International Journal of Geo-Information

MDPI

Article

Detecting Road Intersections from Crowdsourced Trajectory Data Based on Improved YOLOv5 Model

Yunfei Zhang [1,2], Gengbiao Tang [1,2,*] and Naisi Sun [1,2]

[1] National Key Laboratory of Green and Long-Life Road Engineering in Extreme Environment (Changsha), Changsha University of Science & Technology, Changsha 410114, China; zhang.yunfei@csust.edu.cn (Y.Z.); 23101020015@stu.csust.edu.cn (N.S.)
[2] School of Traffic & Transportation Engineering, Changsha University of Science & Technology, Changsha 410114, China
* Correspondence: tanggengbiao@stu.csust.edu.cn

Abstract: In recent years, the rapid development of autonomous driving and intelligent driver assistance has brought about urgent demands on high-precision road maps. However, traditional road map production methods mainly rely on professional survey technologies, such as remote sensing and mobile mapping, which suffer from high costs, object occlusions, and long updating cycles. In the era of ubiquitous mapping, crowdsourced trajectory data offer a new and low-cost data resource for the production and updating of high-precision road maps. Meanwhile, as key nodes in the transportation network, maintaining the currency and integrity of road intersection data is the primary task in enhancing map updates. In this paper, we propose a novel approach for detecting road intersections based on crowdsourced trajectory data by introducing an attention mechanism and modifying the loss function in the YOLOv5 model. The proposed method encompasses two key steps of training data preparation and improved YOLOv5s model construction. Multi-scale training processing is first adopted to prepare a rich and diverse sample dataset, including various kinds and different sizes of road intersections. Particularly to enhance the model's detection performance, we inserted convolutional attention mechanism modules into the original YOLOv5 and integrated other alternative confidence loss functions and localization loss functions. The experimental results demonstrate that the improved YOLOv5 model achieves detection accuracy, precision, and recall rates as high as 97.46%, 99.57%, and 97.87%, respectively, outperforming other object detection models.

Keywords: crowdsourced trajectory data; road intersections; YOLOv5 model; attention mechanism; loss function

check for updates

Citation: Zhang, Y.; Tang, G.; Sun, N. Detecting Road Intersections from Crowdsourced Trajectory Data Based on Improved YOLOv5 Model. *ISPRS Int. J. Geo-Inf.* **2024**, *13*, 176. https://doi.org/10.3390/ijgi13060176

Academic Editors: Wolfgang Kainz and Wei Huang

Received: 17 March 2024
Revised: 20 May 2024
Accepted: 26 May 2024
Published: 28 May 2024

Copyright: © 2024 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) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Road intersections are key hubs in urban road networks, serving as major sites for the convergence of urban traffic flows and thus prone to traffic bottlenecks [1]. Hence, road intersections have become a focal research object in the transportation field, which can provide significant decision-making support for urban management and transportation planning. In particular, generating detailed models of road intersections has played an increasingly important role in urban transportation GIS (geographic information service). However, traditional road map production methods mainly rely on professional surveying technologies, such as remote sensing and mobile mapping, which suffer from high costs, object occlusions, and long updating cycles. In the era of ubiquitous mapping, a large amount of vehicle trajectory data has been increasingly collected. These crowdsourced trajectory data have the advantages of wide coverage, rapid updating, easy collection, and low costs [2], greatly complementing the deficiency of professional surveying methods.

Currently, more and more scholars have been devoted to extracting road intersection information from trajectory data and have proposed many advanced algorithms, which can be classified into two main kinds, i.e., vector-based methods [3–13] and raster-based

methods [14–18]. Vector-based approaches attempt to explore vehicle movement characteristics, such as speed changes, heading changes, and turning time differences, to segment trajectory points or lines into road intersections or non-intersections through supervised or non-supervised methods. However, due to the equipped GNSS (Global Navigation Satellite System) devices, crowdsourced trajectory data possibly suffer from spatiotemporal heterogeneities, typically manifested as high noise, sparse sampling, and uneven density. Additionally, traditional algorithms based on movement features are limited in balancing computation efficiency and identification accuracy. Recently, Zhang et al. (2022) detected road intersection trajectories by combining several motion features, such as direction change, speed change, and turning distance ratio, and then employed a DBSCAN (Density-Based Spatial Clustering of Applications with Noise) cluster to recognize road intersection objects [3]. Zhou et al. (2023) also first detected turning point sets according to the direction change of a single vehicle trajectory and the direction diversity between different vehicle trajectories [4]. They then clustered turning point sets into different groups of road intersections to determine the position of individual road intersections. Chen et al. (2023) tried to develop HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) clustering to extract object-level road intersections from candidate trajectory points [5]. Liu et al. (2022) adopted an extreme Deep Factorization Machine (xDeepFM) model to select trajectory points at road intersections before clustering them as individual road intersections [6]. Wang et al. (2022) identified turning segments and computed their corresponding centroids to adaptively determine the clustering parameter in identifying potential intersections [7]. Meng et al. (2021) first roughly selected trajectory points at road intersections by mining road entrances and exits. They then performed the K-means algorithm, DBSCAN algorithm, and hierarchical clustering algorithm to further determine the location of road intersections [8]. Wan et al. (2019) utilized a decision tree model to detect lane-changing behaviors implied in trajectories based on multiple spatiotemporal characteristics and further recognized road intersection areas using a moving window approach [9]. Deng et al. (2018) detected candidate points of road intersections based on hotspot analysis [10]. Xie et al. (2017) identified common connecting points within intersection areas by calculating the longest common subsequence [11]. Wang et al. (2017) determined the positions and ranges of road intersections by creating a density grid of turning points and adopting mean-shift clustering on these turning points [12]. Tang et al. (2017) also identified pairs of turning points and conducted a growing clustering on these turning points based on their distance and angle metric to find road intersection regions [13]. Raster-based methods convert crowdsourced trajectories to raster images and apply sophisticated image processing methods to extract road structures. For example, Deng et al. (2023) developed a three-step method based on conversion–segmentation–optimization to extract road intersections from rasterized trajectory images [14]. Li et al. (2021) combined morphological processing, density peak clustering, and tensor voting to extract seed intersections [15]. Zhang et al. (2020) proposed partitioning raw trajectory data into a series of multi-temporal raster images, from which they extracted multi-temporal road segments using a mathematical morphology method [16]. Li et al. (2019) combined intersection-related features extracted from original trajectories and rasterized images into a fusion mechanism for detecting road intersections [17]. Hu (2019) integrated remote sensing images and rasterized trajectory images into a convolutional neural network for identifying road intersections [2]. Wang (2017) proposed a mathematical morphology method to extract road intersections based on rasterizing trajectory data [18].

In summary, vector-based methods based on motion features are limited, as they trade-off efficiency and accuracy due to data sampling rates and clustering algorithms. The idea of rasterizing trajectory data can enable the rapid identification and segmentation of intersection trajectories but is also limited by the trade-off efficiency and accuracy due to heterogeneous trajectory density and diverse image processing operators. Generally, the task of detecting road intersections from trajectory data is similar to object detection in computer vision, which focuses on identifying specific objects in images or videos and

determining their positions. This paper draws on the experience of relevant scholars and aims to seek an efficient and accurate method to detect road intersections implied in trajectory data from a computer vision perspective.

Existing deep learning algorithms for object detection can be classified into two categories, namely two-stage object detection algorithms and one-stage object detection algorithms [19]. The most representative of two-stage object detection algorithms is R-CNN (Region-based Convolutional Neural Network), which originated from CNNs (Convolutional Neural Networks) [20] and already derived from a big family, such as R-CNN [21], Fast R-CNN [22], Faster R-CNN [23] and Mask R-CNN [24]. For example, Zhou (2018) trained a Faster R-CNN model to automatically identify and locate different kinds of road intersections from high-resolution remote sensing images [25]. Yang et al. (2022) developed a deep learning framework named Mask-RCNN (Mask Region-based Convolutional Neural Network) to automatically detect the location and size information of road intersections from crowdsourced big trace data [26]. As another branch of CNN, GCN (Graph Convolution Neural Network) [27] was proposed to deal with graph-structured data, also applied to detect or classify urban interchanges in vector road networks [28,29]. Generally, two-stage object detection algorithms have a high detection accuracy but have a high time cost because of multi-run detection and classification processes.

