

4.5 Comparison Analysis

4.5.1 Drivable Area Segmentation Results

In the comparative experiments for drivable area, we selected single-task detection algorithms SegNet [29] and PSPNet [5], as well as multi-task detection algorithms MultiNet [6], DLT-Net [1], YOLOP [2], and TDL-YOLO [30] as comparison models. The experimental results are presented in Table 5. From the data in the table, it can be observed that single-task algorithms exhibit higher accuracy than some multi-task algorithms. However, multi-task models with better structures demonstrate superior detection accuracy. Among them, our model achieved the highest mIoU of 92.2%.

Table 5: Comparison experiment results of drivable area segmentation

Model	mIoU/%
SegNet [29]	79.6
PSPNet [5]	89.6
MultiNet [6]	71.6
DLT-Net [1]	72.1
YOLOP [2]	91.1

(Continued)

Table 5 (continued)	
Model	mIoU/%
TDL-YOLO [30]	91.4
GDMNet	**92.2**

4.5.2 *Lane Detection Results*

In the comparative experiment of lane detection, we selected single-task algorithms ENet [4], SCNN [31], ENet-SAD [9], and LaneNet [32], and multi-task algorithms YOLOP [2] and TDL-YOLO [30] as the comparison models. The experimental results are shown in Table 6. The results indicate that the network structure of single-task perception algorithms has limited effectiveness in feature extraction, resulting in lower detection accuracy. Multi-task perception algorithms perform better than single-task perception algorithms, with our algorithm achieving the highest accuracy, reaching 75.3%, significantly higher than other models.

Table 6: Comparative experimental results of lane detection

Model	Acc/%	IoU/%
ENet [4]	34.1	14.6
SCNN [31]	35.8	15.8
ENet-SAD [9]	36.6	16.0
LaneNet [32]	60.6	–
YOLOP [2]	68.5	26.4
TDL-YOLO [30]	72.3	**26.5**
GDMNet	**75.3**	26.4

4.5.3 *Traffic Object Detection Results*

In the comparative experiment of traffic target detection, we selected single-task perception algorithms Faster R-CNN [11], YOLOv5s [15], and multi-task algorithms MultiNet [6], DLT-Net [1], YOLOP [2], and TDL-YOLO [30] as comparison models. The experimental results are shown in Table 7. These models exhibit good feature processing for images, all achieving satisfactory detection results. Among them, our model demonstrates the best detection performance, with Recall and mAP50 reaching 90.7% and 78.9%, respectively.

Table 7: Comparative experimental results of traffic object detection

Model	Recall/%	mAP50/%
Faster R-CNN [11]	81.2	64.9
YOLOv5s [15]	86.8	77.2
MultiNet [6]	81.3	60.2

(Continued)

Table 7 (continued)

Model	Recall/%	mAP50/%
DLT-Net [1]	89.4	68.4
YOLOP [2]	89.2	76.6
TDL-YOLO [30]	88.7	77.9
GDMNet	**90.7**	**78.9**

4.6 Algorithm Verification

To verify the effectiveness of the proposed algorithm in real-world scenarios, six different typical real road scenes were selected from the BDD100K dataset's test set for comparative experiments between GDMNet and YOLOP, as shown in Fig. 5. The left and right sides of each group represent the detection results of YOLOP and GDMNet, respectively. The first row of images represents urban traffic scenes, where the complexity of city roads and rapid changes in vehicles and roads are evident. From the comparison results, it can be seen that YOLOP experiences interruptions in lane detection, while the proposed algorithm detects continuous and smooth lane lines, performing well in multi-lane situations. The second row of images represents nighttime and tunnel traffic scenes, where poor lighting and various light sources impact the visual detection system. The detection results show that YOLOP has unsatisfactory performance in detecting drivable areas, with instances of missed detections, whereas the proposed algorithm accurately and completely detects drivable areas, maintaining good performance despite the influence of multiple light sources. The third row of images represents traffic scenes under rainy and sunny weather conditions, where the camera's visibility distance is reduced due to adverse weather, being easily obstructed by rain and fog, or significantly affected by strong sunlight. The experimental results indicate that YOLOP exhibits false detections and missed detections in vehicle recognition, while the proposed algorithm provides complete vehicle detection, with intact lane lines and drivable areas, demonstrating the robustness of the proposed algorithm. The visualization experiments validate the effectiveness and reliability of GDMNet, indicating its suitability for real-world autonomous driving scenarios.

