applied sciences

MDPI

Article

Improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses

Rio Arifando *, Shinji Eto and Chikamune Wada

Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu-ku, Kitakyushu 808-0196, Japan; eto.shinji786@mail.kyutech.jp (S.E.); wada@brain.kyutech.ac.jp (C.W.)
* Correspondence: arifandorio@gmail.com

Abstract: Object detection is crucial for individuals with visual impairment, especially when waiting for a bus. In this study, we propose a lightweight and highly accurate bus detection model based on an improved version of the YOLOv5 model. We propose integrating the GhostConv and C3Ghost Modules into the YOLOv5 network to reduce the number of parameters and floating-point operations per second (FLOPs), ensuring detection accuracy while reducing the model parameters. Following that, we added the SimSPPF module to replace the SPPF in the YOLOv5 backbone for increased computational efficiency and accurate object detection capabilities. Finally, we developed a Slim scale detection model by modifying the original YOLOv5 structure in order to make the model more efficient and faster, which is critical for real-time object detection applications. According to the experimental results, the Improved-YOLOv5 outperforms the original YOLOv5 in terms of the precision, recall, and mAP@0.5. Further analysis of the model complexity reveals that the Improved-YOLOv5 is more efficient due to fewer FLOPS, with fewer parameters, less memory usage, and faster inference time capabilities. The proposed model is smaller and more feasible to implement in resource-constrained mobile devices and a promising option for bus detection systems.

Keywords: bus detection; visual impairment; YOLOv5; Ghostnet; SimSPPF; slim scale model

check for updates

Citation: Arifando, R.; Eto, S.; Wada, C. Improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses. *Appl. Sci.* **2023**, *13*, 5802. https://doi.org/10.3390/app13095802

Academic Editors: Ekin Ozer, Shizhi Chen, Jingfeng Zhang, Zilong Ti and Xiaoming Lei

Received: 18 April 2023
Revised: 2 May 2023
Accepted: 3 May 2023
Published: 8 May 2023

1. Introduction

Object recognition and detection has been an important problem for a long time for many different industries, including the medical field, the security industry, and the transportation sector. This issue, however, takes on a greater level of significance for people who are blind or have some other form of visual impairment. According to estimates by the World Health Organization (WHO), at least 2.2 billion people globally suffer from some form of near or distance vision impairment, with associated annual productivity losses of around $411 billion [1]. People who are visually impaired face a wide variety of difficulties on a daily basis. These challenges can range from difficulties with mobility and orientation to difficulties accessing information and services. One of the most significant issues that individuals with visual impairment face on a daily basis is related to the use of public transportation, with the difficulty of getting to their destination, followed by the inconvenience and then safety concerns [2]. In the context of public bus transportation, people with visual impairment require information about their surroundings and any visible details at bus stops and terminals, such as schedules and routes, in order to use the system independently and safely. However, the majority of individuals with visual impairment face challenges when it comes to taking the correct bus and disembarking at the correct destination [3]. As a result, they may have to rely on others for transportation, limit their travel, or choose simpler activities that do not require extensive travel.

In recent years, the advancement of technology has enabled the development of more accurate and efficient object detection and recognition systems. Therefore, bus detection has been an area of active research. RFID and embedded systems are widely used for bus detection in intelligent transportation systems. These systems use RFID tags embedded in buses to communicate with stationary RFID readers at bus stops, enabling real-time bus detection and location tracking [4–7]. For instance, a study by Raj et al. [8] proposed an RFID-based bus detection system using passive RFID tags and RFID readers at bus stops. Another study proposed a mechatronic system architecture to improve transportation for people with visual impairment by combining RFID and wireless sensor networks [9]. However, RFID and embedded systems also have their limitations. These systems require specific hardware and can be expensive to install and maintain [10]. They can also be affected by environmental factors such as interference from nearby metal objects, signal reflections, and other sources of noise [11].

However, deep learning-based methods, such as those using CNNs and object detection models, are becoming increasingly popular for object detection due to their high accuracy and real-time performance. These methods can accurately detect and recognize buses in real-world scenarios, even in complex urban environments. They can also be used on mobile devices, making them accessible to individuals with visual impairment. For example, Liang et al. [12] introduced a deep learning model, a lightweight and energy-efficient transportation model detection system, using only accelerometer sensors in smartphones. Another study conducted a comparative study to evaluate the effectiveness of two deep learning models, AlexNet and Faster R-CNN, for detecting vehicles in an urban video sequence, which produced positive results, and the tests provided important insights into the architectures and strategies used to implement these models for video detection [13].

The deep-learning-based object detection algorithm has made significant progress in detection accuracy and speed. There are two types of deep learning target detection algorithms: those with two stages and those with only one. After feature extraction in the first stage, the two-stage detection algorithm will generate a region proposal that may contain the target to be detected. The second step involves locating and classifying the data using a convolutional neural network with high detection accuracy, such as R-CNN [14], Fast R-CNN [15], Faster R-CNN [16], SPPNet [17], etc. In contrast to two-stage algorithms, which require candidate regions to be generated before positioning and classification can be performed, one-stage detection algorithms perform both operations simultaneously. The detection speed of algorithms such as YOLO [18], SSD [19], RetinaNet [20], and FCOS [21] is better than that of the two-stage algorithm, but the accuracy is lower.

YOLOv5 is an object detection algorithm that belongs to the YOLO (You Only Look Once) family of algorithms. It was released in June 2020 and quickly gained attention in the computer vision community due to its improved performance in speed, accuracy, and model size [22]. In terms of speed, YOLOv5 can detect objects in real time or near real time, meaning it can process a video stream frame by frame and return object detections almost instantly. The combination of fast speed and high accuracy makes YOLOv5 a popular choice for object detection tasks in various fields, including surveillance [23], robotics [24], and autonomous driving [25]. In these applications, quick and accurate object detection is crucial for the success of the system. In terms of fast object detection systems, Huawei's Noah's Ark Lab proposed a new end-to-end neural network architecture called GhostNet, which aimed to improve the detection speed. This architecture was included in CVPR2020 [26]. The GhostNet paper proposed a Ghost module that could generate more feature maps from cheaper operations than conventional convolution operations, thereby reducing the number of floating-point operations (FLOPs) in the network. The experimental results showed that the proposed Ghost module decreased the computational cost of general convolutional layers while maintaining similar recognition performance.

Fast object detection and high accuracy are important factors to consider when developing a bus detection model, especially for people with visual impairment. A lightweight and high-accuracy model is desirable to ensure real-time performance and ease of use on mobile devices. To address this issue, this study proposes a bus detection model that is lightweight, accurate, and suitable for use on mobile devices. The proposed model aims to provide an efficient and reliable solution for individuals with visual impairment to detect and recognize buses in real time. By combining YOLOv5 and GhostNet, we can offer a promising solution for real-time bus detection. Our solution leverages YOLOv5's high speed and accuracy and GhostNet's ability to reduce computational costs. This combination can efficiently detect buses in real-time video streams from different angles and distances. Furthermore, our solution addresses the computational challenges associated with object detection by using GhostNet to reduce the computational burden of YOLOv5. This enables our solution to be deployed on resource-constrained devices, such as mobile phones or embedded systems. As a result, we can implement real-time bus detection on edge computing systems, without relying on cloud computing resources. Moreover, the SimSPPF module is added to the YOLOv5 backbone to enhance the computational efficiency and object detection capabilities. The Slim scale detection model is also developed by modifying the original YOLOv5 structure, making it more efficient and faster for real-time object detection applications.

