

Article

A Lightweight Real-Time Debris Flow Detection Method Based on RF-DETR

Zhen Hu, Ou Ou *, Fuming Ma and Jiabao Zhao

College of Computer Science and Cyber Security (Pilot Software College), Chengdu University of Technology, Chengdu 610059, China; huzhen@stu.cdut.edu.cn (Z.H.); mafuming@stu.cdut.edu.cn (F.M.); zhaojiabao@stu.cdut.edu.cn (J.Z.)

* Correspondence: ouou@cdut.edu.cn

Abstract

As frequent and highly destructive geologic hazards, debris flows necessitate effective monitoring alongside rapid and accurate detection to support disaster prevention and mitigate losses of life and property. Current detection technologies, however, are often limited by high false alarm rates, insufficient accuracy, considerable model complexity that complicates deployment, and the elevated costs of contact-based detection. To overcome these limitations, this paper introduces a lightweight real-time debris flow detection model based on the Roboflow Detection Transformer (RF-DETR). First, we constructed a dataset of realistic debris flow scenarios including debris flow disaster events worldwide. Based on this, the lightweight vision transformer model EfficientFormerV2 is adopted as the backbone. By employing a dimension-consistent architecture, the model avoids the frequent switching between 4D and 3D features found in traditional Vision Transformers (ViTs), thereby reducing a significant number of inefficient operations. Additionally, we optimized the multiscale projection layer by removing downsampling and feature aggregation operations, which reduces redundant computations and improves feature extraction efficiency. Furthermore, the introduction of the Efficient Intersection over Union (EIoU) loss function achieves faster convergence and improved detection accuracy for debris flows. Ablation studies performed on our debris flow dataset demonstrate that the improved model reduces parameters by 54.2% and computational load by 36.4% while ensuring acceptable losses in detection accuracy and latency. This significant reduction in model size and complexity achieves effective lightweighting, fulfilling the requirements for deployment on edge devices and enabling real-time debris flow detection.

Keywords: debris flow; real-time object detection; model lightweighting; loss function; DETR



Academic Editor: Silvia Liberata Ullo

Received: 7 May 2026

Revised: 3 July 2026

Accepted: 5 July 2026

Published: 8 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

A debris flow refers to a special type of flood found in mountainous valleys, triggered by water sources such as heavy rain or snowmelt, and containing large amounts of silt, sand, and rocks. It is highly destructive [1]. According to the China National Disaster Reduction Network, geological disasters, primarily floods and debris flows, accounted for the most severe losses among all natural disasters in China for the year 2025. These events resulted in 568 fatalities and missing persons, alongside direct economic losses of 166.576 billion yuan, representing 74% and 69% of the respective totals across all disaster types [2]. Debris flows pose a serious threat to ecological environments, human lives, and

property and entail substantial economic losses. Consequently, effective monitoring and rapid identification of debris flows during their occurrence are critically important.

Traditional debris flow monitoring technologies, represented by sensor technology, have long played a vital role in debris flow detection. Sensor-based detection methods typically involve deploying field instruments such as debris flow level gauges, rain gauges, and infrasound sensors to collect relevant parameters in real time for subsequent simulation and analysis. However, traditional debris flow monitoring technologies rely on multi-sensor fusion [3] or experimental simulations [4]. Moreover, the complex topography of many debris flow-prone areas makes large-scale sensor deployment infeasible. Additionally, traditional contact-based monitoring methods are costly and highly susceptible to degradation under adverse weather. Consequently, various non-contact debris flow detection methods have been developed, including infrasound-based and real-time video-based approaches, which enable remote debris flow identification. In recent years, with the rapid development of deep learning, Leng [5] and others innovatively proposed an approach based on an improved LeNet-5 network, utilizing image recognition methods to identify the spectral patterns of subsonic signals from debris flows. Nevertheless, while subsonic-based debris flow monitoring systems can provide event alarms, they cannot track the movement of the debris flow in real time or capture visual information on its development [6,7]. Building on these facts, Zhou et al. [8] and Hu et al. [9] proposed modified You Only Look Once v8 (YOLOv8) and Temporal Segment Network (TSN) models, respectively, for real-time debris flow video detection. These models are applicable in complex debris flow scenarios and partially mitigate the limitations of contact-based methods. However, they suffer from low precision and recall. Additionally, video streams of debris flow events generate massive data volumes during transmission for identification. Furthermore, debris flows often occur under severe weather conditions, which may lead to delays or network interruptions. Therefore, model lightweighting is necessary to handle large volumes of video data and satisfy real-time requirements. Striking a balance between the lightweighting needs of real-time detection models and model performance remains a critical challenge in the field of real-time debris flow detection.

To address challenges such as high model parameter counts, complexity and high demands for device computing power, Zhao et al. [10] proposed Frequency Regularization to restrict the non-zero elements of the network parameters in the frequency domain to create lightweight neural networks. This paper adopts another effective approach to achieve model lightweighting by replacing the backbone and simplifying the commonly used projection layer. Consequently, we propose a lightweight real-time debris flow detection method based on the DETR object detection framework. While meeting the recall, precision, latency, and real-time requirements of debris flow video detection, the method introduces a series of lightweight modifications and optimizations to the current state-of-the-art (SOTA), end-to-end, real-time object detection model, RF-DETR.

2. Dataset Construction

2.1. Data Sources

The data used in this study primarily comes from two major sources: The first category of data is sourced from various publicly available online platforms, including public geological hazard databases, open-source data platforms, government open data resources, and public video platforms. These debris flow data cover debris flow events across different regions and time periods worldwide, ensuring a high degree of diversity in data sources. The second category of data consists of video and image data collected by video surveillance systems deployed in the field in areas prone to debris flows, ensuring the objectivity and traceability of the data sources. Most of dataset can be found

at <https://github.com/chronosphere/Debris-flow-open-dataset> (accessed on 3 July 2026). The primary sources of debris flow data are shown in Table 1.

Table 1. Overview of primary sources of debris flow data.