Figure 5: Comparison diagram of traffic scene detection

5 Conclusion

This article proposes a joint and unified multi-task perception network for autonomous driving: GDMNet, which can achieve drivable area segmentation, lane detection, and traffic target detection in a single detection process based on images captured by cameras on roads, overcoming the resource consumption issue of single-task networks. The GDMNet is based on an encoder-decoder structure, where the decoder is used for feature extraction, utilizing GELAN instead of CSPNet for feature extraction and gradient propagation. In the decoding stage, detection heads are designed separately for drivable areas and lane lines based on their characteristics. The drivable area detection head employs DySample to enlarge the feature map, while the lane detection head integrates previous feature maps and undergoes fusion processing using FMN. The MPDIoU is employed to accurately calculate the matching degree between traffic target boxes and detection boxes, facilitating model adjustment and convergence. Experimental results on the BDD100K dataset demonstrate that the detection accuracy is significantly better than other models. Compared to baseline models, the GDMNet achieves higher accuracy and better robustness, performing well even in adverse weather conditions or low light situations such as nighttime. In future work, we aim to lightweight the model without compromising accuracy, enabling smooth operation of the model in autonomous vehicles with limited computing resources.

Acknowledgement: Thanks to the infrared dataset provided by Berkley Deep Drive Project Team at the University of California.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Yunxiang Liu, Haili Ma; data collection: Haili Ma, Jianlin Zhu; analysis and interpretation of results: Haili Ma, Qiangbo Zhang; draft manuscript preparation: Yunxiang Liu, Haili Ma. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available at http://bdd-data.berkeley.edu/ (accessed on 20 September 2023).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Y. Qian, J. M. Dolan, and M. Yang, "DLT-Net: Joint detection of drivable areas, lane lines, and traffic objects," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4670–4679, Nov. 2020. doi: 10.1109/TITS.2019.2943777.

[2] D. Wu *et al.*, "YOLOP: You only look once for panoptic driving perception," *Mach. Intell. Res.*, vol. 19, no. 6, pp. 550–562, 2022. doi: 10.1007/s11633-022-1339-y.

[3] 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.

[4] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," Jun. 07, 2016. doi: 10.48550/arXiv.1606.02147.

[5] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network'," in *2017 IEEE Conf. on Comput. Vis. and Pattern Recognit. (CVPR)*, Honolulu, HI, USA, IEEE, Jul. 2017, pp. 6230–6239. doi: 10.1109/CVPR.2017.660.

[6] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time joint semantic reasoning for autonomous driving," May 08, 2018. doi: 10.48550/arXiv.1612.07695.

[7] D. Vu, B. Ngo, and H. Phan, "HybridNets: End-to-end perception network," Mar. 16, 2022. doi: 10.48550/arXiv.2203.09035.

[8] B. C. Şenel, M. Mouchet, J. Cappos, O. Fourmaux, T. Friedman and R. McGeer, "EdgeNet: A multi-tenant and multi-provider edge cloud," in *Proc. of the 4th Int. Workshop on Edge Syst., Anal. and Netw.*, New York, NY, USA, Association for Computing Machinery, Apr. 2021, pp. 49–54. doi: 10.1145/3434770.3459737.

[9] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection CNNs by self attention distillation," presented at the Proc. of the IEEE/CVF Int. Conf. on Comput. Vis. Seoul, Republic of Korea, 2019, pp. 1013–1021. Accessed: Apr. 30, 2024. [Online]. Available: https://openaccess.thecvf.com/content_ICCV_2019/html/Hou_Learning_Lightweight_Lane_Detection_CNNs_by_Self_Attention_Distillation_ICCV_2019_paper.html.

[10] L. Zhang, F. Jiang, J. Yang, B. Kong, and A. Hussain, "A real-time lane detection network using two-directional separation attention," *Comput.-Aided Civil Eng.*, vol. 39, no. 1, pp. 86–101, Dec. 2023. doi: 10.1111/mice.13051.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Jan. 06, 2016. doi: 10.48550/arXiv.1506.01497.

[12] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," Apr. 08, 2018. doi: 10.48550/arXiv.1804.02767.

[13] T. Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conf. on Comput. Vis. and Pattern Recognit. (CVPR)*, Honolulu, HI, IEEE, Jul. 2017, pp. 936–944. doi: 10.1109/CVPR.2017.106.

[14] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," Apr. 22, 2020. doi: 10.48550/arXiv.2004.10934.