In summary, our proposed combination has the potential to provide an efficient and accurate solution for real-time bus detection for individuals with visual impairments, as real-time bus detection can aid in navigation and improve accessibility for those with disabilities. Additionally, our solution can also be beneficial for smart city transportation systems and autonomous driving. The ability to reduce the computational costs and deploy on resource-constrained devices makes this combination an attractive option for edge computing systems.

The following bullet points summarize the key contributions of the research study, which proposes a lightweight and high-accuracy bus detection model:

- We proposed a lightweight and high-accuracy bus detection model based on an improved YOLOv5 model;
- We integrated the GhostConv and C3Ghost Modules into the YOLOv5 network to reduce the number of parameters and floating-point operations per second (FLOPs), ensuring the detection accuracy while reducing the model parameters;
- We added the SimSPPF module to replace the SPPF in the YOLOv5 backbone for increased computational efficiency and accurate object detection capabilities;
- We developed a Slim scale detection model by modifying the original YOLOv5 structure to make the model more efficient and faster, which is critical for real-time object detection applications;
- The experimental results showed that the Improved-YOLOv5 outperformed the original YOLOv5 in terms of the precision, recall, and mAP@0.5;
- Further analysis of the model complexity revealed that the Improved-YOLOv5 was more efficient due to fewer FLOPS, fewer parameters, less memory usage, and faster inference time capabilities;
- The proposed model is smaller and more feasible to implement in resource-constrained mobile devices and a promising option for bus detection systems.

The structure of this paper consists of four main sections, with each section covering essential topics related to the proposed model for bus detection. Section 2 elaborates on the principles of YOLOv5 and the Improved-YOLOv5, providing an in-depth discussion of the technical aspects of the proposed model. Moving forward, Section 3 focuses on the model training process and experimental results. The section highlights the various experiments performed to evaluate the proposed model's performance. Finally, in Section 4, the authors conclude the presented work, summarizing the significant findings and contributions of the study. This section highlights the strengths of the proposed model, including its efficiency, accuracy, and feasibility.

2. Materials and Methods

2.1. YOLOv5

Ultralytics developed the YOLOv5 object detection algorithm, which is based on the You Only Look Once (YOLO) family of models. Due to its high accuracy and quick inference speed, YOLOv5 has quickly gained popularity since its initial release in June 2020. YOLOv5 is available in the following architectures: YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. These architectures vary in terms of the network depth and feature map width, allowing users to select the one that best meets their particular requirements. To meet the industrial requirements for bus detection algorithms, such as real-time detection and simple deployment, we chose YOLOv5n, which has the smallest model size and fastest detection speed, as the infrastructure. Figure 1 depicts the network architecture of YOLOv5, which consists primarily of four components: Input, Backbone, Neck, and Head.

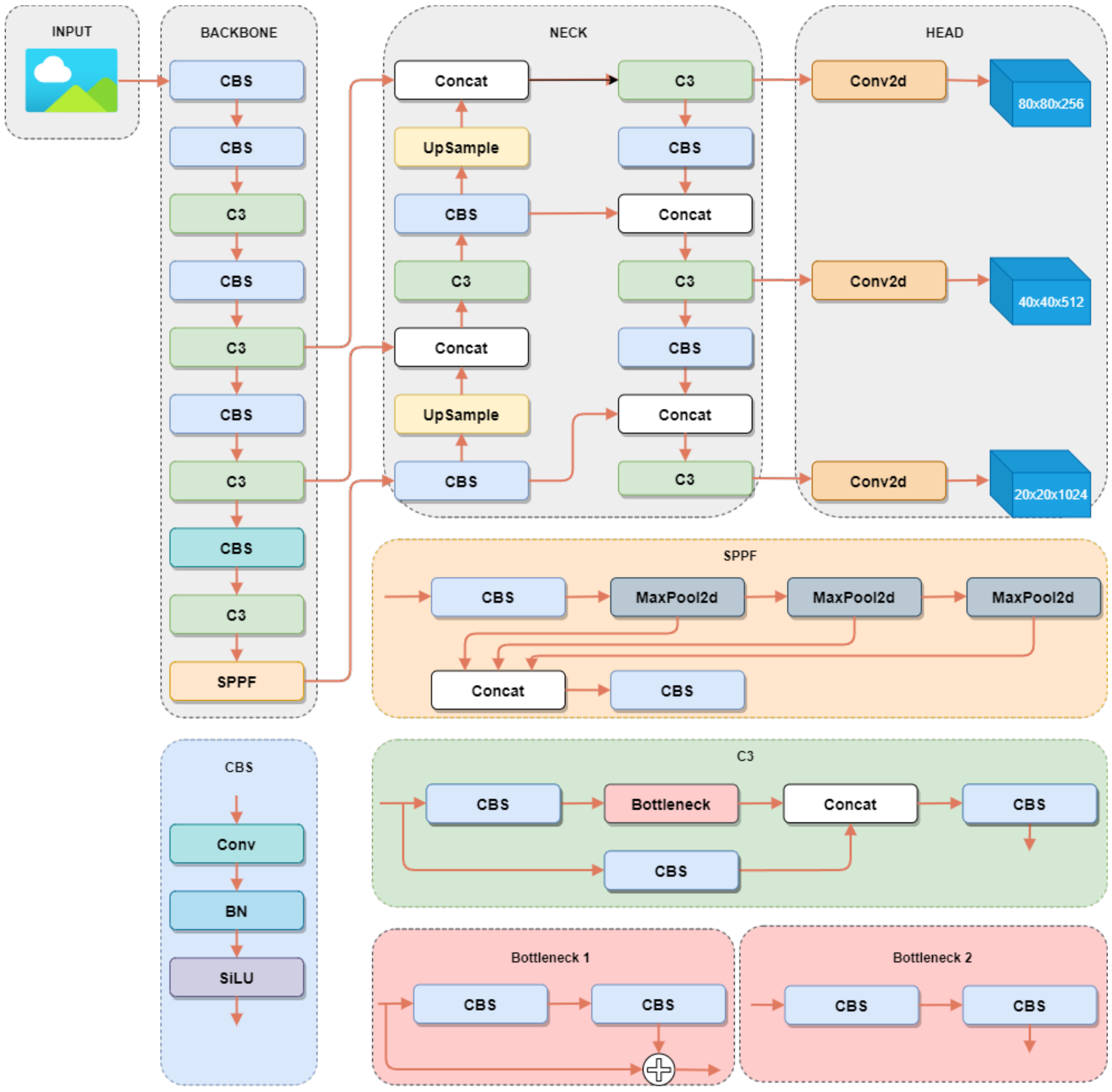


Figure 1. The YOLOv5 model structure.

Input: The Input component is in charge of receiving and preparing the input image for processing. As part of the input, an image preparation step, such as adaptive image scaling, mosaic data enhancement, or adaptive anchor frame calculation, is frequently performed. When images of varying sizes are entered into the Input, the adaptive image scaling technique first normalizes the image size to 640 $\times$ 640. It then chooses four images at random for cropping, scaling, and layout processing using mosaic data.

Backbone: The Backbone component is the network's main feature extractor. It is in charge of extracting features from the input image through the use of a series of convolutional layers. The backbone component in YOLOv5 consists of a modified version of the

Conv module, C3 module, and Spatial Pyramid Pooling—Fast (SPPF [17]) module. Convolution, batch normalization, and SiLU activation are all included in the Conv module. The C3 module is based on the CSPNet [27] concept and includes three standard convolutional layers as well as multiple bottleneck modules. The SPPF module employs serial maxpool to perform multiscale fusion in order to expand the receptive field's feature map.