Data Source	Type	Description	Source
Video platforms	Video	Debris flow video records	BiliBili, TikTok, YouTube
Kaggle open datasets	Image	Flood images	https://www.kaggle.com/datasets/naiyakhalid/flood-prediction-dataset (accessed on 3 July 2026)
GitHub	Image	Visible-light images of debris flow	https://github.com/anzhi1211/debris-flow-automatic-annotation-dataset (accessed on 3 July 2026)
Field data	Image and video	Data collected on-site	Field data collection in debris flow-prone areas

2.2. Data Preprocessing

- Due to inconsistent data collection formats, there are issues with varying video clarity and frame rates of the collected debris flow video data, and the videos contain subtitles, watermarks, and other interfering information. Therefore, relevant tools were used to regenerate the frame rate and remove subtitles and watermarks from the raw debris flow videos. This standardizes the frame rate of the debris flow videos and reduces the impact of irrelevant interference factors in subsequent dataset annotation and model optimization studies.
- After the aforementioned processing, video frames were extracted from the debris flow video data to construct a debris flow dataset, with a frame extraction interval of 1 s to balance data quality and reduce the impact of data overlap [8]. The resulting image set was manually screened to remove images unrelated to debris flows as well as blurry or distorted images.

Following this preprocessing, the dataset was optimized to ensure that critical details of the debris flows are preserved without introducing excessive redundant features, which could lead to a surge in data volume, increased training time, and a higher risk of overfitting.

2.3. Data Annotation

Analysis of the compiled debris flow image dataset reveals a strong correlation between the characteristics of different debris flow types and their material composition. Therefore, based on the material composition of the collected data, the debris flows are classified into mudslide, mud–rock, and water–rock types [11].

- Mudslide type: primarily composed of silt and clay, with uniform grain size;
- Mud–Rock type: may contain clay, silt, sand, gravel, cobbles, and boulders of various grain sizes, with highly uneven distribution;
- Water–Rock type: contains very little silt and clay, with highly uneven grain size.

Significantly, flash floods and certain types of debris flows share some similarities in material composition, which can interfere with the accuracy of the model. At the same time, debris flow components such as stones and mud clumps serve as key criteria for distinguishing between different types of debris flows and significantly influence the feature extraction process. Therefore, additional labels for flash floods, stones, and mud clumps were added to the data annotation.

2.4. Data Augmentation and Dataset Split

The original dataset is limited and fails to capture the full spectrum of characteristics inherent in real-world debris flow disasters. Appropriately expanding the dataset through

data augmentation techniques enriches the sample repository and improves model accuracy in debris flow recognition. At the same time, it mitigates overfitting during training, enhances sample diversity, and ensures model robustness under real-world conditions. Common data augmentation methods include random flipping, random rotation, random cropping, random scaling, random shifting, random Gaussian noise, and brightness adjustment. However, data augmentation may introduce bias into the model by assuming prior knowledge of the dataset's properties [12]. Certain transformations, such as vertical flipping, random flipping and random cropping, sometimes do not align with actual debris flow scenarios and thus can negatively impact the training of real-time debris flow video detection models. Consequently, only horizontal flipping, random shifting, random rotation, brightness adjustment and Gaussian noise were applied in this study. Following data augmentation, the number of image samples for training the debris flow detection model increased to 8534. The details of the adopted data augmentation strategies are presented in Table 2.

Table 2. Overview of data augmentation methods.

Method	Param	Value
Horizontal flipping	None	None
Random shifting	None	None
Random rotation	Angle of rotation	-8° – $+8^{\circ}$
Brightness adjustment	Brightness	0.55–1
Gaussian noise	Mean and variance	Mean is 1 and variance is 0.01

To ensure dataset independence and robust model evaluation, the debris flow dataset is partitioned into training, validation, and test subsets in a ratio of 7:1:2. Among these, the training and validation sets are constructed using Stratified Sampling, meaning the proportion of samples per label in both sets matches that of the overall dataset. Additionally, the test dataset remains relatively independent of the training and validation sets, thereby reducing data overlap. The distribution of label categories is listed in Table 3.

Table 3. Proportion of samples by debris flow type.

Label Category	Count			Percentage/%		
	Train	Valid	Test	Train	Valid	Test
Mudslide type	1428	214	438	24.6	24.6	23.7
Mud–rock type	2353	359	398	40.5	41.2	21.5
Water–rock type	300	41	454	5.2	4.7	24.6
Flash flood	273	41	191	4.7	4.7	10.3
Negative samples	1461	216	367	25.1	24.8	19.9
Total	5815	871	1848	100	100	100

3. Improved Method Based on RF-DETR

3.1. RF-DETR—End-to-End, Real-Time Object Detection Model

RF-DETR is a real-time, end-to-end object detection model based on deep learning and computer vision, jointly proposed by Roboflow and research teams in Carnegie Mellon University [12]. Building upon the lightweight detection transformer (LW-DETR) [13], RF-DETR enhances generalization capabilities across different detection targets by simplifying its architecture and optimizing the training process. It chooses self-distillation with no labels ViT v2 (DINOv2) as its backbone. Specifically, it replaces the context autoencoder v2 (CAEv2) [14] backbone in LW-DETR with DINOv2 [15] to achieve superior object detection performance. Furthermore, RF-DETR employs a weight-sharing neural architecture

search (NAS) [16] for network design, significantly reducing the latency introduced by architectural simplification and backbone replacement. The model achieves state-of-the-art (SOTA) performance on the COCO and RF100-VL object detection benchmarks, achieving an optimal balance between latency and accuracy.

The RF-DETR architecture integrates a deformable cross-attention module [17], which focuses only on a limited set of key sampling points around reference points on the feature map and introduces a set of two-dimensional offsets. These offsets are learned from the query elements themselves via linear projection. Utilizing these learnable offsets, the sampling points can adapt dynamically to the shape of the target object. This design provides strong adaptability to object deformation, making it particularly suitable for object detection tasks involving targets with significant spatial deformation and dynamic characteristics, such as debris flows and flash floods. Furthermore, although the RF-DETR model exhibits robust feature extraction and detection performance, it currently suffers from drawbacks such as a high parameter count and substantial model complexity, which pose significant challenges for deployment on resource-constrained edge devices. Consequently, achieving the optimal balance between accuracy and a lightweight design for real-time debris flow video detection models remains a major research challenge.

3.2. Improvements to the Lightweight Backbone

RF-DETR is a real-time object detection model based on a transformer [18]. Although its backbone DINOv2, demonstrates outstanding performance on various standard datasets, it incorporates more encoder layers and exhibits higher model complexity and computational costs than other backbones. These factors significantly limit the practical applicability of RF-DETR in resource-constrained environments.