Comparatively, one-stage algorithms treat object detection as a regression task and directly extract features from the input image, reducing redundant computations and greatly improving detection speed. The first one-stage algorithm is the well-known YOLO (You Only Look Once) algorithm, proposed by Redmon et al. [30] in 2016. The YOLO algorithm analyzes all pixels in the input image and directly predicts the bounding box information of each detected object and its corresponding class label. It has the advantages of fast detection speed, global inference capability, and good generalization. Up to now, the YOLO algorithm has undergone multiple versions of development iteration, typically such as YOLOv2 [31], YOLOv3 [32], YOLOv4 [33], YOLOv5, and so on. Compared with two-stage algorithms, one-stage algorithms need lower time costs and may show more reliable detection results. Some studies have attempted to improve the YOLO model to accomplish automated detection of road intersections [34,35]. Particularly, as the literature [36–38] stated, the YOLOv5 algorithm not only has a smaller model size and faster detection speed but also achieves a high detection accuracy. Hence, we try to improve the YOLOv5 model to detect road intersections from crowdsource trajectory data. The specific workflow of the improved YOLOv5 model is shown in Figure 1.

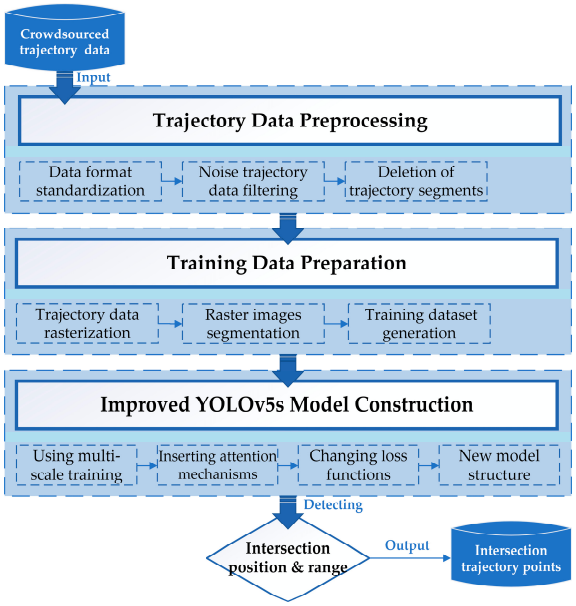


Figure 1. Flow chart of road intersection detection.

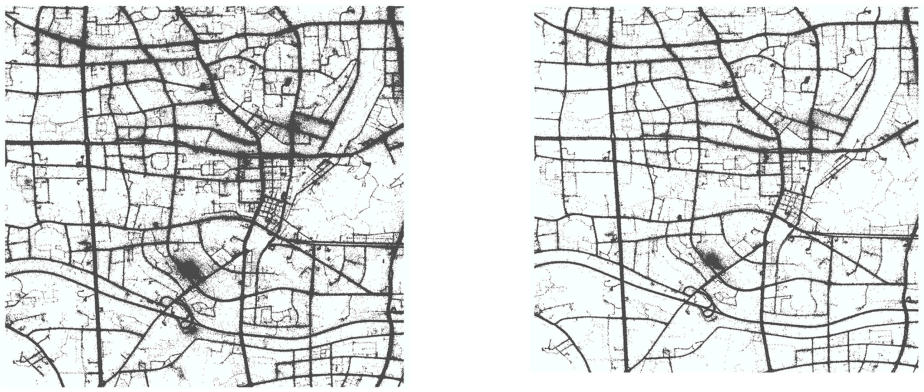
2. Training Data Preparation

2.1. Trajectory Data Preprocessing

Trajectory data record the temporal sequence of vehicle positions and motion states [14]. Due to GNSS device anomaly or signal failure, raw trajectory data may contain some noise data or redundant data, which increases computational complexity in the following processing steps. To ensure the used trajectory data portrays the actual road network, we conducted trajectory data preprocessing:

1. Data format standardization processing
   First, the ISO 8601 [39] format of stored time is converted to Unix timestamp format in seconds, and the encrypted vehicle identification (string) is converted into an integer. Secondly, the WGS84 geographical coordinate system of raw trajectory data is converted to the UTM project coordinate system. The vehicle trajectory can be obtained by connecting the chronologically ordered points collected from one identical vehicle.
2. Noise trajectory data filtering
   When the distance of two successive points is close to 0, the latter point will be deleted. When the time interval or distance between two successive points exceeds a given threshold, the original trajectory will be split into two sub-trajectories at these two successive points. Considering that the speed limit on most urban roads in China is 80 km/h, the distance threshold is set as 666 m according to an average sampling interval of about 30 s (also set as the time threshold).
3. Deletion of unrepresentative trajectory segments
   If some trajectory segments after the aforementioned preprocessing steps contain fewer than six points, those segments are unrepresentative to portray road networks and will be deleted.

Figure 2a illustrates the original trajectory data before preprocessing, and the reserved trajectory data after preprocessing is shown in Figure 2b.



(**a**) Trajectory data before preprocessing (**b**) Trajectory data after preprocessing

Figure 2. Trajectory data comparison before and after preprocessing.

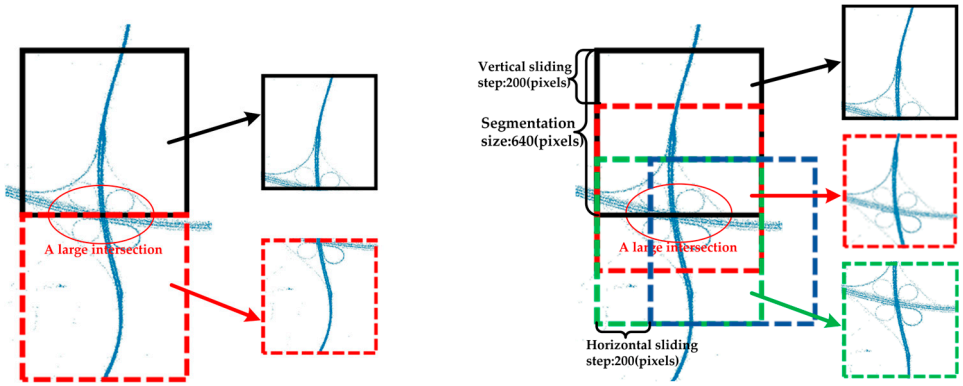
2.2. Trajectory Data Rasterization

Trajectory data rasterization aims to convert vector trajectory data into raster images. In this study, we utilize the Python package "TransBigData v0.5.3" (https://github.com/ni1o1/transbigdata/, accessed on 25 May 2024) to perform trajectory data rasterization. Through multiple experiments, we set the rasterization parameters as follows: the grid size is set as 2.5 m, the grid shape is set as "rectangle", and the DPI (dots per inch) value is set as 2560. To ensure that the rasterization image fully covers the whole study area, the parameters of upper, lower, left, and right spacing are all set as 0.

2.3. Raster Image Segmentation

To meet the input requirements of the experimental model, it is necessary to segment the original rasterization image into raster images with specified sizes. There are two kinds

of raster image segmentation, namely, translation segmentation and sliding segmentation. As shown in Figure 3a, translation segmentation may result in some road intersections being segmented as different raster images, generating insufficient training samples. As shown in Figure 3b, sliding segmentation can generate more raster images for training data preparation, and the segmented raster images can better maintain complete structures of road intersections. Therefore, the study authors chose to use sliding segmentation for raster image segmentation. According to the actual input requirements of the model, the segmentation size was set as 640 × 640 pixels, and the step sizes of vertical sliding and horizontal sliding were both set to 200 pixels. For explanation, vertical sliding moves the segmentation window from top to bottom while horizontal sliding moves the segmentation window from left to right for the purpose of more completely covering road intersections.