Neck: The Neck component is in charge of fusing the backbone features and improving their representational power. For its Neck section, YOLOv5 employs the Feature Pyramid Network (FPN [28]) and Pixel Aggregation Network (PAN [29]) structures. The FPN structure transmits semantic information from top to bottom via an upsampling operation, while the PAN structure transmits location information from bottom-to-top via a downsampling operation. By combining the two structures, the network is able to fuse more feature information, forming a multiscale feature fusion module that can retain both large-scale and small-scale target feature information.

Head: The Head component is in charge of predicting the bounding boxes and class probabilities for each object in the input image. The loss function of the bounding box is CIOU Loss [30], and the Non-Maximum Suppression (NMS [31]) is used to screen the multi-object box and output the predicted image.

2.2. Improved-YOLOv5

2.2.1. SimSPPF (Simplified SPPF)

The SPPF (Spatial Pyramid Pooling with Factorization) module in YOLOv5 is essential for capturing context information at multiple scales by pooling features with different window sizes. The CBS (Conv-BN-SiLU) structure is used by the original SPPF module in YOLOv5, which is a convolution layer followed by batch normalization and the Sigmoid Linear Unit (SiLU) function. SimSPPF (Simplified SPPF) was proposed in YOLOv6 [32], an upgraded version of the SPPF that uses the CBR (Conv-BN-ReLU) structure in the SPP block, where the convolution layer is followed by batch normalization and ReLU activation. When deciding between YOLOv5 and YOLOv6, the choice should be made based on priorities and constraints. If the primary concern is accuracy, YOLOv5 with the original SPPF module is the better option as it captures context information at multiple scales effectively using the CBS structure with the SiLU activation function. If computational efficiency is the top priority, then YOLOv6 with the SimSPPF module is preferable as it uses the CBR structure with the ReLU activation function, which is less computationally expensive. While SimSPPF has slightly lower accuracy, it might be an acceptable tradeoff for our application. The goal of this modification was to reduce the computational complexity of the SPP block while maintaining the accuracy. The main distinction between the SPPF and the SimSPPF is the activation function used after the batch normalization layer in the SPP block's convolution layer. The SimSPPF employs the ReLU activation function, whereas the SPPF employs the SiLU activation function. The authors discovered that heavily implementing the SimSPPF only resulted in a small increase in accuracy with increased computational complexity, while also improving the FPS. Figure 2 depicts the structure of the SimSPPF Module.

The SimSPPF module was used in our model in this research project. The decision to use the SimSPPF module was based on the observed advantages of SimSPPF over SPPF, such as the reduced computational complexity and higher FPS while maintaining good object detection accuracy. The equations are shown in Equations (1)–(5) [33].

$$F1 = CBR(F) \tag{1}$$

$$F2 = \text{ Maxpooling } (F1) \tag{2}$$

$$F3 = \text{ Maxpooling } (F2) \tag{3}$$

$$F4 = \text{ Maxpooling } (F3) \tag{4}$$

$$F5 = CBR([F; F2; F3; F4]) \tag{5}$$

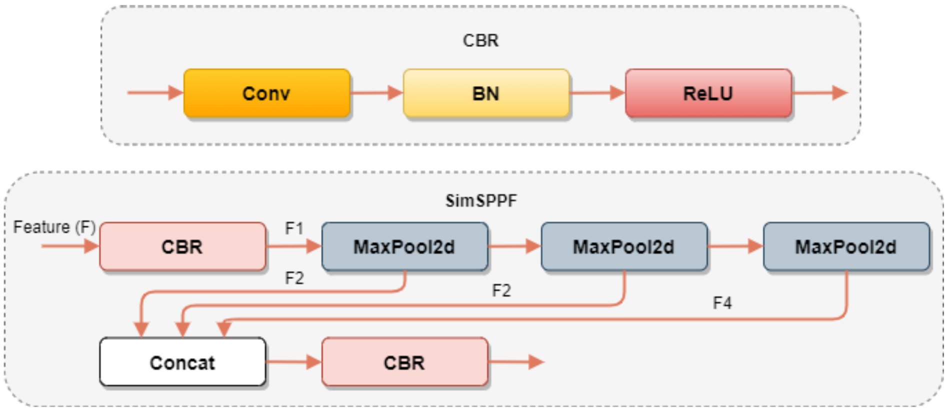


Figure 2. Structure of the SimSPPF Module.

2.2.2. GhostNet Module

GhostNet is a neural network architecture that seeks to balance high accuracy with a low computational cost. It was created to address the shortcomings of traditional deep neural network models, which are too complex to run on resource-constrained devices [26]. In this research, the YOLOv5 architecture was modified by replacing the original C3 module with the C3Ghost module from GhostNet. This module was created to reduce the model's computational complexity by sharing weights across multiple convolution layers. As a result, the model had fewer parameters while maintaining its accuracy. This was especially important for a real-time application such as bus detection, where detection speed is critical. We also replaced the CBS module with the Ghost conv module. The Ghost conv module is a lightweight alternative to the traditional convolution layer that reduces the number of model parameters. This enabled the model to make better use of the computational resources, making it more suitable for deployment on resource-constrained devices. Figure 3 depicts the operation of the ghost module.

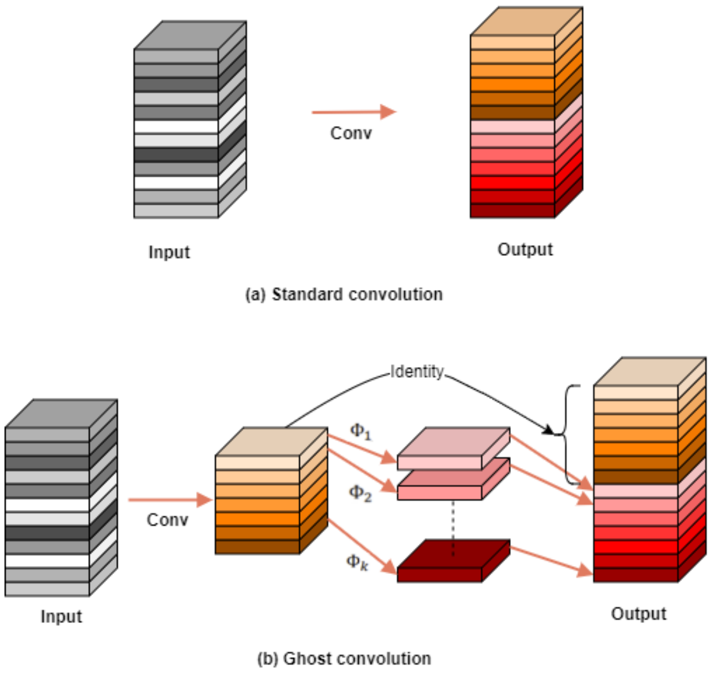


Figure 3. Schematic diagram of the Ghost module.