To address these limitations, this study replaces the original RF-DETR DINOv2 backbone with a lightweight alternative, EfficientFormerV2 [19,20]. EfficientFormerV2 utilizes a dimension-consistent architecture design by dividing the network into MetaBlock 4D (MB 4D), which employs a convolutional network style, and MetaBlock 3D (MB 3D), which performs linear projection and attention calculations based on 3D tensors. This design circumvents the frequent switching between 4D and 3D features found in traditional ViT architectures and significantly reduces inefficient reshaping operations. Furthermore, while ViT architectures like DINOv2 employ Patch Embedding with large-kernel convolutions, such operations are poorly supported by most mobile and edge device compilers [19], making them difficult to optimize and accelerate. In contrast, EfficientFormerV2 replaces Patch Embedding with a Convolution Stem [21], which uses two 3×3 small convolutions for downsampling instead of a single large-kernel convolution. This modification significantly reduces inference latency when deployed on edge devices. The improved network architecture is shown in Figure 1.

This paper also optimizes the projection layer of RF-DETR to better adapt to the lightweight backbone EfficientFormerV2. Specifically, for the four-stage, multiscale feature maps output by EfficientFormerV2, we apply sequential 1×1 convolutions, batch normalization, and the ReLU activation function to project them into a 256-dimensional space, allowing for subsequent processing by the RF-DETR decoder. In the original model, hyperparameters determine how features are extracted from the outputs of multiple transformer blocks. Subsequently, a multiscale projection layer performs upsampling, downsampling, or hold operations based on predefined scaling ratios for different feature layers, followed by feature fusion via the C2f module [22] to project the features into a 256-dimensional space. In contrast, the optimized projection layer omits complex sampling, fusion, and the C2f module entirely. It utilizes a straightforward sequence of 1×1 convolution, BatchNorm, and a rectified linear unit (ReLU), thereby yielding a more streamlined and

lightweight architecture. The implementation differences between the projection layers for the EfficientFormerV2 and DINOv2 backbones within RF-DETR are illustrated in Figure 2.

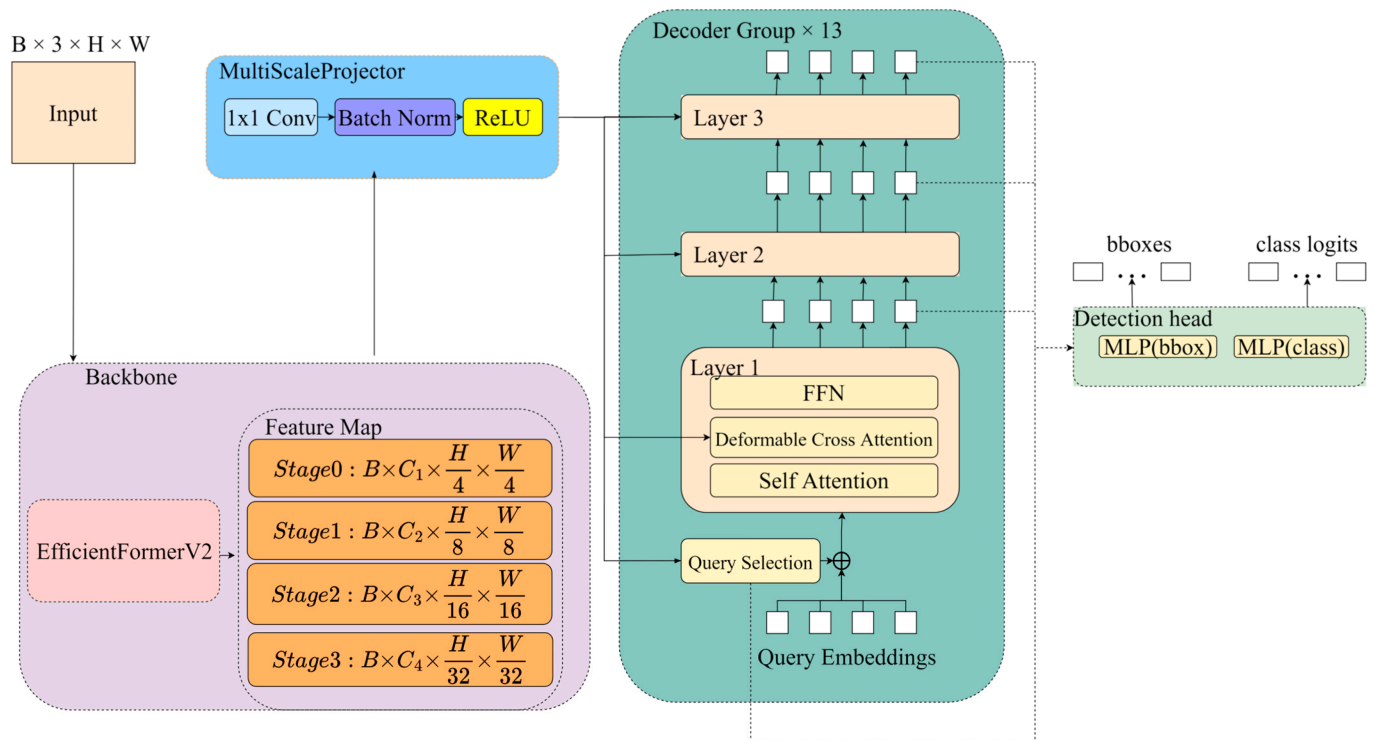


Figure 1. Improved RF-DETR network architecture.