[15] X. Dong, S. Yan, and C. Duan, "A lightweight vehicles detection network model based on YOLOv5," *Eng. Appl. Artif. Intell.*, vol. 113, pp. 104914, Aug. 2022. doi: 10.1016/j.engappai.2022.104914.

[16] C. Y. Wang, I. H. Yeh, and H. Y. M. Liao, "YOLOv9: Learning what you want to learn using programmable gradient information," Feb. 21, 2024. doi: 10.48550/arXiv.2402.13616.

[17] H. Qi, P. Shi, Z. Liu, and A. Yang, "TSF: Two-stage sequential fusion for 3D object detection," *IEEE Sens. J.*, vol. 22, no. 12, pp. 12163–12172, Jun. 2022. doi: 10.1109/JSEN.2022.3175192.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015. doi: 10.1109/TPAMI.2015.2389824.

[19] C. Y. Wang, H. Y. Mark Liao, Y. H. Wu, P. Y. Chen, J. W. Hsieh and I. H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN'," in *2020 IEEE/CVF Conf. on Comput. Vis. and Pattern Recognit. Workshops (CVPRW)*, Seattle, WA, USA, IEEE, Jun. 2020, pp. 1571–1580. doi: 10.1109/CVPRW50498.2020.00203.

[20] W. Liu, H. Lu, H. Fu, and Z. Cao, "Learning to upsample by learning to sample," Aug. 29, 2023. doi: 10.48550/arXiv.2308.15085.

[21] J. Yang, C. Li, X. Dai, and J. Gao, "Focal modulation networks," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 4203–4217, Dec. 2022.

[22] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," presented at the Proc. of the IEEE Conf. on Comput. Vis. and Pattern Recognit. Salt Lake City, UT, USA, 2018, pp. 8759–8768. Accessed: Oct. 01, 2023. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Liu_Path_Aggregation_Network_CVPR_2018_paper.html

[23] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," presented at the Proc. of the IEEE Int. Conf. on Comput. Vis. Venice, Italy, 2017, pp. 2980–2988. Accessed: Apr. 30, 2024. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Lin_Focal_Loss_for_ICCV_2017_paper.html

[24] S. Ma and X. Yong, "MPDIoU: A loss for efficient and accurate bounding box regression," Accessed: Nov. 02, 2023, Jul. 14, 2023. [Online]. Available: http://arxiv.org/abs/2307.07662

[25] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems*. Montreal, Canada: Curran Associates, Inc., 2018. Accessed: Apr. 30, 2024. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/hash/f2925f97bc13ad2852a7a551802feea0-Abstract.html

[26] F. Yu *et al.*, "BDD100K: A diverse driving dataset for heterogeneous multitask learning," presented at the Proc. of the IEEE/CVF Conf. on Comput. Vis. and Pattern Recognit. Seattle, Washington, DC, USA, 2020, pp. 2636–2645. Accessed: Apr. 30, 2024. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2020/html/Yu_BDD100K_A_Diverse_Driving_Dataset_for_Heterogeneous_Multitask_Learning_CVPR_2020_paper.html

[27] S. Sun, B. Mo, J. Xu, D. Li, J. Zhao and S. Han, "Multi-YOLOv8: An infrared moving small object detection model based on YOLOv8 for air vehicle," *Neurocomputing*, vol. 588, pp. 127685, Jul. 2024. doi: 10.1016/j.neucom.2024.127685.

[28] Y. Rao, W. Zhao, Y. Tang, J. Zhou, S. N. Lim and J. Lu, "HorNet: Efficient high-order spatial interactions with recursive gated convolutions," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 10353–10366, Dec. 2022.

[29] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017. doi: 10.1109/TPAMI.2016.2644615.

[30] G. C. Niu and X. N. Wang, "A multi-task traffic scene detection model based on cross-attention," (in Chinese), *J. Beijing Univ. Aeronautics and Astronautics*, vol. 48, pp. 1491–1499, 2022. doi: 10.13700/j.bh.1001-5965.2022.0610.

[31] A. Parashar *et al.*, "SCNN: An accelerator for compressed-sparse convolutional neural networks," *ACM SIGARCH Comput. Archit. News.*, vol. 45, no. 2, pp. 27–40, Jun. 2017. doi: 10.1145/3140659.3080254.

[32] Z. Wang, W. Ren, and Q. Qiu, "LaneNet: Real-time lane detection networks for autonomous driving," Jul. 04, 2018. doi: 10.48550/arXiv.1807.01726.