For the input image, $X \in \mathbb{R}^{c \times h \times w}$, where c, h, and w are the number of input channels, input image height, and input image width, respectively. In general, the operations of an arbitrary convolutional layer for producing n feature maps can be written as Equation (6).

$$Y = X * f + b, \tag{6}$$

where $*$ represents the convolution operation, b represents the bias, $Y \in \mathbb{R}^{h' \times w' \times n}$ represents the output feature map with n channels, and $f \in \mathbb{R}^{c \times k \times k \times n}$ represents the convolution filters in this layer. Furthermore, h' and w' are the height and width of the output feature maps, respectively, and $k \times k$ is the kernel size of the convolution filters f [26]. The required number of FLOPs during this convolution procedure can be calculated as $n \cdot h' \cdot w' \cdot c \cdot k \cdot k$, which is often in the hundreds of thousands because the number of filters n and the number of channels c are generally very large.

We can discover that some features are similar by visualizing some feature maps, and we can obtain them by performing some simple operations instead. Equation (7) can be used to generate m intrinsic feature maps $Y' \in \mathbb{R}^{h' \times w' \times m}$, where $f' \in \mathbb{R}^{c \times k \times k \times m}$ represents the convolution filters, and it is similar to Equation (6). However, we only work with partial convolutions and generate the remaining feature maps with a linear operation, as shown in Equation (8).

$$Y' = X * f' + b \tag{7}$$

$$y_{i,j} = \Phi_{i,j}(y'_i), \forall i = 1, \ldots, m, j = 1, \ldots, s, \tag{8}$$

where y'_i is the i-th intrinsic feature map in Y', and $\Phi_{i,j}$ is the linear operation for y'_i that generates the j-th ghost feature map y_i, j.

In general, the parameter p utilizes standard 2D convolution, as shown in Equation (9), and the parameter P' is shown in Equation (10) with group convolution, where h_1, w_1, C_1, C_2, and g are the filter height, the filter width, the input channels, the output channels, and the number of groups, respectively. We can obtain similar feature maps with $1/\mathrm{g}$ times the number of parameters.

$$P = h_1 \times w_1 \times C_1 \times C_2 \tag{9}$$

$$P' = h_1 \times w_1 \times C_1 \times C_2 \times \frac{1}{g} \tag{10}$$

The Conv and GhostConv modules both have the same number of input and output channels. The number of input channels is $C1$, and the number of output channels is $C2$. Figure 4 depicts the detailed structure of YOLOv5 with GhostNet.

2.2.3. Slim Scale Detection Layer

The image is processed through a series of convolutional layers in the original YOLOv5 framework to generate feature maps at multiple scales. The detection head then uses these feature maps to make predictions about the objects in the image. The YOLOv5 framework, in particular, generates three levels of feature maps: P3 (80 × 80 × 256), P4 (40 × 40 × 512), and P5 (20 × 20 × 1024). The input image is downsampled by factors of 8, 16, and 32 to produce these feature maps. This means that the highest level feature maps, P5, have the lowest resolution but cover the most area of the input image. In this study, we removed the P5 feature map from the YOLOv5 framework. As a result, the detection head only received feature maps from the P3 and P4 levels, which have higher resolutions but cover smaller areas of the input image.

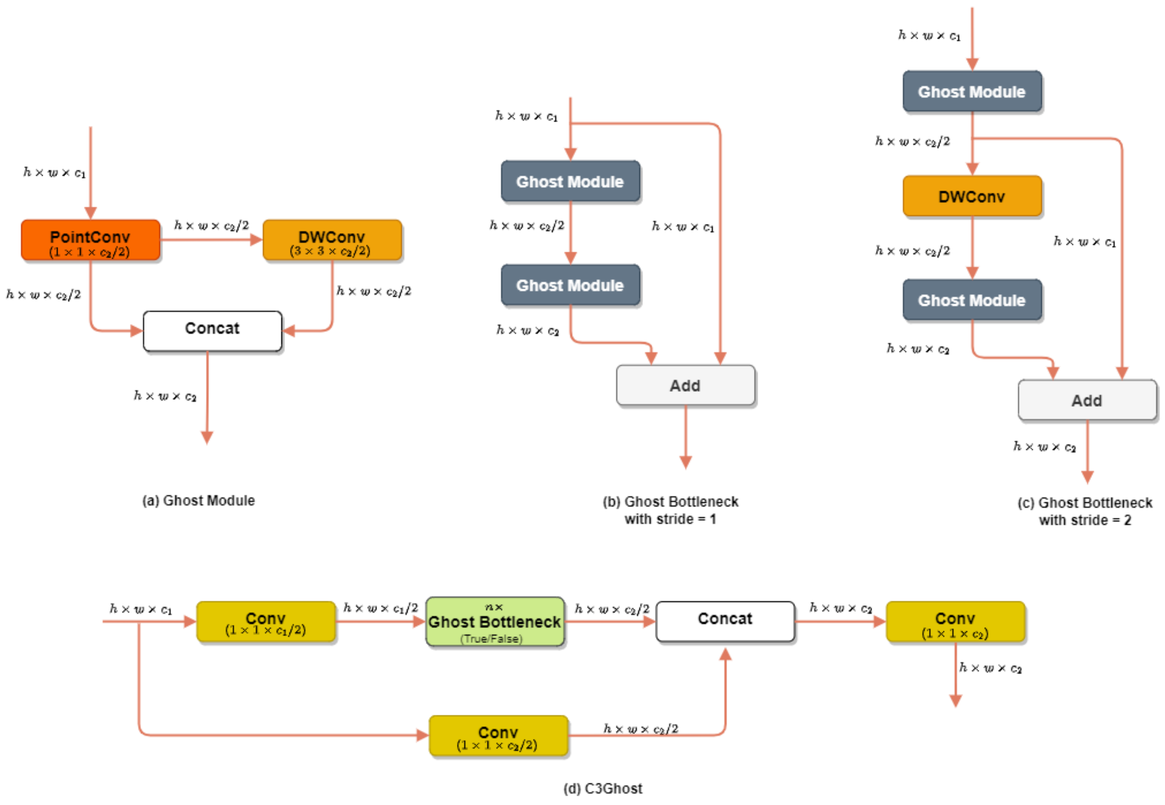


Figure 4. The structure of YOLOv5 with GhostNet. (**a**) Ghost Module. (**b**) Ghost Bottleneck with stride = 1. (**c**) Ghost Bottleneck with stride = 2. (**d**) C3Ghost.

This change was made because we are specifically interested in detecting buses, which tend to be larger objects in the image. By removing the P5 feature map, we prioritized the higher resolution feature maps capable of providing more detailed information about the buses in the image. Removing the P5 feature map and using only the higher resolution P3 and P4 feature maps in the Improved-YOLOv5 structure was expected to improve the accuracy of detecting larger objects such as buses in the input image. This is because the higher resolution feature maps contain more detailed information about the objects in the image and can thus provide better localization and recognition of the objects. By prioritizing the P3 and P4 feature maps, the model can potentially reduce the false positives and increase the precision of object detection, especially for larger objects such as buses.

In addition to improving the accuracy, removing the P5 feature map can also result in a more efficient model. The P5 feature map is the largest and thus requires the most computation to process. By removing it, we can reduce the computational resources required to run the model, making it more efficient and faster. This can be particularly important for applications that require real-time object detection, where speed and efficiency are crucial. It is important to note, however, that removing the P5 feature map may make detecting smaller objects in the image more difficult. This is due to the fact that the P5 feature map covers the largest area of the input image and can thus capture information about objects that are further away from the image's center. The Improved-YOLOv5 structure is shown in Figure 5.