(**a**) Translation segmentation (**b**) Sliding segmentation

Figure 3. Translation segmentation and sliding segmentation.

2.4. Training Dataset Generation

Considering diverse shapes and different sizes in road intersections, we selected five trajectory datasets collected in Wuhan and Changsha, China, for the fusion experiments. As shown in Table 1, the trajectory data of experimental areas 1 and 2 in Changsha were collected between 1 October and 30 October 2018, totaling 25,497,345 trajectory points, with an average sampling interval of 27.24 s. The trajectory data of experimental areas 3, 4, and 5 in Wuhan were collected between 1 May to 6 May 2017, totaling 83,009,353 trajectory points, with an average sampling interval of 6.01 s.

Table 1. Statistical description of experimental datasets.

Experimental Area	Longitude Range (E)	Latitude Range (N)	Coverage (km^2)	Number of Trajectory Points	Average Time Interval (s)
Experimental area 1 in Changsha	112.9690°~113.0320°	28.1370°~28.1910°	6.18 × 5.98	14,839,457	27.24
Experimental area 2 in Changsha	112.9690°~113.0320°	28.0770°~28.1370°	6.18 × 6.65	10,657,888	27.24
Experimental area 3 in Wuhan	114.1423°~114.4435°	30.5342°~30.7041°	28.5 × 18.8	57,746,881	6.01
Experimental area 4 in Wuhan	114.3723°~114.4847°	30.4352°~30.5272°	10.8 × 10.2	6,364,574	6.16
Experimental area 5 in Wuhan	114.1981°~114.2949°	30.4802°~30.5342°	9.30 × 6.00	15,134,054	6.71

Despite sliding segmentation, there were some segmented raster images that did not qualify as training or validation samples. Hence, the manual selection of the segmented raster images was necessary to generate a training dataset that met the input requirement of the YOLO model. Table 2 lists the number of raster images before and after manual selection. Specifically, experimental area 2 in Changsha was used for results validation, and thus, it was processed by a different sliding segmentation rule from other experimental areas, indicating significantly fewer segmented raster images. The sliding segmentation rule is presented in detail in Section 4.2. After the manual selection process, a total of 5244 training samples were obtained. We then utilized the "Make Sense" annotation tool

(https://www.makesense.ai/, accessed on 25 May 2024) and referred to the annotation method described in [40] to manually create the class label for each raster image in the training sample data.

Table 2. Statistical description of the segmented raster images before and after manual selection.

Experimental Area	Data Usage Purpose	Original Rasterization Image Size	Raster Images before Selection	Raster Images after Selection
Experimental area 1 in Changsha	Training sample	14,332 × 12,288 pixels	3944	1415
Experimental area 2 in Changsha	Results validation	12,903 × 12,288 pixels	36	36
Experimental area 3 in Wuhan	Training sample	23,627 × 12,288 pixels	6666	1443
Experimental area 4 in Wuhan	Training sample	20,981 × 10,219 pixels	4837	1319
Experimental area 5 in Wuhan	Validation sample	11,413 × 12,288 pixels	3074	1067

3. Improved YOLOv5s Model Construction

3.1. Overview of YOLOv5 Model

In recent years, there have been several breakthroughs in deep learning algorithms for object detection. As an end-to-end (one-stage) object detection algorithm, the YOLO algorithm is characterized by its small model size, fast processing efficiency, low false detection rate, and strong generalization ability. Up to now, the YOLO algorithm has undergone multiple versions of development iteration. Among the YOLO family, the YOLOv5 model was first released by the Ultralytics company in June 2020. So far, the development team has continuously offered some minor-update versions of YOLOv5, and we employed version v6.1, released in February 2022, in this paper. Based on a similar network structure, YOLOv5 also contains five practical models according to different weights, widths, and depths. These five models are YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, respectively, of which we used YOLOv5s model for transfer learning to detect road intersections. In general, the trained YOLOv5 model outputs the bounding boxes of each detected object from the input image as the form like [class, x, y, w, h, confidence], where "class" represents the detected object's category, "x, y" represents the centroid coordinate of the bounding box, "w, h" represents the corresponding box's width and height, and "confidence" represents the confidence score labeled as the corresponding category. The original YOLOv5 model mainly consists of four components:

(1) Input. It comprises three main parts, i.e., mosaic data augmentation, adaptive computation of anchor boxes, and adaptive image scaling. This component enhances the model's ability to recognize and locate multi-scale objects;
(2) Backbone. It contains three main modules, i.e., Focus module, C3 (the improved Bottleneck CSP) module, and SPP (Spatial Pyramid Pooling) module. This component is to achieve good performance in detecting various objects with different scales and layouts;
(3) Neck. There are two main modules, i.e., the FPN (Feature Pyramid Network) and PAN (Path Aggregation Network). This component is to increase the capability of multi-scale feature fusion in handling multi-scale objects;
(4) Head. As the output layer, it introduces the CIoU (Complete Intersection over Union) loss function to detect the object box and further uses the NMS (Non-Maximum Suppression) algorithm to find multiple objects or duplicate detection of a single object.

3.2. Using Multi-Scale Training Strategy

Although the YOLOv5 model has a certain ability to detect multi-scale objects, it may be more or less impacted by the input training dataset. Particularly, due to the complex structures of road intersections, a fixed segmentation size given in Section 2.3 may hardly ensure the training samples can cover different sizes and scales of road intersections. Hence, we adopted a multi-scale training strategy to enhance the adaptability of YOLOv5 to input

the training dataset. The multi-scale training strategy is to resample the original raster images segmented by Section 2.3 into different sizes and iteratively input them into our improved YOLOv5 model presented in Section 3.5. The rule of multi-scale training, including enlarging and shrinking operations, is described by Formulas (1) and (2), respectively.

$$imgsize_1 = imgsize * 1.5 + rand \quad (1)$$

$$imgsize_2 = imgsize * 0.5 \quad (2)$$

where *imgsize* is the image size originally segmented in Section 2.3. $imgsize_1$ and $imgsize_2$ are the resample image sizes after enlarging and shrinking operations, respectively. *rand* is a random integer that does not exceed the original image size.

To verify that multi-scale training can improve the performance of the YOLOv5s model, we conducted a comparative experiment. As listed in Table 3, the precision of the YOLOv5s model increased by 0.7%, and mAP_0.5 increased by 0.4% when multi-scale training was employed, indicating the necessity of using multi-scale training. Hence, we will default to using multi-scale training in subsequent experiments.

Table 3. Comparative experiment of YOLOv5s model using/not using multi-scale training.

No.	Group	Precision (%)	Recall (%)	mAP_0.5 (%)
1	YOLOv5s	82.2	79.7	83.1
2	YOLOv5s + multi-scale training	82.9	79.2	83.5 (+0.4)

3.3. Inserting Attention Mechanisms

In the YOLOv5 model, each convolutional kernel corresponds to a local receptive field, which only contains limited information about the local context. Some studies found that inserting attention mechanism modules can effectively solve the above-mentioned problems of the local receptive field [41]. To verify that inserting an attention mechanism can improve the performance of the YOLOv5s model, we selected several mainstream modules of a convolutional attention mechanism, including SE (squeeze-and-excitation attention module) [42], NAM (Normalization-based Attention Module) [43], GAM (Global Attention Module) [44], ShuffleAM (Shuffle Attention Module) [45], and SimAM (Similarity-based Attention Module) [46], and we conducted a comparative experiment. The comparative results are shown in Table 4.

Table 4. Comparative experiments of inserting different attention mechanism modules.