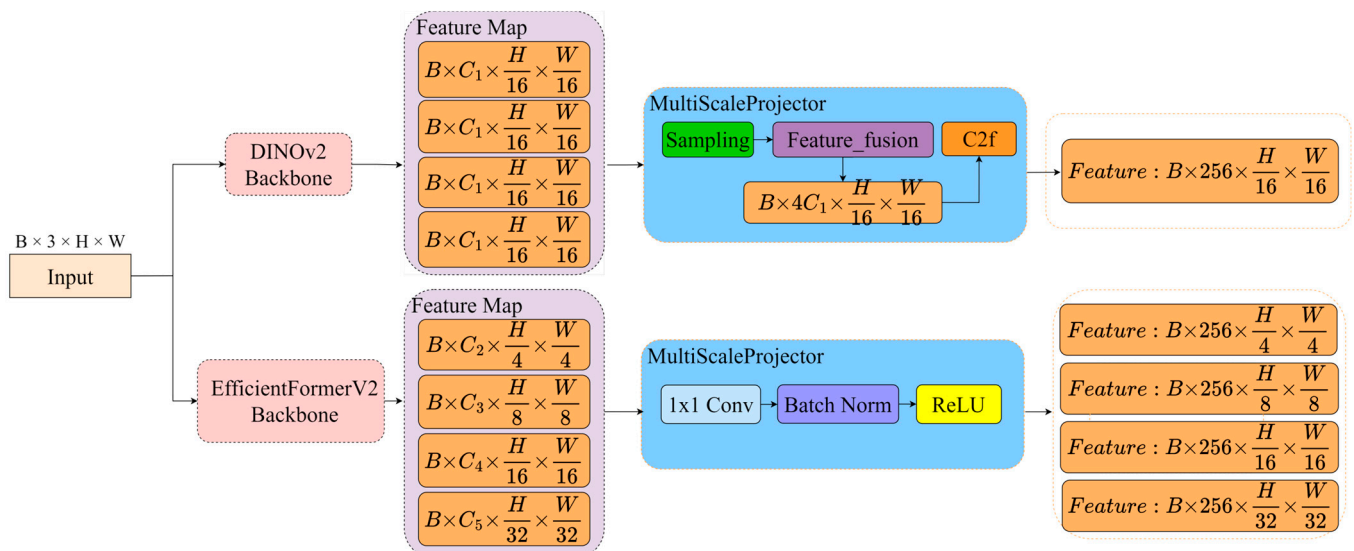


Figure 2. Differences between the DINOv2 and EfficientFormerV2 projection layers.

3.3. Improvements to the Loss Function

The loss function of the original RF-DETR model consists of three parts: classification loss, bounding box regression loss, and Generalized Intersection over Union (GIoU) loss [23]. The classification loss employs Sigmoid Focal Loss (SFL) [24], which mitigates class imbalance issues and contributes to training stability. For bounding box regression, L1 loss [25] is applied due to its robustness to outliers, making it suitable for real-world training scenarios. The GIoU loss measures the spatial overlap between predicted and ground-truth bounding boxes. It better reflects the positional relationship between boxes

than traditional IoU, thereby providing better guidance to the model to optimize parameters and improve prediction accuracy.

This paper focuses solely on modifications to the GIoU loss to improve the model's recall and precision in real-time debris flow detection. The GIoU loss is defined as

$$IoU = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

$$GIoU = IoU - \frac{|C - (A \cup B)|}{|C|} \quad (2)$$

where $A, B \subseteq S \in R^n$ represent two arbitrary bounding boxes. C is the minimum bounding rectangle containing both A and B , satisfying $C \subseteq S \in R^n$. The GIoU loss remains valid even when $|A \cap B| = 0$, but it has the following two drawbacks:

When $|A \cap B| = 0$, the GIoU loss tends to increase the area of the predicted bounding box during optimization, causing it to overlap with the ground-truth bounding box and thereby reduce the loss, as shown in Figure 3. This does not align with our expectation that the loss should be reduced by having the predicted bounding box converge toward the ground-truth bounding box.

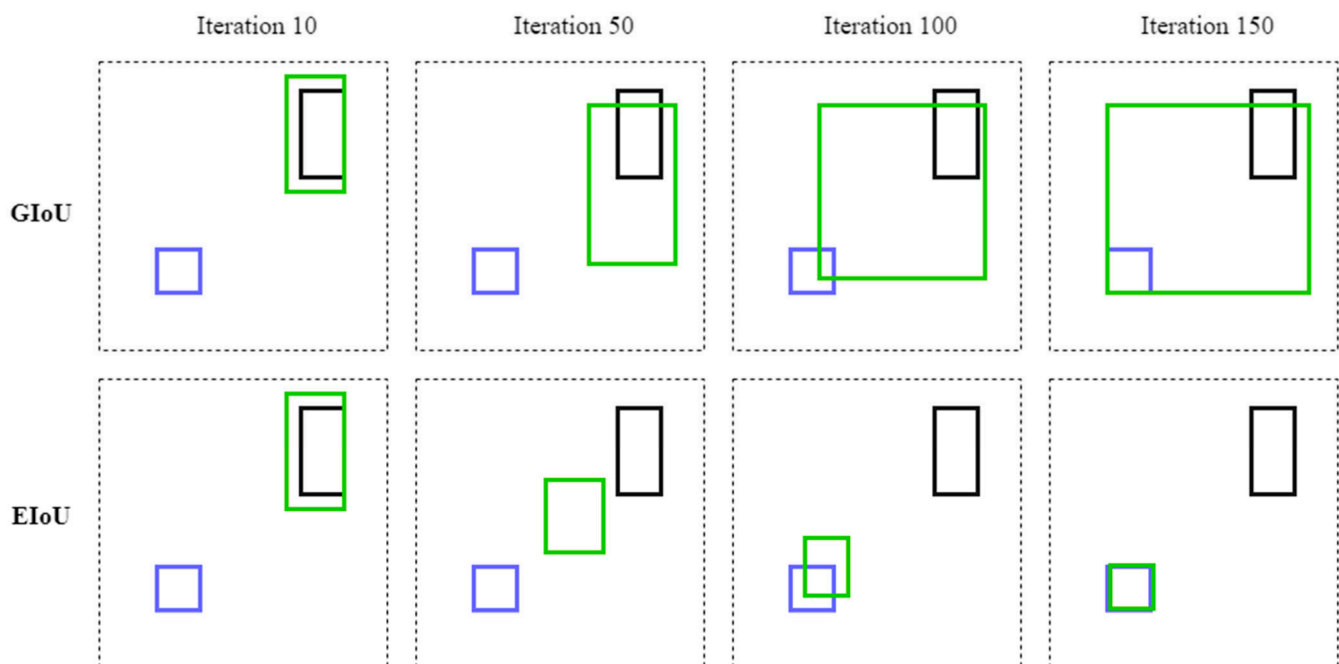


Figure 3. Convergence process of the GIoU and EIoU loss functions.

When $|A \cap B| > 0$, the region of $|C - A \cup B|$ is typically very small (this value is zero when $B \subseteq A$ or $A \subseteq B$). In this case, the GIoU loss degenerates into the standard IoU loss, resulting in very slow convergence, as shown in Figure 3.

To address the limitations of the GIoU loss, this paper employs the EIoU [26] in place of GIoU to represent the degree of spatial overlap between predicted and ground-truth bounding boxes. EIoU is a further optimization built upon a series of improvements to GIoU, including Distance Intersection over Union (DIoU) [27] and Complete Intersection over Union (CIoU) [27]. By integrating shape features, EIoU enhances the similarity metric between bounding boxes, thereby more accurately describing their spatial relationships, improving the localization accuracy of target objects, and further enhancing the performance of object detection algorithms.