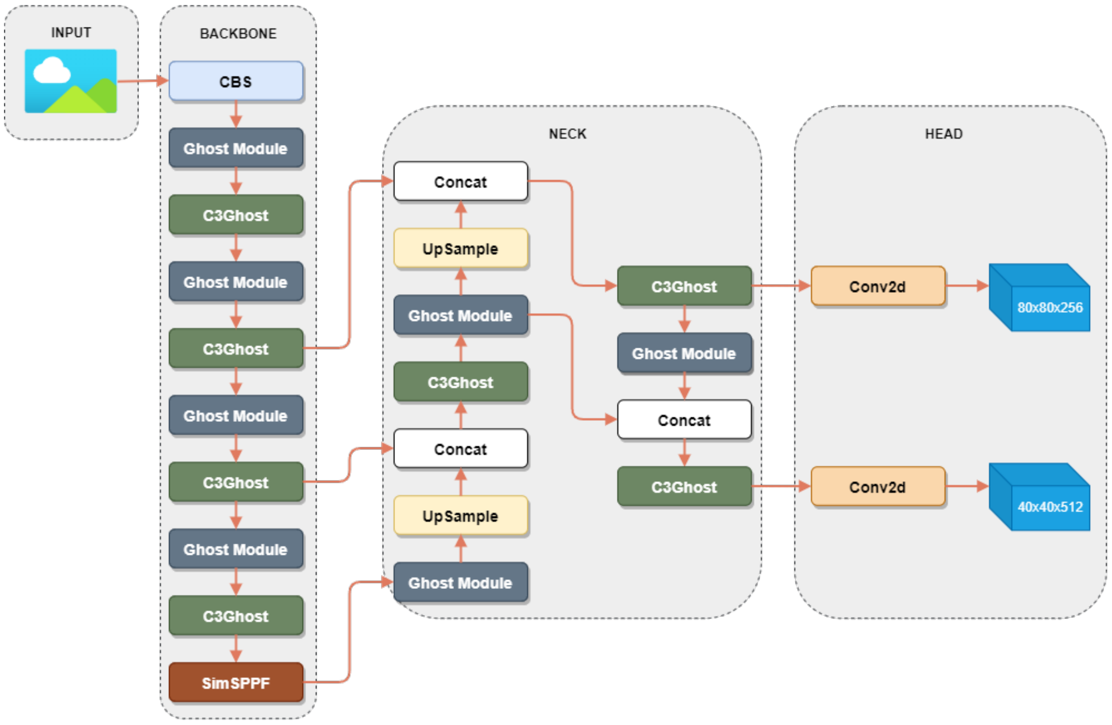


Figure 5. The structure of the Improved-YOLOv5.

3. Experiment and Results

3.1. Datasets

The dataset used in this study was composed of images that depicted buses in various positions, orientations, perspectives, and environmental conditions. The data collection process was conducted meticulously to ensure high-quality data. The images were collected using a mobile phone camera, as well as from the internet. The purpose of this dataset was to aid people with visual impairment in identifying nearby buses, and as such, it was focused on bus detection. The dataset contained three main classes: bus, door, and route. Figure 6 provides an example of an image in the dataset, which shows a bus during daylight conditions. The image contains one instance of a bus, two instances of doors, and one instance of a route. Each image in the dataset was labeled with bounding boxes that indicated the location and area of each instance. This labeling information was then utilized by the model to conduct object detection within the images.

To elaborate further on the nature of the dataset, it is worth noting that it was a labeled dataset, which means that each image was annotated with bounding boxes that precisely identified the locations of the buses, doors, and routes present within the images. The dataset contained a variety of environmental conditions, which included different lighting conditions, weather conditions, and backgrounds. Furthermore, the dataset contained instances of buses with different colors, shapes, and sizes, which made it a diverse dataset that could be used to train object detection models that were robust and adaptable to various real-world scenarios.

The dataset consisted of 1602 images, with 75% used as the training set, which was a total of 1202 images, 20% used as the validation set, which was a total of 320 images, and 5% used as the testing set, which was a total of 80 images. The images were labeled using the roboflow.com website with YOLOv5 PyTorch output format, which is .txt, with a total of 5148 annotations and an average of 3.2 annotations per image. The annotation details per class were 1745 instances for bus, 2254 instances for door, and 1149 instances for route. Figure 7a presents a graph that shows the number of annotations per class in the dataset.

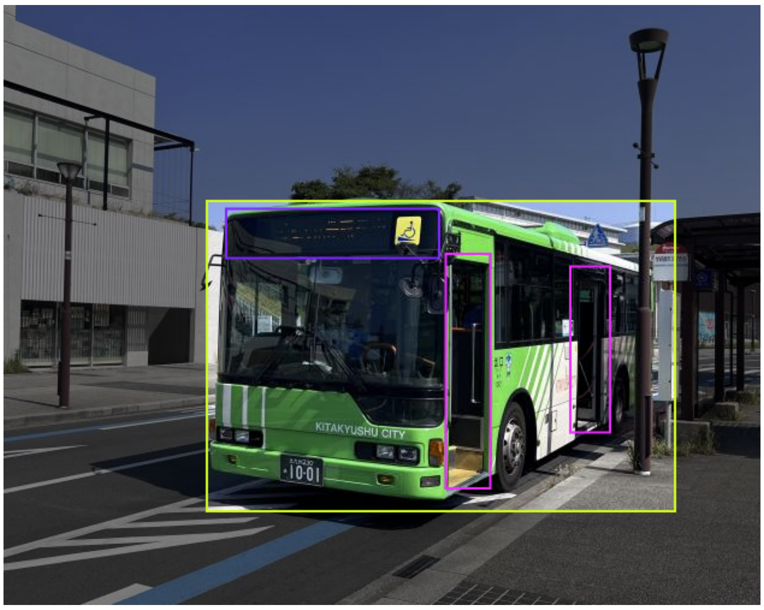


Figure 6. Example image of the dataset.

Figure 7b shows the distribution of the bounding boxes in the dataset, by visualizing the location and size of each bounding box. This helps in understanding the distribution of the bounding boxes in the dataset and ensures that there is enough variation in the position and size of objects for the model to recognize. Figure 7c,d, show the statistical distribution of the position and size of the bounding boxes, which shows how the distribution of the bounding boxes is spread across the dataset. This graph helps in determining whether the bounding boxes have an even distribution or whether there are parts of the dataset that are heavier. This is important to ensure that the model will not have problems detecting objects in the image, because all objects vary in size and position.

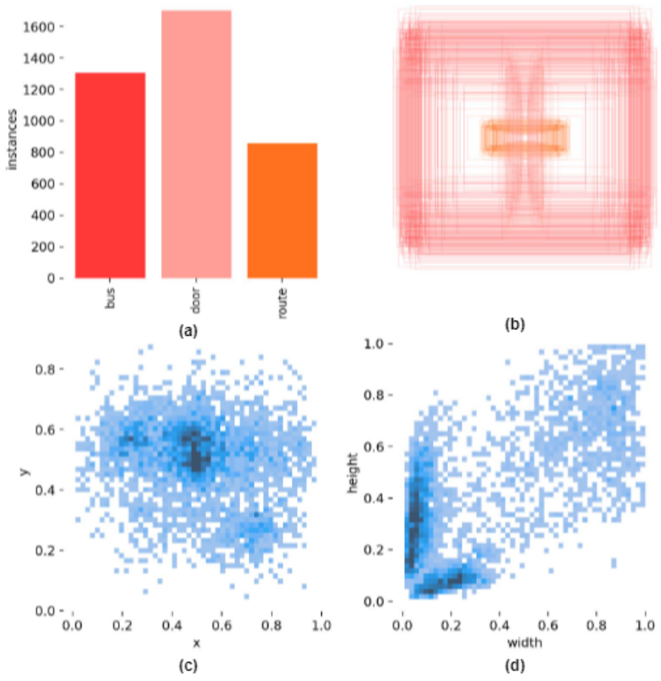


Figure 7. Visualization of the dataset. (**a**) Number of annotations per class. (**b**) Visualization of the location and size of each bounding box. (**c**) The statistical distribution of the bounding box position. (**d**) The statistical distribution of the bounding box sizes.