No.	Group	Precision (%)	Recall (%)	mAP_0.5 (%)
1	YOLOv5s	82.9	79.2	83.5
2	YOLOv5s + GAM	83.2	80.8	86.9 (+3.4)
3	YOLOv5s + NAM	80.8	82.3	86.3 (+2.8)
4	YOLOv5s + SE	85.1	79.6	85.8 (+2.3)
5	YOLOv5s + ShuffleAM	84.7	82.7	85.9 (+2.4)
6	YOLOv5s + SimAM	84.9	79.2	86.3 (+2.8)

As listed in Table 4, the GAM attention mechanism module has the highest value of mAP_0.5 than the other attention mechanisms (increased by 3.4%), followed by NAM and SimAM. Therefore, we inserted the GAM attention mechanism module into the original YOLOv5s network structure to enhance the model's attention to the targets' features and improve the model's detection performance and generalization ability. GAM combines channel attention and spatial attention to amplify global cross-dimensional interaction. GAM incorporates the global features of the input data to understand the overall structure and extract global features. By performing attention operations on the global features of

the input data, the model can better capture global relationships and thereby improve detection performance. A schematic diagram of the GAM attention module is shown in Figure 4 below.

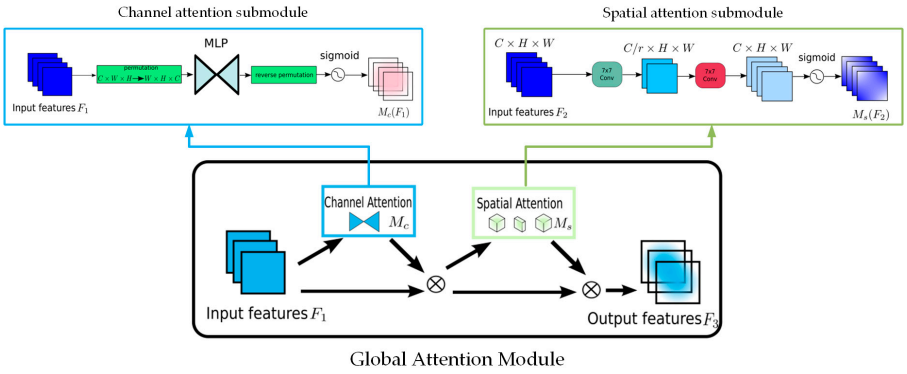


Figure 4. Working principle of GAM attention module [44].

3.4. Changing Loss Functions

The loss function is used to measure the difference between the model's predicted values and the actual values. The loss function has a significant impact on the model's performance. The loss function of the YOLOv5 model consists of three main parts: classification loss, confidence loss, and localization loss, as shown in Formula (3):

$$Loss = L_{Class} + L_{Conf} + L_{Loc} \tag{3}$$

In the original YOLOv5 model, both classification loss and confidence loss are calculated based on Binary Cross-Entropy Loss (BCE Loss), while localization loss is computed based on CIoU Loss (Complete Intersection over Union Loss) [47]. In this paper, we focused on detecting road intersections in urban areas where dense targets may exist, and the background trajectory points may have a significant impact on extraction accuracy. To reduce the model's false detection rate, it was necessary to change the original confidence loss function to another function that performs better detection accuracy in dense scenarios. Additionally, to ensure that the predicted bounding boxes are accurately centered at the actual centroids of road intersections and that the box size corresponds to the intersection size. Thus, it was also necessary to change the original localization loss function with a function that provides higher localization accuracy.

- Changing the localization loss function

To verify that changing the localization loss function can improve the performance of the YOLOv5s model, we retained the GAM attention mechanism module and selected several mainstream localization loss functions, such as GIoU Loss (Generalized Intersection over Union Loss) [48], DIoU Loss (Distance Intersection over Union Loss) [49], EIoU Loss (Efficient Intersection over Union Loss) [50] and SIoU Loss (Softmax Intersection over Union Loss) [51], to substitute the original localization loss function (CIoU Loss) for a comparative experiment. The comparative experimental results are shown in Table 5 below.

Table 5. Comparative experiment when using different localization loss functions.

No.	Group	Precision (%)	Recall (%)	mAP_0.5 (%)
1	YOLOv5s + GAM + CIoU	83.2	80.8	86.9
2	YOLOv5s + GAM + GIoU	82.5	82.0	86.3 (−0.6)
3	YOLOv5s + GAM + DIoU	79.1	75.4	80.0 (−6.9)
4	YOLOv5s + GAM + EIoU	84.9	81.2	88.0 (+1.1)
5	YOLOv5s + GAM + SIoU	82.7	78.7	83.7 (−3.2)

It can be seen from Table 5 that only the EIoU Loss function shows a certain improvement in detection performance (increased by 1.1%), and other compared loss functions actually show a lower model performance. Therefore, we chose to change the original localization loss function to EIoU Loss. EIoU Loss considers the overlapping area, the centroid distance, the width difference, and the height difference between the predicted box and the ground truth box. In CIoU Loss, the penalty term uses the relative proportions of width and height rather than their absolute values. In CIoU Loss, when the width and height of the predicted box satisfy $\left\{\left(w = k\hat{w}, h = k\hat{h}\right) \middle| k \in R^{+}\right\}$, the penalty term based on relative proportions will cease to take effect, thus affecting localization accuracy. Therefore, in EIoU Loss, the width and height values of the predicted box and real target are both considered to ensure the prediction accuracy and enhance the convergence speed and regression accuracy. The formula for calculating EIoU Loss is shown as Formulas (4) and (5):

$$L_{EIoU} = 1 - IoU + \frac{d^2_{center}}{d^2_{diag}} + \frac{d^2_{w,\hat{w}}}{W^2} + \frac{d^2_{h,\hat{h}}}{H^2} \tag{4}$$

$$d^2_{w,\hat{w}} = \left(\frac{w}{\hat{w}} - \frac{\hat{w}}{w}\right)^2; d^2_{h,\hat{h}} = \left(\frac{h}{\hat{h}} - \frac{\hat{h}}{h}\right)^2 \tag{5}$$

where d_{center} is the distance between the center points of the predicted bounding box and the ground truth bounding box; d_{diag}, W, and H are the diagonal length, the width and height values of the minimum enclosing rectangle of merging the predicted bounding box, and the ground truth bounding box; w and h are the width and height values of the ground truth bounding box; $\hat{w}$ and $\hat{h}$ are the width and height values of the predicted bounding box. These variables for calculating EIoU Loss are shown in Figure 5.

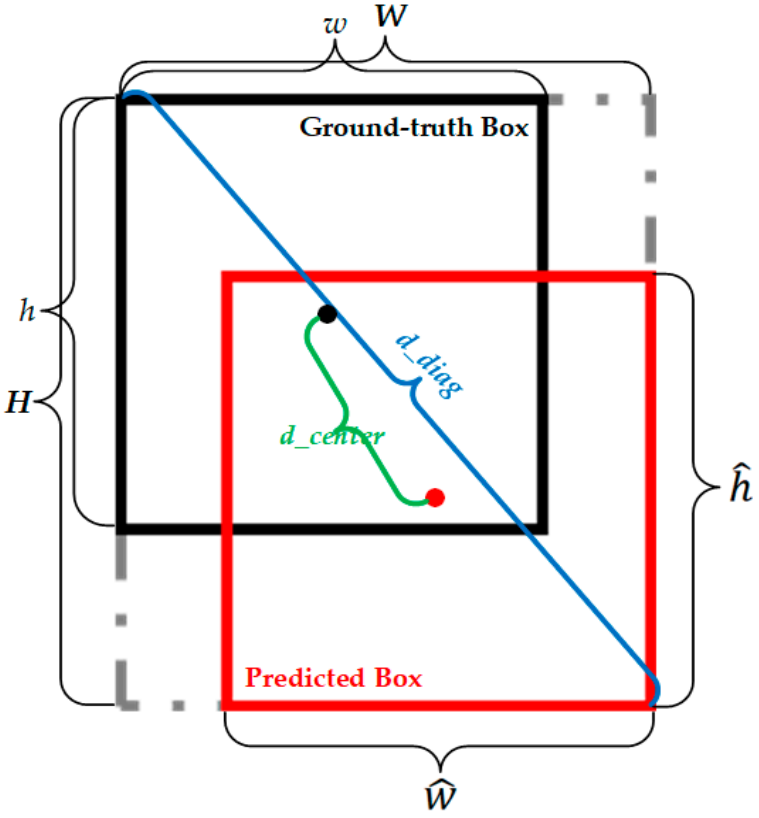


Figure 5. The schematic diagram of EIoU Loss calculation.