The EIoU loss is defined as

$$L_{EIou} = L_{IoU} + L_{dis} + L_{asp} \\ = 1 - IoU + \frac{\rho^2(b, b^{gt})}{(w^c)^2 + (h^c)^2} + \frac{\rho^2(w, w^{gt})}{(w^c)^2} + \frac{\rho^2(h, h^{gt})}{(h^c)^2} \quad (3)$$

EIoU divides the loss function into three components: IoU loss L_{IoU} , distance loss L_{dis} , and orientation loss L_{asp} . Here, w^c and h^c are the width and height of the minimum bounding rectangle covering the predicted bounding box and the ground-truth bounding box, respectively, while w , h , w^{gt} , and h^{gt} represent the width and height of the predicted bounding box and the ground-truth bounding box, respectively. b and b^{gt} denote the center points of the predicted bounding box and the ground-truth bounding box, respectively. $\rho(\cdot) = \|b - b^{gt}\|_2$ represents the Euclidean distance. In this way, the characteristics of the CIoU loss are preserved, and the EIoU loss directly minimizes the difference in width and height between the ground-truth and predicted bounding boxes, thereby achieving faster convergence and better localization results.

4. Experiments and Analysis

4.1. Environment and Parameter Settings

This paper utilized a model training platform to conduct the relevant experiments; the experimental environment configuration is detailed in Table 4. Key training parameters used in the experiments are shown in Table 5.

Table 4. Experimental environment configuration table.

Name	Detailed Configuration
Operating System	Ubuntu 22.04.4 LTS (x86_64)
CPU	AMD EPYC 7K62, 2-core
GPU	NVIDIA GeForce RTX 4060 Ti, 16 GB
Memory	377 GB
Deep Learning Framework	PyTorch
	Python v3.10.19
	Torch v2.2.0+cu118
Integrated Development Environment	TorchAudio v2.2.0+cu118
	TorchVision v0.17.0+cu118

Table 5. Key training parameters of real-time debris flow detection model. Specially, RF-DETR use both grad_accum_steps and batch_size to calculate total batch size. Thus, we set grad_accum_steps and batch_size to 4 in RF-DETR. Batch size of the others is set to 16.

Training Parameters	Value
Epochs	150
Learning Rate (lr)	0.0001
Batch size	16
Random seed	42
Pretrain_weights	None
Automatic Mixed Precision (amp)	True
Optimizer	AdamW

4.2. Results and Data Analysis

4.2.1. Evaluation Metrics

Typically, key metrics used to evaluate model performance in detection tasks include precision, recall, AP50 and AP50:0.5:95, the number of parameters (Params), latency, and

floating-point operations (FLOPs). Given the destructive nature and unique characteristics of debris flow disasters, the most critical evaluation metric for debris flow detection models is recall. Higher recall indicates a lower probability of missed debris flow events. Precision is the second most important metric. Higher precision corresponds to more-accurate classification and a lower likelihood of false positives. Other evaluation metrics are the same as those used in typical detection tasks. To ensure consistency and fairness, all models were trained and validated under the same experimental environment, and all metrics were derived from the debris flow test set.

4.2.2. Comparison of Different Real-Time Object Detectors

This paper evaluates the performance of the RF-DETR model and other mainstream real-time object detection models on the debris flow dataset, demonstrating that the RF-DETR model outperforms other mainstream real-time object detection models in metrics such as recall and precision. The experimental results are shown in Table 6.

Table 6. Performance comparison of different real-time object detectors. All latency, Params and GFLOPs are measured on the same PC, with warmup set to 10, batch_size set to 1 and image size to 224×224 . Specially, RT-DETRv2-S uses 640×640 and EfficientDet-D0 uses 512×512 because 224×224 does not satisfy their input.

Model	Precision	Recall	AP50	AP50:0.5:95	Latency	Params	GFLOPs
RFDETRSmall [12] (baseline)	91.3	90.3	91.5	75.5	31.4 ms	32.1 M	12.9
YOLOv8n [22]	86.7	80.3	83.5	61.8	23.5 ms	2.7 M	0.9
YOLOv8s [22]	89.2	83.9	88.3	65.6	22 ms	9.8 M	2.9
YOLOv9s [28]	89.1	81.2	86.3	63.7	35.5 ms	6.3 M	2.8
YOLO11s [29]	88.7	81.8	86.3	58.7	24.3 ms	9.4 M	2.6
YOLO11m [29]	90.5	83.3	88.4	67	28.6 ms	20.1 M	8.4
YOLO-NAS-S [30]	81.6	76.9	79.1	54.2	32.4 ms	19 M	4.1
EfficientDet-D0 [31]	68.1	84.7	66.2	52.3	45.7 ms	3.9 M	4.9
RT-DETRv2-S [32,33]	90	86.3	90.9	75.7	45.8 ms	20.1 M	61.2

The fill color in the table indicates the overall optimal performance.

As shown in Table 6, the RF-DETR model significantly outperforms the YOLO series models on the debris flow dataset in terms of metrics including recall, precision, AP50 and AP50:0.5:95 and lags in latency, parameters and FLOPs. Compared with RT-DETRv2-S, RF-DETR performs slightly better than RT-DETRv2 in all metrics except AP50:0.5:95 and model parameters. These results highlight the significant advantages of the RF-DETR model over other mainstream real-time object detection models in the field of real-time debris flow detection.

4.2.3. Comparison of Different Backbones

To evaluate the impact of the lightweight optimization experiments on the model, we introduced a lightweight modification to RF-DETR. We replaced the original RF-DETR backbone DINOv2 with representative lightweight backbones of different types including EfficientFormerV2 and “Re-parameterization convolutions in a ViT-like MetaFormer structure” (RepViT). The experimental results are shown in Table 7.