This study used the default data augmentation techniques from the YOLOv5 model, including HSV-Hue augmentation, image translation, image scale, image flip left–right, and image mosaic, as part of the model training process. This technique helped to expand the amount of data used during the training process, strengthened the model to perform better object detection in images and reduce the risk of overfitting, and strengthened the model to perform object detection in different environments. All images in the dataset were also preprocessed by auto-orienting and resizing to a size of 640 × 640, with an average image size of 0.41 MP. This information helped to ensure that the model could detect the objects in the image properly, because the objects had various sizes and positions, and the bounding boxes were evenly distributed.

3.2. Experimental Environment

The experimental environment in this study required the use of several critical components, including a GPU with CUDA support and the PyTorch framework. To perform the learning process and object detection quickly and accurately, the YOLOv5 deep learning model required parallel computing. As a result, a GPU and CUDA were used to accelerate the computing, while the PyTorch framework was used to build and train models. Table 1 shows the configuration details for this experimental environment.

Table 1. Experimental setting.

Device	Configuration
System	Windows 11 Pro
CPU	12th Gen Intel(R) Core (TM) i7-12700 2.10 GHz
GPU	NVIDIA GeForce RTX 3060 Ti
Framework	PyTorch 1.13.1
IDE	Visual Studio Code
Python version	version 3.7.9

3.3. Evaluation Metrics

Model evaluation is very important to determine the model performance and model compatibility with the research objectives. There are several evaluation metrics that can be used in bus object detection, which include evaluation metrics related to the accuracy, speed, and efficiency of the model. Evaluation metrics related to accuracy are usually the main focus in detecting bus objects, because the model must be able to identify buses with high accuracy. The basic evaluation metrics that are often used are precision (P) and recall (R). Precision is the ratio between the number of true positives (TP) and the number of positive predictions (TP + false positives (FP)). This metric measures how accurately the model predicts the positive class. Recall is the ratio between the number of true positives (TP) and the number of true positive classes (TP + false negatives (FN)). This metric measures how many positive classes the model can identify. Apart from precision and recall, there are other evaluation metrics that are more commonly used in object detection, namely average precision (AP) and mean average precision (mAP). The AP measures how well the model finds relevant objects and ignores the irrelevant objects. The AP is calculated by plotting the precision–recall curve and calculating the area under the curve. The mAP, on the other hand, is the average of the APs for all the object classes found by the model. The mAP provides a more comprehensive picture of the model's performance because it measures the accuracy for all classes of objects, not just for one class. The equations are shown in Equations (11)–(14).

$$P = \frac{TP}{TP + FP'} \tag{11}$$

$$R = \frac{TP}{TP + FN'} \tag{12}$$

$$AP = \int_0^1 P(r)dr \tag{13}$$

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{k} \tag{14}$$

In addition to the evaluation metrics related to accuracy, there are evaluation metrics related to the model speed and efficiency. For example, gigaflops per second (GFLOPS) measures the number of floating-point operations a model can perform in one second, and the inference time measures the time it takes a model to process a single image or video. The evaluation metrics related to the model speed and efficiency are critical in real-time applications such as object detection in video.

Apart from that, there are other evaluation metrics such as the number of model parameters and the model size. The number of model parameters is the number of parameters (weights) used by the model to learn patterns in the training data. The more parameters used, the more complex the model. This metric gives an idea of the complexity of the model. Model size, on the other hand, measures the size of the model file in bytes or megabytes (MB). This metric gives an idea of the complexity and practicality of the model, especially in applications that require fast and space-efficient file transfers.

3.4. Training Results and Analysis

In this study, the accuracy of two models was compared: YOLOv5n and Improved-YOLOv5. The baseline model for comparison was the YOLOv5n because it has the same depth and width structure as the Improved-YOLOv5. To avoid overfitting, the YOLOv5n and Improved-YOLOv5 models were trained for 1000 epochs with early stopping. Early stopping is a regularization technique that halts training if the model's performance on a validation set stops improving after a certain number of epochs, called the "patience". The patience was set to 100 epochs in this case. Both models' performance on the validation set was monitored during training. When the models failed to improve their validation performance after 100 epochs, the training was halted to avoid overfitting. The Improved-YOLOv5, on the other hand, was stopped earlier than YOLOv5n, at epoch 816 versus epoch 517, indicating that it was better able to learn and generalize the data features. The ability of the Improved-YOLOv5 to continue training for a longer period of time suggests that it was less vulnerable to overfitting than YOLOv5n. This could be because the Improved-YOLOv5 had a better architecture or a more appropriate hyperparameter configuration, which resulted in better model generalization. Ultimately, the improved ability of the Improved-YOLOv5 to generalize its features and avoid overfitting may have led to the better performance compared to YOLOv5n. Comparison of the detection results of the different models is shown in Figure 8, which contains a visual comparison of the detection results obtained by the two models, YOLOv5n and Improved-YOLOv5. The figure shows how the two models performed in terms of detecting buses in images, and it can provide a clear comparison of their accuracy.

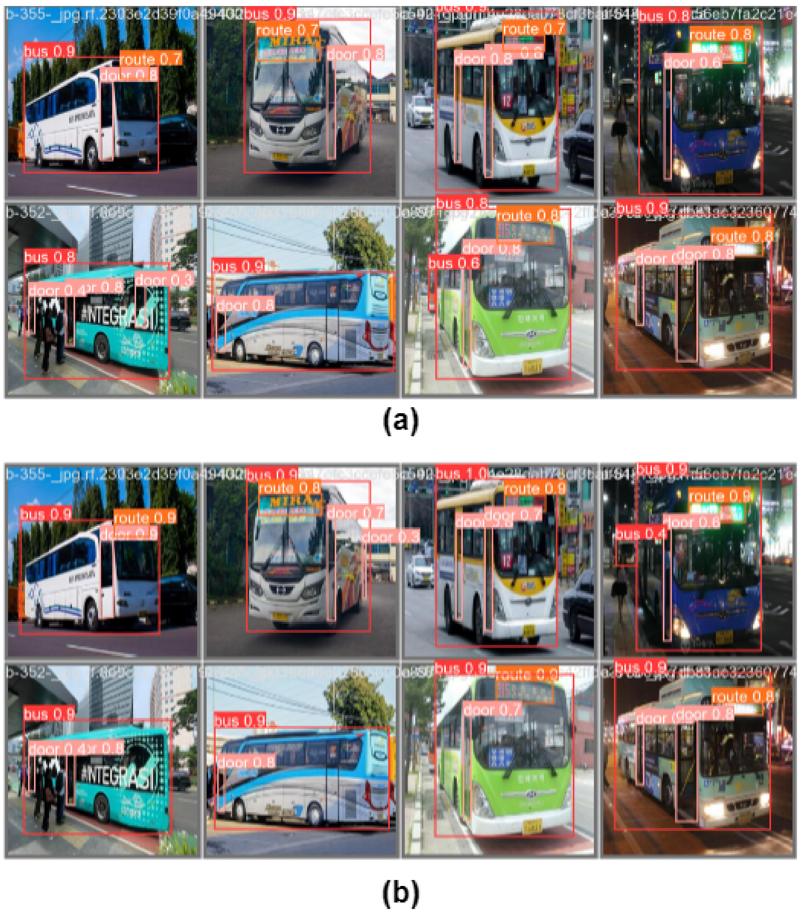


Figure 8. Comparison of the detection results of different models. (**a**) YOLOv5n model. (**b**) Improved-YOLOv5.