Table 5 illustrates that EIoU Loss is more robust than the original CIoU Loss, especially in handling small and overlapping targets. EIoU Loss can efficiently ensure that the predicted boxes will not deviate too far from the ground truth box during the model training process, significantly improving the convergence speed of the basic YOLOv5 model.

- Changing confidence loss function

To verify that changing the confidence loss function can improve the performance of the YOLOv5s model, we retained the GAM attention mechanism module and the EIoU Loss function and selected several mainstream confidence loss functions, such as Focal Loss [52], VariFocal Loss [53] and Poly Loss [54], to substitute the original confidence loss function (BCE Loss) for a comparative experiment. The comparative experimental results are shown in Table 6 below.

Table 6. Comparative experiment when using different confidence loss functions.

No.	Group	Precision (%)	Recall (%)	mAP_0.5 (%)
1	YOLOv5s + GAM + EIoU + BCE Loss	84.9	81.2	88.0
2	YOLOv5s + GAM + EIoU + Focal Loss	85.1	81.5	88.7 (+0.7)
3	YOLOv5s + GAM + EIoU + VariFocal Loss	81.9	80.4	84.8 (−3.2)
4	YOLOv5s + GAM + EIoU + Poly Loss	81.4	74.4	81.6 (−6.4)

It can be seen from Table 6 that Focal Loss shows a certain improvement in detection performance (increased by 0.7%), and other compared loss functions actually show a lower model performance. Therefore, we chose to change the original confidence loss function to Focal Loss. Focal Loss is a loss function used in object detection tasks to deal with the problem of class imbalance and improve the performance of object detection in the presence of background classes. In practical scenarios of object detection, there are typically a large number of background objects but only a few target samples. Traditional Binary Cross-Entropy Loss may focus on the majority classes and ignore the minority category due to class imbalance. Focal Loss introduces two parameters, i.e., α (alpha) and γ (gamma), to adjust the loss function. When $\alpha = 1$ and $\gamma = 0$, Focal Loss reverts to the original BCE Loss. The calculation formula for Focal Loss is shown as follows:

$$L_{Focal} = -\alpha(1-p)^{\gamma} * \log(p) \quad (6)$$

where α is the balanced factor, p is the predicted probability of the correct class, and γ is the focusing parameter.

The core idea of Focal Loss is to reduce the weights of easily classified samples and increase the weights of hard-classified samples during training to make the model focus on these samples hard to classify.

3.5. New Model Structure

The modified network structure of YOLOv5s is shown in Figure 6 below. The main improvements to this model concentrate on three aspects: inserting attention mechanisms, changing confidence loss functions, and localization loss functions. Inserting attention mechanism modules will alter the original network structure. In this paper, a total of four attention mechanism modules were inserted into the improved YOLOv5s model, of which one attention mechanism is inserted into the backbone network, positioned at the 9th layer, and the other three attention mechanisms were inserted into the neck network, located at the 19th, 23rd, and 27th layers, respectively [55].

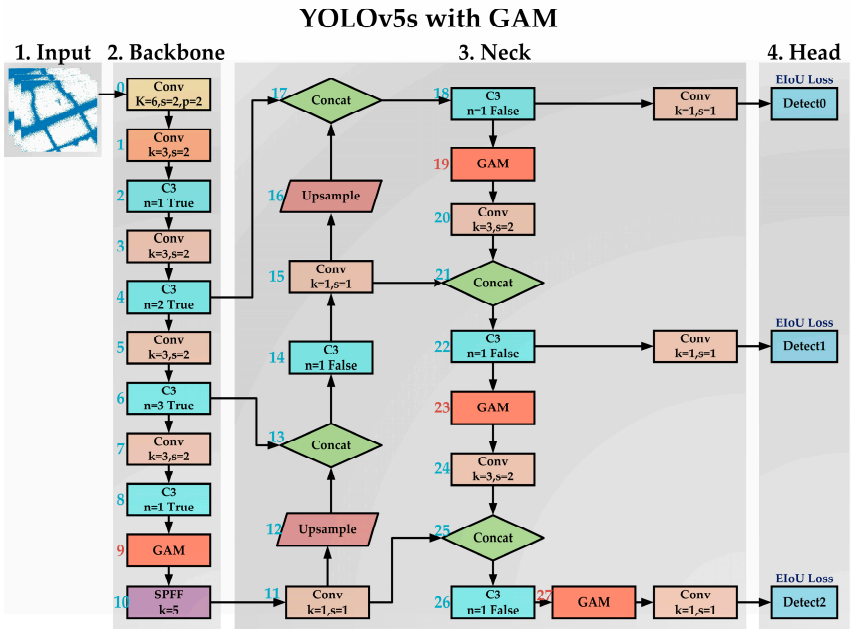


Figure 6. Network structure of improved YOLOv5s.

4. Experiment and Result Analysis

4.1. Experimental Setup and Model Training Parameters

All experiments in this study were conducted and run on a unified server platform. The detailed server configuration is as follows: the operating system version was Windows server* 2019, with a 64-bit operating platform; the central processor unit (CPU) consisted of 2 Intel(R) Xeon(R) Silver 4210 CPUs @2.20 GHz, with 20 cores and 40 threads; the graphics processing unit (GPU) was an NVIDIA GeForce RTX 2080Ti with 11 GB GDDR6 memory; the random access memory (RAM) comprised 4 × 32 GB DDR4 2400 MHz memory sticks, totaling 128 GB memory; the PyCharm integrated development environment and Python programming language were used for experiment test, with Python interpreter version 3.7.10; and the model was implemented under the open-source deep learning framework PyTorch and CUDA general-purpose parallel computing architecture, with PyTorch version 1.13.1 and CUDA version 11.7. The main training parameters are shown in Table 7 below.

Table 7. Key parameters for model training.

Parameter Name	Variable Name	Value/Type
Pretrained weight file	weights	YOLOv5s/YOLOv5m/YOLOv5l/YOLOv5x
Input image size	image-size	640 × 640
Training batch size	batch-size	16
Maximum training epochs	epochs	500
Optimizer	optimizer	Adam
Maximum learning rate	lrf	1×10^{-5}
Learning rate adjustment method	cos-lr	cosine annealing
Warm-up epochs	warmup_epochs	100

Approximately 80% of the samples were allocated for training the classification model, with the remaining samples reserved for classification testing. In detail, the training set consisted of 4177 positive images and 90 negative images (accounting for 2% of the total samples), which contained a total of 9933 instances. The validation set consisted of 1067 positive images and 0 negative images, totaling 2365 instances.

4.2. Evaluation Metrics for Models

In this study, precision (P), recall (R), and mean average precision (mAP) were used as the model evaluation metrics. Precision refers to the proportion of true positive samples among the total predicted samples by our model. Recall refers to the proportion of true positive predicted samples among the total number of real target samples. These are calculated according to Formulas (7) and (8), respectively. mAP_0.5 is defined as the average area enclosed by the P–R curve and two axes when the IoU (Intersection over Union) value is 0.5. During each epoch of model training, the aforementioned evaluation metrics are automatically calculated to assess the performance changes and convergence situation. If the mAP_0.5 value does not increase in the subsequent 150 epochs within the specified maximum training epochs, the model is regarded as convergence, and thus, model training is stopped.

$$P = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{7}$$

$$R = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{8}$$

where *True Positive* is the number of road intersections correctly identified by the model. *False Positive* is the number of road intersections wrongly identified by the model. *False Negative* is the number of road intersections missed by the model.