As presented in Table 7, employing EfficientFormerV2 as the backbone yields a 5.4% reduction in recall, a 2.7% improvement in precision, a 3.5% decrease in AP50, and a 4% decline in AP50:0.5:95, along with substantial increases of 152.7% in GFLOPs and 54% in latency. This rise in GFLOPs is primarily attributed to the varying feature channels of EfficientFormerV2 (illustrated in Figure 2). Unlike the original DINOv2 backbone,

which outputs uniform feature maps, EfficientFormerV2 requires additional computational overhead in the MultiScaleProjector module to reconcile these dimensional differences. To counteract the significant increase in GFLOPs, we optimized the projection layer, achieving a notable reduction in parameters and FLOPs by 54.2% and 36.4%, respectively, with only a 20.4% increase in average latency and a 6.4% loss in recall. Moreover, compared with the RepViT backbone, EfficientFormerV2 exhibits superior performance across most evaluation metrics. The comparative analysis of different backbones within the RF-DETR framework validates the efficacy of substituting the DINOv2 backbone with EfficientFormerV2.

Table 7. Performance comparison of RF-DETR using different backbones. All Params are measured on same device, with warmup set to 10, batch_size set to 1 and image size to 224×224 . Latency is measured on our test dataset.

Backbone	Type	Projector Simplify	Precision	Recall	AP50	AP50:0.5:95	Latency	Params	GFLOPs
DINOv2 [15]	Attention	-	91.3	90.3	91.5	75.5	31.4 ms	32.1 M	12.9
RepViT [34]	Conv	-	93.9	82.9	87.5	70.1	40.3 ms	19.7 M	7.34
EfficientFormerV2 [20]	Hybrid	No	94	84.9	88	71.5	48.4 ms	17.4 M	32.6
		Yes	94.1	84.7	88.7	71.8	37.8 ms	14.7 M	8.21

The fill color in the table indicates the overall optimal performance.

4.2.4. Comparison of Different Loss Functions

To validate the effectiveness of the introduced EIoU loss function, this paper evaluates the performance of the RF-DETR model using various IoU loss functions, including the original GIoU [23], DIoU [27], CIoU [27], EIoU [26], and Scylla Intersection over Union (SIoU) [35]. The experimental results on the debris flow dataset are shown in Table 8.

Table 8. Comparison of RF-DETR performance using different loss functions. All Params and FPS are measured on same device, with warmup set to 10, batch_size set to 1 and image size to 224×224 . Latency is measured on our test dataset.

Loss Function	Precision	Recall	AP50	AP50:0.5:95	Latency	Params	GFLOPs
GIoU (baseline)	91.3	90.3	91.5	75.5	31.4 ms	32.1 M	12.9
DIoU	91.3	90.4	91.8	75.8	31.2 ms	32.1 M	12.9
CIoU	91.8	90.7	92.1	76.1	32.4 ms	32.1 M	12.9
EIoU	92.5	91.2	92.8	76.8	30.2 ms	32.1 M	12.9
SIoU	92.2	91	92.3	76.1	30.9 ms	32.1 M	12.9

The fill color in the table indicates the overall optimal performance.

A detailed analysis of Table 7 shows EIoU achieves the highest AP50:0.5:95, along with the highest recall and precision, indicating that the EIoU loss function demonstrates better localization and training performance than other loss functions. Meanwhile, GIoU, DIoU, CIoU, and EIoU represent successive improvements based on the same IoU loss calculation method, as illustrated by their consistent gains in recall and AP50:0.5:95. In contrast, the SIoU loss function is designed relatively independently and outperforms GIoU, DIoU, and CIoU on recall, precision, and mean precision, while performing slightly below EIoU. In addition, the results show that introducing the EIoU loss function has an almost negligible impact on latency, parameters and FLOPs. These results proved the superiority of the EIoU loss function.

4.2.5. Ablation Experiments

To verify that the modifications described in the previous experiments are indeed effective in the RF-DETR model, we designed a series of ablation experiments. The results of these experiments on the debris flow dataset are shown in Table 9.

Table 9. Ablation experiments for real-time debris flow detection. All Params and FPS are measured on same device, with warmup set to 10, batch_size set to 1 and image size to 224×224 . Latency is measured on our test dataset.

Backbone	Loss Function	Projector Simplify	Precision	Recall	AP50	AP50:0.5:95	Latency	Params	GFLOPs
DINOV2 (baseline)	GIoU	No	91.3	90.3	91.5	75.5	31.4 ms	32.1 M	12.9
	DIoU	No	91.3	90.4	91.8	75.8	31.2 ms	32.1 M	12.9
	CIoU	No	91.8	90.7	92.1	76.1	32.4 ms	32.1 M	12.9
	EIoU	No	92.5	91.2	92.8	76.8	30.2 ms	32.1 M	12.9
	SIoU	No	92.2	91	92.3	76.1	30.9 ms	32.1 M	12.9
EfficientFormerV2 (ours)	GIoU	No	94	84.9	88	71.5	48.4 ms	20.1 M	32.6
	GIoU	Yes	94.1	84.7	88.7	71.8	37.8 ms	14.7 M	8.21
	DIoU	Yes	94.1	84.6	88.7	72.2	38.8 ms	14.7 M	8.21
	CIoU	Yes	94.6	84.4	88.8	72.5	36 ms	14.7 M	8.21
	EIoU	Yes	95.2	85.1	88.9	72.6	36.4 ms	14.7 M	8.21
	SIoU	Yes	94.9	85.1	88.8	72.5	38.8 ms	14.7 M	8.21
YOLOv8s	-	-	89.2	83.9	88.3	65.6	22 ms	9.8 M	2.9

The fill color in the table indicates the overall optimal performance.

As shown in Table 9, replacing the DINOV2 backbone in RF-DETRSmall with EfficientFormerV2, switching the loss function to EIoU, and simplifying the projector yield a modified model with reduced performance on several key metrics: recall decreases by 5.2%, AP50 by 2.6%, and AP50:95 by 2.9%, while latency increases by 15.9%. Concurrently, our improved model reduces Params by 54.2% and GFLOPs by 36.4%. In summary, the proposed improvements effectively reduce model complexity and computational cost, incurring only minor losses in recall and precision. This optimized balance makes the model suitable for deployment on edge devices dedicated to real-time debris flow detection.

4.2.6. Benchmark on Edge Devices

To further validate the effectiveness of our model improvement, we deployed the model on an NVIDIA Jetson AGX Orin device to assess its performance and compared the effects of different adjustments. The environment is detailed in Table 10. The benchmark of our model is detailed in Table 11.

Table 10. Environment of NVIDIA Jetson AGX Orin.