The results in Figure 9 provide insights into how the models learned over time and whether there were any particular trends or patterns in their performance. Additionally, Figure 10 shows a comparison of the models' accuracy. When the two models were compared, the Improved-YOLOv5 outperformed the YOLOv5n in terms of the precision and mAP@0.5. The precision of the Improved-YOLOv5 model was 0.939, while the precision of the YOLOv5n model was 0.921. This suggests that the Improved-YOLOv5 was better at identifying true positives, which is important for people with visual impairment because false positives can lead to missed buses, frustration, and potential safety hazards. Both models had high recall, indicating that they correctly identified the majority of the buses in the image. The recall for the YOLOv5n was 0.899, while the recall for the Improved-YOLOv5 was 0.895. This demonstrates that both models detected the majority of the buses in the images. The mean Average Precision (mAP) is a metric that measures how well the model performs at various IoU thresholds. Both models in this study had a good mAP@0.5, indicating that they could detect buses with a high level of accuracy. The Improved-YOLOv5, on the other hand, outperformed YOLOv5n, with a mAP@0.5 of 0.933 compared to YOLOv5n's score of 0.923. The Improved-YOLOv5 had a lower mAP@0.5:0.9 score of 0.619, while the YOLOv5n had a higher score of 0.628. This means that the YOLOv5n detected buses with greater accuracy at various IoU thresholds, especially at higher thresholds, than the Improved-YOLOv5.

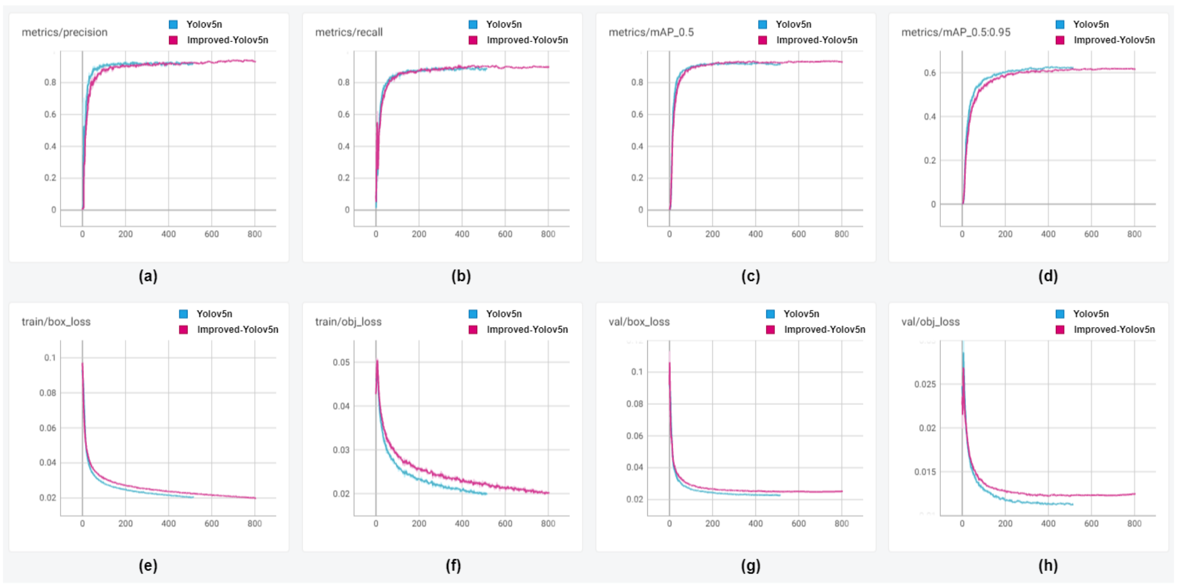
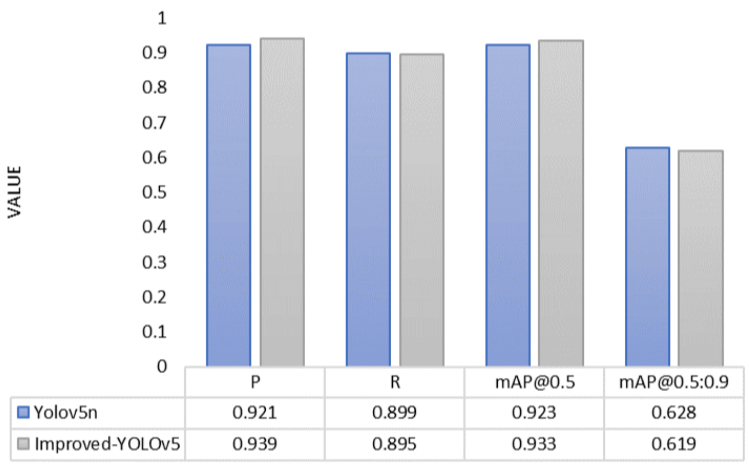


Figure 9. Comparison of the models' accuracy. (**a**) Precision. (**b**) Recall. (**c**) mAP@0.5. (**d**) mAP@0.5:0.9. (**e**) Train/box_loss. (**f**) Train/obj_loss. (**g**) Val/box_loss. (**h**) Val/obj_loss.



	P	R	mAP@0.5	mAP@0.5:0.9
Yolov5n	0.921	0.899	0.923	0.628
Improved-YOLOv5	0.939	0.895	0.933	0.619

Figure 10. Comparison of the models' accuracy.

In summary, the comparison revealed that the Improved-YOLOv5 outperformed the YOLOv5n in terms of the precision, recall, and mAP@0.5, indicating its ability to detect true positives with greater accuracy, detect the majority of objects in images, and detect objects with a high level of accuracy, respectively. YOLOv5n, on the other hand, had a higher mAP@0.5:0.9 score, indicating that it was better at detecting buses with greater accuracy at different IoU thresholds, particularly at higher thresholds. According to the findings, the Improved-YOLOv5 could be a better option for bus detection systems because it demonstrated better generalization and less overfitting, resulting in better performance. The difference in the mAP@0.5:0.9, on the other hand, highlights the need for additional research and testing to determine the best model for the specific application.

The comparison of the models' complexity is shown in Table 2. In this analysis, the model complexity comparisons were YOLOv5n and the Improved-YOLOv5, with YOLOv5n serving as the baseline. The goal was to determine which model was more efficient and less complicated, making it a more practical solution for bus detection systems. The number of parameters or the total number of learnable parameters in the model was the first metric considered. YOLOv5n had 1763,224 parameters, while the Improved-YOLOv5 only had 717,020 parameters. This demonstrates that the Improved-YOLOv5 was a simpler model to learn because it had fewer parameters to learn. This is advantageous because it means the model is less complex and has a lower risk of overfitting, which is necessary to ensure that the model generalizes well to new data. Furthermore, having fewer parameters means that the model requires less memory and storage, which can be beneficial in resource-constrained environments.