4.3. Ablation Experiment and Result Analysis

- Ablation experiment

Table 8 presents a comprehensive statistical summary of all model improvement ablation experiments, where Group 1 serves as the control group for the original model, and Groups 2–5 represent experiments using the multi-scale training strategy, inserting the attention mechanism, changing the localization loss function, and changing the confidence loss function, respectively. All groups utilized the YOLOv5s model for transfer learning. In Table 8, "✓" indicates improvement in this aspect, while "✗" indicates no improvement in this aspect.

Table 8. Comprehensive statistics of ablation experiments.

Group	Multi-Scale Training	GAM Model	EIoU Loss	Focal Loss	Precision (%)	Recall (%)	mAP_0.5 (%)
1	✗	✗	✗	✗	82.2	79.7	83.1
2	✓	✗	✗	✗	82.9	79.2	83.5 (+0.4)
3	✓	✓	✗	✗	83.2	80.8	86.9 (+3.8)
4	✓	✓	✓	✗	84.9	81.2	88.0 (+4.9)
5	✓	✓	✓	✓	85.1	81.5	88.7 (+5.6)

According to Table 8, the accuracy metrics show a growing trend during the improvement process (from Group 1 to Group 5) of multi-scale training, inserting the GAM attention mechanism and modifying the loss function, indicating the advantages of the improved YOLOv5 model. It can be concluded that utilizing the YOLOv5s model for transfer learning, along with enabling a multi-scale training strategy, inserting GAM attention mechanism modules into the original network structure, and changing the original confidence loss function BCE Loss to Focal Loss, as well as changing the original localization loss function CIoU Loss to EIoU Loss, represents the best approach for enhancing model performance (Group 5). Compared with the original model (Group 1), the most improved model exhibits the following trends in various evaluation metrics when using the validation set: precision (P) increased by 2.9%, recall (R) increased by 1.8%, and mean average precision (mAP_0.5) increased by 5.6%.

- Result analysis

A visual result comparison of road intersection detection using the original YOLOv5 model and the improved YOLOv5 model is shown in Figure 7. Figure 7a displays the detected road intersections using the original YOLOv5 model, and Figure 7b reveals the detected road intersections using the best-improved YOLOv5 model. Red boxes represent the road intersections both predicted by the original YOLOv5 model and the improved YOLOv5 model, and green boxes represent the road intersections newly detected by the YOLOv5 improved model.

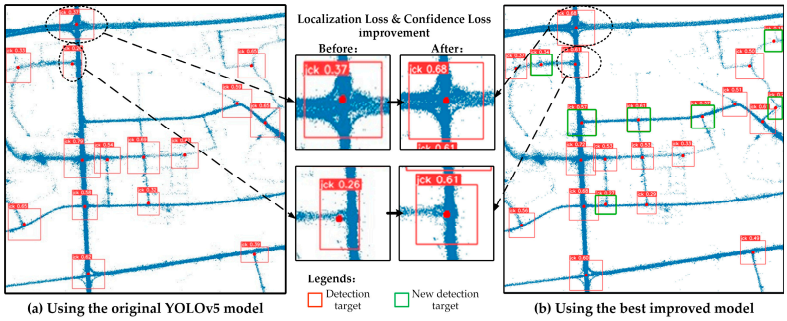


Figure 7. Comparison of road intersection detection using original and improved YOLOv5 models.

As shown in Figure 7a,b, after inserting the attention mechanism into the original YOLOv5 model, the improved model can detect more road intersections, especially the small road intersections at the edge regions of images. As shown in the middle enlarged view of Figure 7, after changing the localization loss function in the original YOLOv5 model, the positions and boundaries of road intersections detected by the improved model are closer and more similar to the centroids and spatial coverages of actual road intersections. Additionally, the scores of road intersections calculated by our improved model are significantly larger than that of the original YOLOv5 model, increasing the robustness of detected road intersections.

Figure 8 shows the changing curves of loss and accuracy values during the training process. The left plot of Figure 8 represents the changing curves of loss values for training and validation sets, while the right plot of Figure 8 represents the changing curve of mean average precision for the validation set.

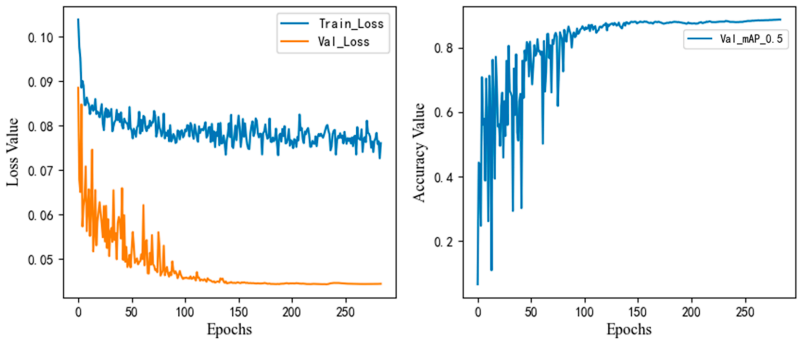


Figure 8. Curve of loss and accuracy values' variation during model training process.

4.4. Intersection Recognition and Extraction

- Intersection position detection experiment

Experimental area 2 in Changsha was used for the model validation experiments, which was not involved in the model training process. Before conducting intersection detection in this experimental area, the following preparatory work should be carried out:

1. Original raster image segmentation

The original raster image of experimental area 2 needed to be segmented because it was too large to be directly inputted into the trained model. The segmentation method was sliding segmentation with a size of 2560 × 2560 (pixels) and a sliding step of 500 (pixels), resulting in a total of 36 segmented sample images for results validation.

2. Actual georeferencing calculation

Based on the spatial coverage of the experimental area and the segmented image pixel distribution, we calculated the latitude and longitude range for each segmented raster sample image, further facilitating the determination of the actual position and range of each detected road intersection.

3. Deduplication of detected road intersections

Due to sliding segmentation, it is inevitable that the same intersection may be included in multiple sample images, leading to redundantly detected objects. If the distance between the central points of two detected intersections was less than 20 m, one of them was deleted to filter out the redundantly detected road intersections.

Figure 9 shows the detection results of our improved model in experimental area 2 in Changsha. The red circles represent the centroids of road intersections correctly identified by the improved model, the orange squares represent the centroids of road intersections incorrectly identified intersection centroids, and the green triangles represent the road intersections missed by the improved model.



Figure 9. Results of road intersection detection for experimental area 2 in Changsha.

- Comparative experiment with other models

We conducted a comparative experiment to analyze the improved model and other deep learning models for object detection. Four widely used models for object detection were selected for the experiment: YOLOv3, YOLOv5, Fast R-CNN, and Faster R-CNN. All parameters used in the compared models were set as the same values described in

Section 4.1. There were really a total of 235 road intersections in experimental area 2 in Changsha. The comparison results of road intersections detected by different models can be summarized as follows.

1. The improved YOLOv5 model proposed in this paper identified a total of 231 road intersections, of which 230 were correctly identified, 1 was misidentified, and 5 were missed;
2. The YOLOv3 model identified a total of 204 road intersections, of which 192 were correctly identified, 12 were misidentified, and 43 were missed;
3. The YOLOv5 model identified a total of 235 intersections, of which 215 were correctly identified, 20 were misidentified, and 20 were missed;
4. The Fast R-CNN model identified a total of 287 road intersections, of which 210 were correctly identified, 77 were misidentified, and 25 were missed;
5. The Faster R-CNN model identified a total of 291 road intersections, of which 214 were correctly identified, 77 were misidentified, and 21 were missed.