Name	Detailed Configuration
Operating System	Ubuntu 22.04.4 LTS
Hardware	NVIDIA Jetson AGX Orin
Jtop	Version 4.2.8
Jetpack	6.0
L4T	36.3.0
CUDA	12.2.140
cuDNN	8.9.4.25
Memory	64 GB
Deep Learning Framework	PyTorch
	Python v3.10.19
Integrated Development Environment	torch 2.4.0a0+3bcc3cddb5.nv24.7 torchvision 0.19.0

As shown in the table, on the edge platform Jetson AGX Orin, our improved model exhibits performance consistent with the ablation study. Although the latency is 25.4 ms

higher than that of the baseline model and the FPS is reduced by 36.8%, it still satisfies the requirements for real-time debris flow detection.

Table 11. Benchmarks of our improved model. All Params and FPS are measured on same device, with warmup set to 10, batch_size set to 1 and image size to 224×224 . Latency is measured on our test dataset.

Backbone	Loss Function	Projector Simplify	Latency	Params	FPS
DINOv2 (baseline)	GIoU	No	43.3 ms	32.1 M	23.1
EfficientFormerV2 (ours)	GIoU	No	87.6 ms	20.1 M	11.4
	GIoU	Yes	68.8 ms	14.7 M	14.5
	EIoU	Yes	68.7 ms	14.7 M	14.6
YOLOv8n [22]	-	-	31 ms	2.7 M	34
YOLOv8s [22]	-	-	31.5 ms	9.8 M	33.2
YOLO11s [29]	-	-	35.6 ms	9.43 M	30.1

The fill color in the table indicates the overall optimal performance.

5. Discussion

Comprehensive experiments demonstrate that the lightweight modifications to the original baseline model, along with adjustments to the loss function and simplification of projection layer, enable the improved model to effectively balance performance metrics such as recall and parameters on the debris flow dataset. However, some issues still require further discussion and attention.

- In this study, all models were trained exclusively based on visible-light debris flow images. Using various data augmentation techniques, we simulated video data corresponding to special scenarios, such as those involving improperly installed monitoring equipment, unclear camera lenses, and overcast or rainy weather. However, the models are incapable of processing infrared images captured at night. As conventional cameras are largely ineffective after dark, supplementary lighting equipment is required, increasing both deployment complexity and cost. Furthermore, under low-visibility conditions, the incomplete characteristics of debris flow in video data inevitably lead to a significant reduction in the model's recognition rate for debris flow targets. Instances of both false positives and false negatives were also observed in our experiments. Consequently, the model proposed in this paper is subject to certain limitations in terms of applicable scenarios.
- This paper has demonstrated that the series of lightweight modifications applied to RF-DETR are reasonable and effective. However, we also introduce an increase in latency and decrease in recall and AP50:0.5:95. In future research, we will focus on the model's performance on mobile devices and NPUs, optimizing its architecture and algorithms with a specific focus on real-time inference performance, model conversion, quantization, and memory efficiency.
- This study has certain limitations. RF-DETR is a method based on NAS [16], and the DINOv2 backbone it uses is already an optimal architecture. If we were to replace it with the lightweight EfficientFormerV2 backbone, the overall architecture would need to be redesigned to ensure the resulting model achieves optimal performance. In our future research, we will consider optimizing the overall model architecture to further improve the model's performance and efficiency.

6. Conclusions

Detecting debris flows is a challenging problem due to the limitations of existing methods. Traditional contact-based methods are often costly and poorly suited to the

complex topography of many debris flow-prone areas. Non-contact methods such as infrasound-based and real-time video-based approaches struggle with real-time tracking and suffer from low precision and recall. To address these issues, this paper proposes and implements a lightweight real-time debris flow detection method based on RF-DETR. To validate our approach, we conducted a series of research, comparison and ablation experiments with reasonable and effective improvements. First, we constructed a dataset of realistic debris flow scenarios. This dataset encompasses debris flow disaster events from various regions and time periods worldwide, as well as field-collected debris flow video and image data, ensuring both data diversity and reliability. We then employed effective methods to filter out irrelevant factors that could interfere with the validation of this method. Second, we optimized the baseline RF-DETR model for lightweight deployment by replacing its backbone with the hybrid architecture EfficientFormerV2 and adapting its projection layer. Finally, to mitigate the slow convergence and inherent limitations of the traditional GloU loss function, we introduce EIou loss to comprehensively evaluate predicted and ground-truth bounding boxes, thereby better guiding model parameter optimization to enhance real-time debris flow detection performance.

In summary, we reduced the parameter count and computational load of RF-DETR significantly, lowering its complexity and providing a solid foundation for deployment on edge devices. This improvement came at the cost of a decrease in recall, precision, AP50 and AP50:0.5:95 and small increase in latency. The approach enables timely and effective detection through edge devices before debris flow disasters occur, offering a more accurate object detection solution for debris flow early warning. The optimized model is particularly suitable for debris flow real-time detection applications on embedded edge devices, demonstrating substantial practical value and broad application prospects.

Author Contributions: Conceptualization, O.O.; Methodology, Z.H.; Software, Z.H.; Validation, Z.H.; Data Curation, Z.H., F.M. and J.Z.; Writing—Original Draft Preparation, Z.H.; Writing—Review and Editing, O.O.; Supervision, O.O.; Project Administration, O.O. and Z.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Dataset is available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Flash Floods, Debris Flows and Their Response Measures. 2024. Available online: https://www.mem.gov.cn/xw/ztzl/2024/fxkhjxs2024/yjkp/202404/t20240419_485508.shtml (accessed on 19 April 2026).
- Overview of National Natural Disasters in 2025. 2026. Available online: https://www.mem.gov.cn/xw/yjglbgzdt/202601/t20260116_592135.shtml (accessed on 19 April 2026).
- Xiao, Y.; Guo, Y.; Wei, L. Design and Implementation of a Debris Flow Monitoring and Warning System in Southeast Tibet. *Acta Geol. Sichuan* **2024**, *44*, 708–715.
- Li, L.; Li, T.; He, Z.; Li, S.; Dong, J.; Wang, B. Monitoring and Early Warning of Landslide and Debris Flow Disaster Chain Risk Based on Experimental Simulation. *Bull. Soil Water Conserv.* **2024**, *44*, 167–175. [CrossRef]
- Leng, X.; Feng, L.; Ou, O.; Du, X.; Liu, D.; Tang, X. Debris Flow Infrasound Recognition Method Based on Improved LeNet-5 Network. *Sustainability* **2022**, *14*, 15925. [CrossRef]
- Hu, Y.; Yuan, L.; Ma, D.; Li, M. Research Progress on Debris Flow Infrasound Warning. *Adv. Earth Sci.* **2018**, *33*, 606–613.
- Liu, D.; Leng, X.; Wei, F.; Zhang, S.; Hong, Y. Visualized localization and tracking of debris flow movement based on infrasound monitoring. *Landslides* **2018**, *15*, 879–893. [CrossRef]
- Zhou, J.; Ju, N.; Zhang, Y.; Tian, H.; He, C. Debris Flow Tracking and Detection Method Research Via Video Image Analysis. *Adv. Earth Sci.* **2025**, *40*, 388–400. [CrossRef]
- Hu, M.; Liu, D.; Sang, X.; Zhang, S.; Chen, Q. Intelligent Identification Method of Debris Flow Scene Based on Camera Video Surveillance. *Comput. Mod.* **2024**, 41–46. [CrossRef]

10. Zhao, C.; Dong, G.; Zhang, S.; Tan, Z.; Basu, A. Frequency Regularization: Reducing Information Redundancy in Convolutional Neural Networks. *IEEE Access* **2023**, *11*, 106793–106802. [\[CrossRef\]](#)
11. T/CAGHP 006-2018; Specification of Geological Investigation for Debris Flow Stabilization. China Association of Geological Disaster Prevention: Beijing, China; University of Geosciences Press: Wuhan, China, 2018.
12. Isaac, R.; Peter, R.; Matvei, P.; Deva, R.; Neehar, P. RF-DETR: Neural Architecture Search for Real-Time Detection Transformers. In Proceedings of the 14th International Conference on Learning Representations (ICLR 2026), Rio de Janeiro, Brazil, 23–27 April 2026. [\[CrossRef\]](#)
13. Chen, Q.; Su, X.; Zhang, X.; Wang, J.; Chen, J.; Shen, Y.; Han, C.; Chen, Z.; Xu, W.; Li, F.; et al. LW-DETR: A Transformer Replacement to YOLO for Real-Time Detection. *arXiv* **2016**, arXiv:2406.03459. [\[CrossRef\]](#)
14. Zhang, X.; Chen, J.; Yuan, J.; Chen, Q.; Wang, J.; Wang, X.; Han, S.; Chen, X.; Pi, J.; Yao, K.; et al. CAE v2: Context Autoencoder with CLIP Latent Alignment. *Trans. Mach. Learn. Res.* **2023**. [\[CrossRef\]](#)
15. Oquab, M.; Darcet, T.; Moutakanni, T.; Vo, H.V.; Szafraniec, M.; Khalidov, V.; Fernandez, P.; Haziza, D.; Massa, F.; El-Nouby, A.; et al. DINOv2: Learning Robust Visual Features without Supervision. *Trans. Mach. Learn. Res.* **2024**, 1–31. [\[CrossRef\]](#)
16. Zoph, B.; Le, Q.V. Neural Architecture Search with Reinforcement Learning. In Proceedings of the 5th International Conference on Learning Representations (ICLR 2017), Toulon, France, 24–26 April 2017. [\[CrossRef\]](#)
17. Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In Proceedings of the 9th International Conference on Learning Representations (ICLR 2021), Online, 3–7 May 2021. [\[CrossRef\]](#)
18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All you Need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017. [\[CrossRef\]](#)
19. Li, Y.; Yuan, G.; Wen, Y.; Hu, E.; Evangelidis, G.; Tulyakov, S.; Wang, Y.; Ren, J. EfficientFormer: Vision Transformers at MobileNet Speed. In Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, LA, USA, 28 November–9 December 2022. [\[CrossRef\]](#)
20. Li, Y.; Hu, J.; Wen, Y.; Evangelidis, G.; Salahi, K.; Wang, Y.; Tulyakov, S.; Ren, J. Rethinking Vision Transformers for MobileNet Size and Speed. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023. [\[CrossRef\]](#)
21. Graham, B.; El-Nouby, A.; Touvron, H.; Stock, P.; Joulin, A.; J'egou, H.E.; Douze, M. LeViT: A Vision Transformer in ConvNet's Clothing for Faster Inference. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021. [\[CrossRef\]](#)
22. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLOv8. 2023. Available online: <https://docs.ultralytics.com/models/yolov8> (accessed on 3 July 2026).
23. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019. [\[CrossRef\]](#)
24. Lin, T.-Y.; Goyal, P.; Girshick, R.B.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017. [\[CrossRef\]](#)
25. Zhao, H.; Gallo, O.; Frosio, I.; Kautz, J. Loss Functions for Image Restoration with Neural Networks. *IEEE Trans. Comput. Imaging* **2017**, *3*, 47–57. [\[CrossRef\]](#)
26. Zhang, Y.F.; Ren, W.; Zhang, Z.; Jia, Z.; Wang, L.; Tan, T. Focal and efficient IOU loss for accurate bounding box regression. *Neurocomputing* **2022**, *506*, 146–157. [\[CrossRef\]](#)
27. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. In Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-20), New York, NY, USA, 7–12 February 2020. [\[CrossRef\]](#)
28. Wang, C.; Yeh, I.; Mark Liao, H. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. In Proceedings of the Computer Vision—ECCV 2024, Milan, Italy, 29 September–4 October 2024. [\[CrossRef\]](#)
29. Jocher, G.; Qiu, J. Ultralytics YOLO11, 2024. Available online: <https://docs.ultralytics.com/models/yolo11/> (accessed on 3 July 2026).
30. Aharon, S.; Louis-Dupont; Masad, O. Super-Gradients. 2021. Available online: <https://zenodo.org/records/7789328> (accessed on 3 July 2026).
31. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020), Seattle, WA, USA, 13–19 June 2020. [\[CrossRef\]](#)
32. Lv, W.; Zhao, Y.; Chang, Q.; Huang, K.; Wang, G.; Liu, Y. RT-DETRv2: Improved Baseline with Bag-of-Freebies for Real-Time Detection Transformer. *arXiv* **2024**, arXiv:2407.17140. [\[CrossRef\]](#)

33. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs Beat YOLOs on Real-time Object Detection. In Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024), Seattle, WA, USA, 16–22 June 2024. [\[CrossRef\]](#)
34. Wang, A.; Chen, H.; Lin, Z.; Han, J.; Ding, G. Rep ViT: Revisiting Mobile CNN From ViT Perspective. In Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024. [\[CrossRef\]](#)
35. Gevorgyan, Z. Siou Loss: More Powerful Learning for Bounding Box Regression. *arXiv* **2022**, arXiv:2205.12740. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.