The accuracy, precision, recall, and other characteristics of compared models are shown in Table 9 below.

Table 9. Results of comparative experimental with other models.

Model in Use	Accuracy	Precision	Recall	Others
Ours (improved YOLOv5)	97.46%	99.57%	97.87%	GPU memory usage 9.5 GB
YOLOv3	77.73%	94.12%	81.70%	GPU memory usage 4.5 GB
YOLOv5	84.31%	91.49%	91.49%	GPU memory usage 4.5 GB
Fast-RCNN	67.31%	73.17%	89.36%	GPU memory usage 20 GB
Faster-RCNN	68.59%	73.54%	91.06%	GPU memory usage 20 GB

- Intersection range extraction experiment

In the intersection range extraction experiment, the predicted box boundaries were directly used as the boundaries of detected intersections. Based on the pixel range occupied by the predicted boxes, the actual rectangular range of each road intersection was calculated from the georeferencing raster images. When there was a partial overlapping area between adjacent intersections, we evenly divided the overlapping area and redefined the spatial range of the two intersections. The trajectory segments in the spatial range of each road intersection detected in experimental area 2 in Changsha are shown in Figure 10 below.

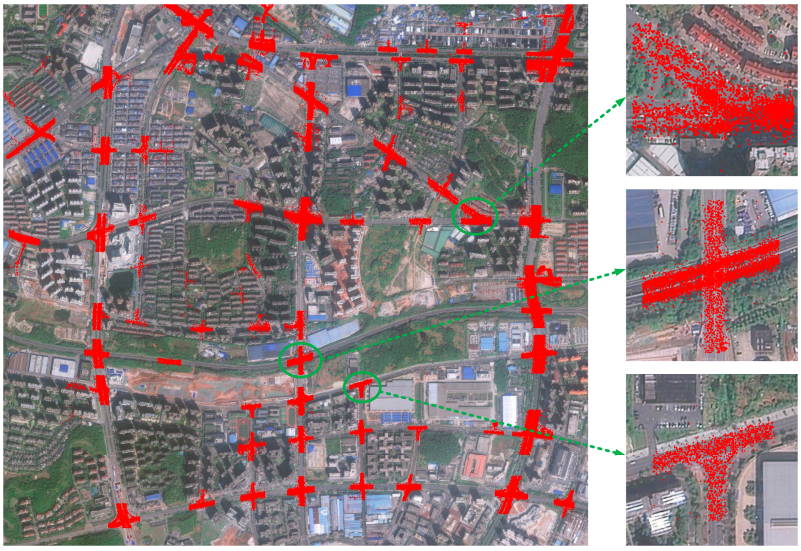


Figure 10. Trajectory segments extracted from the detected intersections in experimental area 2 in Changsha.

5. Discussion

The improved YOLOv5 model proposed in this paper can effectively detect and recognize road intersections from rasterization-based trajectory images. The detected road intersections are overlapped with satellite images for ground verification, and accuracy, precision, and recall were calculated for accuracy evaluation. It was found that the improved model used in the validation experiment misidentified one road intersection, and only five road intersections were not identified by the improved model, with all accuracy metrics above 97%. Particularly, the precision rate reached 99.57%, indicating excellent model performance. A comparison with other mainstream object detection models revealed that the improved model significantly outperformed the other models in all accuracy metrics, demonstrating its superior performance. Moreover, the computational resource requirement for the model proposed in this paper is much lower than those for the RCNN series models, indicating its low cost and high efficiency in practical applications. The results shown in Figure 10 indicate the expected results of trajectory segmentation based on the predicted bounding boxes detected by the improved model. As shown in the right enlarged view in Figure 10, the segmentation results of "+"-shaped intersection, "T"-shaped intersection, and "Y"-shaped intersection are very consistent with the actual shapes of such road intersections.

6. Conclusions

In this study, we improved upon the original YOLOv5 model by inserting attention mechanism modules, changing the original loss function, and adopting a multi-scale training strategy. In actual intersection detection tasks, compared with other deep-learning-based object detection models, the improved model achieved higher recognition accuracy, a lower misidentification rate, and stronger generalization ability. Using the model proposed in this paper, the position and range of road intersections can be quickly and accurately detected, and the intersection trajectory points can be segmented from original trajectory data based on these detected objects, greatly enriching the means of road intersection extraction and improving detection efficiency. Areas that require improvement in future research include, firstly, studying the classification of road intersections of different shapes and carefully subdividing road intersection categories to meet other applications' needs. Secondly, we must focus on segmenting the traffic patterns within road intersections to establish accurate and complete road intersection maps.

Author Contributions: Conceptualization, Yunfei Zhang; extracting method, Gengbiao Tang; software, Gengbiao Tang; validation, Naisi Sun; formal analysis, Gengbiao Tang; investigation, Gengbiao Tang; writing—original draft preparation, Gengbiao Tang; writing—review and editing, Yunfei Zhang; visualization, Gengbiao Tang and Naisi Sun; funding acquisition, Yunfei Zhang and Gengbiao Tang. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Nature Science Foundation of China (grant numbers 42371474 and 41971421), the Science and Technology Innovation Program of Hunan (grant numbers 2021RC3099 and 2022JJ30590), and Changsha University of Science and Technology practical innovation project (grant number CLSJCX22002).

Data Availability Statement: Data are contained within the article.

Acknowledgments: We thank the anonymous reviewers for their constructive comments.

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

References

1. Wang, C. *Analysis and Prediction of Intersection Traffic Conflicts Based on Trajectory Data Extraction*; Beijing Jiaotong University: Beijing, China, 2022. [CrossRef]
2. Hu, H. *Research on Urban Road Deep Learning Identification Method Based on Fusion of Multi-Source Data*; Wuhan University: Wuhan, China, 2019.
3. Zhang, Y.; Tang, G.; Fang, X.; Chen, T.; Zhou, F.; Luo, Y. Hierarchical Segmentation Method for Generating Road Intersections from Crowdsourced Trajectory Data. *Appl. Sci.* **2022**, *12*, 10383. [CrossRef]

4. Zhou, Q.; Yang, C.; Li, J.; Yan, K. Road Intersection Extraction Algorithm Based on Trajectory Directional Features. *Geogr. Inf.* **2023**, *21*, 7–10.
5. Chen, W.; Du, J. A Method for Extracting Road Intersections Using Low-frequency Trajectory Data. *Surv. Mapp. Bull.* **2023**, *1*, 127–133. [CrossRef]
6. Liu, Y.; Qing, R.; Zhao, Y.; Liao, Z. Road Intersection Recognition via Combining Classification Model and Clustering Algorithm Based on GPS Data. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 487. [CrossRef]
7. Wang, F.; Guo, Q.; Xu, X. Method for Extracting Road Intersection and Its Structure Based on Trajectory Data. *Sci. Geogr. Sin.* **2022**, *47*, 212–218+244. [CrossRef]
8. Meng, Q.; Song, Z.; Wang, J.; Ge, Z. Identification and Extraction of Urban Road Intersections Using Floating Car GPS Trajectory Data. *Surv. Mapp. Bull.* **2021**, *9*, 59–63. [CrossRef]
9. Wan, Z.; Li, L.; Yang, M.; Zhou, X. Decision Tree Model for Extracting Road Intersection Features from Vehicle Trajectory Data. *Acta Geod. Et Cartogr. Sin.* **2019**, *48*, 1391–1403.
10. Deng, M.; Huang, J.; Zhang, Y.; Liu, H.; Tang, L.; Tang, J.; Yang, X. Generating Urban Road Intersection Models from Low-frequency GPS Trajectory Data. *Int. J. Geogr. Inf. Sci.* **2018**, *32*, 2337–2361. [CrossRef]
11. Xie, X.; Liao, W.; Aghajan, H.; Veelaert, P.; Philips, W. Detecting Road Intersections from GPS Traces Using Longest Common Subsequence Algorithm. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 1. [CrossRef]
12. Wang, J.; Wang, C.; Song, X.; Raghavan, V. Automatic Intersection and Traffic Rule Detection by Mining Motor-vehicle GPS Trajectories. *Comput. Environ. Urban Syst.* **2017**, *64*, 19–29. [CrossRef]
13. Tang, L.; Niu, L.; Yang, X.; Zhang, X.; Li, Q.; Xiao, S. City Road Intersection Identification and Structure Extraction Using Trajectory Big Data. *Acta Geod. Et Cartogr. Sin.* **2017**, *46*, 770–779.
14. Deng, M.; Luo, B.; Tang, J.; Yao, Z.; Liu, G.; Weng, X.; Hu, R.; Chai, H.; Hu, W. Road Intersection Extraction Method Considering Heterogeneity of Trajectory Density Distribution. *Acta Geod. Et Cartogr. Sin.* **2023**, *52*, 1000–1009.
15. Li, Y.; Xiang, L.; Zhang, C.; Wu, H.; Gong, J. Road Intersection Recognition Based on Multilevel Fusion of Vehicle Trajectory and Remote Sensing Images. *Acta Geod. Et Cartogr. Sin.* **2021**, *50*, 1546–1557.
16. Zhang, Y.; Zhang, Z.; Huang, J.; She, T.; Deng, M.; Fan, H.; Xu, P.; Deng, X. A Hybrid Method to Incrementally Extract Road Networks Using Spatio-Temporal Trajectory Data. *Int. J. Geo-Inf.* **2020**, *9*, 186. [CrossRef]
17. Li, S.; Xiang, L.; Zhang, C.; Gong, J. Research on the Extraction of Urban Road Network Intersections Based on Low-frequency Taxi Trajectory. *J. Geo-Inf. Sci.* **2019**, *21*, 1845–1854.
18. Wang, D. *Extraction of Road Network Information Based on Low-Frequency Taxi GPS Trajectory Data*; Wuhan University: Wuhan, China, 2017.
19. Leng, M. *Research and Application of Improved Algorithm for Small Object Detection Based on YOLO*; Chongqing Technology and Business University: Chongqing, China, 2023.
20. LeCun, Y.; Boser, B.; Denker, J.; Henderson, D.; Howard, R.; Hubbard, W.; Jackel, L. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
21. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014. [CrossRef]
22. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448. [CrossRef]
23. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
24. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, Venice, Italy, 22–29 October 2017. [CrossRef]
25. Zhou, W. *Road Extraction from High-Resolution Remote Sensing Images Based on Road Intersections*; Wuhan University: Wuhan, China, 2018.
26. Yang, X.; Hou, L.; Guo, M.; Cao, Y.; Yang, M.; Tang, L. Road Intersection Identification from Crowdsourced Big Trace Data Using Mask-RCNN. *Trans. GIS TG* **2022**, *26*, 278–296. [CrossRef]
27. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907. [CrossRef]
28. Yang, M.; Jiang, C.; Yan, X.; Ai, T.; Cao, M.; Chen, W. Detecting Interchanges in Road Networks Using a Graph Convolutional Network Approach. *Int. J. Geogr. Inf. Sci.* **2021**, *36*, 1119–1139. [CrossRef]
29. Fang, Z.; Zhong, H.; Zou, X. Urban Road Extraction Based on Combination of Trajectory Continuity and Image Feature Similarity. *Acta Geod. Et Cartogr. Sin.* **2020**, *49*, 1554–1563.
30. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; IEEE: Toulouse, France, 2016. [CrossRef]
31. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; IEEE: Toulouse, France, 2017; pp. 6517–6525. [CrossRef]
32. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767. [CrossRef]

33. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934. [CrossRef]
34. Shao, X.; Zhang, C.; Wei, Y.; Zhang, X.; Zhou, Y.; Zhang, Z. Automatic Recognition of Road Intersections in Remote Sensing Images Based on Improved YOLOv3 Algorithm. *Spacecr. Recovery Remote Sens.* **2022**, *43*, 123–132.
35. Wang, L.; Liu, Z.; Jin, F.; Wang, F. Research on Automatic Detection Algorithm of Road Intersections. *Surv. Mapp. Sci.* **2020**, *45*, 126–131+146. [CrossRef]
36. Hu, H.; Li, Z.; He, Z.; Wang, L.; Cao, S.; Du, W. Road surface crack detection method based on improved YOLOv5 and vehicle-mounted images. *Measurement* **2024**, *229*, 114443. [CrossRef]
37. Shao, Y.; Zhang, D.; Chu, H.; Zhang, X.; Rao, Y. Review of YOLO Object Detection Based on Deep Learning. *J. Electron. Inf. Technol.* **2022**, *44*, 3697–3708.
38. Dong, W.; Liang, H.; Liu, G.; Qiang, H.; Xu, Y. A Review of Deep Convolution Applied to Object Detection Algorithms. *Comput. Sci. Explor.* **2022**, *16*, 1025–1042.
39. *ISO 8601:2004*; Data Elements and Interchange Formats—Information Interchange—Representation of Dates and Times; International Organization for Standardization: Geneva, Switzerland, 2004.
40. Xiong, Y. *Intelligent Identification Technology for Drainage Pipe Network Defects Based on Improved YOLO v5*; Beijing University of Architecture: Beijing, China, 2023.
41. Vaswani, A.; Shazeer, N.; Parmar, N.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. *arXiv* **2017**, arXiv:1706.03762. [CrossRef]
42. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef]
43. Liu, Y.; Shao, Z.; Teng, Y.; Hoffm, Y. NAM: Normalization-based Attention Module. *arXiv* **2021**, arXiv:2111.12419. [CrossRef]
44. Liu, Y.; Shao, Z.; Hoffmann, N. Global Attention Mechanism: Retain Information to Enhance Channel-Spatial Interactions. *arXiv* **2021**, arXiv:2112.05561. [CrossRef]
45. Yang, Y.B. *SA-Net: Shuffle Attention for Deep Convolutional Neural Networks*; IEEE: Toulouse, France, 2021. [CrossRef]
46. Yang, L.; Zhang, R.Y.; Li, L.; Xie, X. SimAM: A Simple, Parameter-Free Attention Module for Convolutional Neural Networks. International Conference on Machine Learning. *PMLR* **2021**, *139*, 11863–11874.
47. Zheng, Z.; Wang, P.; Ren, D.; Liu, W.; Ye, R.; Hu, Q.; Zuo, W. Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation. *arXiv* **2020**, arXiv:2005.03572. [CrossRef]
48. Rezatofighi, H.; Tsoi, N.; Gwak, J.Y.; Sadeghian, A.; Reid, I.; Savarese, S. *Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression*; IEEE: Toulouse, France, 2019. [CrossRef]
49. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. *arXiv* **2019**, arXiv:1911.08287. [CrossRef]
50. 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]
51. Gevorgyan, Z. SIoU loss: More powerful learning for bounding box regression. *arXiv* **2022**, arXiv:220512740.
52. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *42*, 2999–3007. [CrossRef] [PubMed]
53. Zhang, H.; Wang, Y.; Dayoub, F.; Sunderhauf, N. VarifocalNet: An IoU-aware Dense Object Detector. *arXiv* **2020**, arXiv:2008.13367. [CrossRef]
54. Leng, Z.; Tan, M.; Liu, C.; Cubuk, C.; Shi, X.; Cheng, S.; Anguelov, D. PolyLoss: A Polynomial Expansion Perspective of Classification Loss Functions. *arXiv* **2022**, arXiv:2204.12511. [CrossRef]
55. Zhang, J.; Bi, Z.; Yan, Y.; Wang, P.; Hou, C.; Lv, S. Rapid Recognition of Greenhouse Tomatoes Based on Attention Mechanism and Improved YOLO. *Trans. Chin. Soc. Agric. Mach.* **2023**, *54*, 236–243.

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.