Table 2. Comparison of the models' complexity.

Model	Parameters	GFLOPs	Weight/MB
Yolov5n	1,763,224	4.1	3.72 MB
Improved-YOLOv5	717,020	2.1	1.76 MB

The second metric considered was GFLOPs, which measures the number of floating-point operations per second that the model can perform. The Improved-YOLOv5 outperformed the YOLOv5n in this case, with 2.1 GFLOPs versus 4.1 GFLOPs. This means that the Improved-YOLOv5 completed more operations in less time, which can be advantageous in applications where speed is critical, such as real-time object detection. Furthermore, this metric is significant because it determines the model's speed and efficiency, which can have significant implications for real-world applications.

The third metric is weight/MB, which measures the model's size in terms of its weight (i.e., the size of the model parameters) per megabyte of storage. In this regard, the Improved-YOLOv5 outperforms the YOLOv5n, with a weight of 1.76 MB versus 3.72 MB. This means that the Improved-YOLOv5 is more compact and takes up less storage space, which is useful in situations where storage space is limited. This metric is also significant because it determines how much storage is required to store the model, which has significant implications for real-world applications. Overall, our analysis of the models' complexity indicates that the Improved-YOLOv5 is a more efficient and less complex model for bus detection systems than YOLOv5n. It has fewer parameters, uses less memory, and can perform more operations in less time. It is also more compact, requiring less storage space. These metrics are significant because they determine how well the model performs in real-world applications as well as the resources needed to run it. As a result, the Improved-YOLOv5 is the more practical solution for bus detection systems because it is simpler, more efficient, and more compact.

3.5. Ablation Experiment

An ablation experiment involves systematically removing or modifying individual components or parts of a model in order to assess their contribution to the overall performance. To accomplish this, we used the same dataset, hyperparameters, and training/testing procedures for all models, with the exception of the specific components being evaluated, throughout the study. This enabled us to attribute any differences in performance between the modified model and the original model to the specific modifications under consideration.

The ablation experiments are presented in Table 3. The results of the ablation study showed that the changes made to the YOLOv5 model had a significant impact on the accuracy and mAP@0.5 performance metrics. When compared to the original YOLOv5, the YOLOv5-Slim model had a slightly lower precision but a higher mAP@0.5. The addition of the ghost module to the YOLOv5-Slim-Ghost model improved the mAP@0.5 score, while keeping the precision score relatively high. This demonstrates that incorporating a ghost module can be an effective way to reduce the computational cost while maintaining the accuracy and mAP@0.5. The YOLOv5-Slim-Ghost-SimSPPF (Improved-YOLOv5) model, which combined the previous modifications with a Simplified Spatial Pyramid Fusion module, outperformed the previous models in terms of the precision and mAP@0.5, while maintaining a similar recall score. This suggests that the SimSPPF module can significantly improve the model's performance, particularly in terms of the accuracy and mAP@0.5.

The ablation study demonstrated a clear tradeoff between the number of parameters and the computational cost, as measured by GFLOPs. The YOLOv5 model had the most parameters (1,763,224) and GFLOPs (4.1), respectively. The YOLOv5-Slim-Ghost model, on the other hand, had the fewest parameters and GFLOPs, with 716,636 and 2.1, respectively. This suggests that combining the slim and ghost modules can result in a more efficient model with a significantly lower computational cost, while still maintaining relatively good performance. However, the addition of the SimSPPF module in the Improved-YOLOv5 model increased the number of parameters and GFLOPs over the YOLOv5-Slim-Ghost model. The Improved-YOLOv5 model had 717,020 parameters and 2.1 GFLOPs, which was comparable to the YOLOv5-Slim-Ghost model. This implies that the addition of the SimSPPF module can improve the model performance without significantly increasing the computational cost. Overall, the ablation study results indicated that the changes made to the YOLOv5 model led to more efficient and effective bus detection models. The slim and ghost modules, which reduced the parameters, significantly reduced the computational cost while maintaining relatively good performance. The SimSPPF module improved the model's performance, particularly in terms of the precision, without significantly increasing the computational cost.

Table 3. Ablation experiments.

Model	YOLOv5	Slim	Ghost	SimSPPF	Precision	Recall	mAP@0.5	Parameters	GFLOPs
1	✓	-	-	-	0.921	0.899	0.923	1,763,224	4.1
2	✓	✓	-	-	0.917	0.901	0.933	1,313,792	3.8
3	✓	-	✓	-	0.92	0.902	0.923	940,628	2.3
4	✓	-	✓(Bone)	-	0.931	0.883	0.935	1,286,412	2.9
5	✓	-	✓(Neck)	-	0.922	0.906	0.925	1,409,024	3.4
6	✓	-	✓	✓	0.93	0.89	0.935	941,012	2.3
7	✓	✓	✓	-	0.91	0.896	0.927	716,636	2.1
8 [1]	✓	✓	✓	✓	0.939	0.895	0.933	717,020	2.1

[1] In this study, the YOLOv5-Slim-Ghost-SimSPPF is referred to as the Improved-YOLOv5.

3.6. Comparative Experiments

The comparison of various models for the bus detection system using the YOLOv5 architecture provides valuable insights into the performance of these models. The evaluation criteria for these models included the inference time, the mAP@0.5, the number of parameters, and the GFLOPs. The Improved-YOLOv5 stood out from the other models, as it achieved the best tradeoff between these evaluation criteria. The inference time for the Improved-YOLOv5 was the lowest among all models, which means that it generated predictions quickly, making it well-suited for real-time applications. The mAP@0.5 score for the Improved-YOLOv5 was also impressive, indicating that it was highly accurate in detecting objects in an image. One of the most notable features of the Improved-YOLOv5 was its lightweight architecture. With only 717,020 parameters, the Improved-YOLOv5 required significantly less storage space than other models, making it more efficient for deployment in resource-limited environments. Moreover, with only 2.1 GFLOPs, the Improved-YOLOv5 required less computational power than the other models, reducing energy consumption and processing time. It is worth noting that YOLOv8n also achieved a high mAP@0.5 score, but it required more parameters and computational power than the Improved-YOLOv5. Similarly, the YOLOv5x achieved a high mAP score but at the cost of requiring significantly more parameters and computational power. Therefore, the Improved-YOLOv5 is the optimal choice for bus detection systems, as it achieves a better balance between accuracy, efficiency, and resource requirements. The comparison of the various models for the bus detection system using the YOLOv5 architecture highlights the superiority of the Improved-YOLOv5. Its low inference time, high accuracy, and lightweight architecture make it the ideal choice for real-time bus detection and tracking systems. The Improved-YOLOv5 represents a significant improvement over the previous models, offering the best performance and efficiency for this application. The performance comparison with various models can be found in Table 4 and Figure 11.

Table 4. Performance comparison of different models.

Model	Inference/ms	mAP@0.5	Parameters	GFLOPs
YOLOv3	2.3	0.925	61,508,200	154.6
YOLOv3-tiny	2.8	0.888	8,671,312	12.9
YOLOv5s	5	0.924	7,018,216	15.8
YOLOv5m	7.9	0.923	20,861,016	47.9
YOLOv5l	15	0.896	46,119,048	107.7
YOLOv5x	15	0.924	86,186,872	203
YOLOv8n	3.1	0.932	3,006,233	8.1
Improved-YOLOv5	2.1	0.933	717,020	2.1

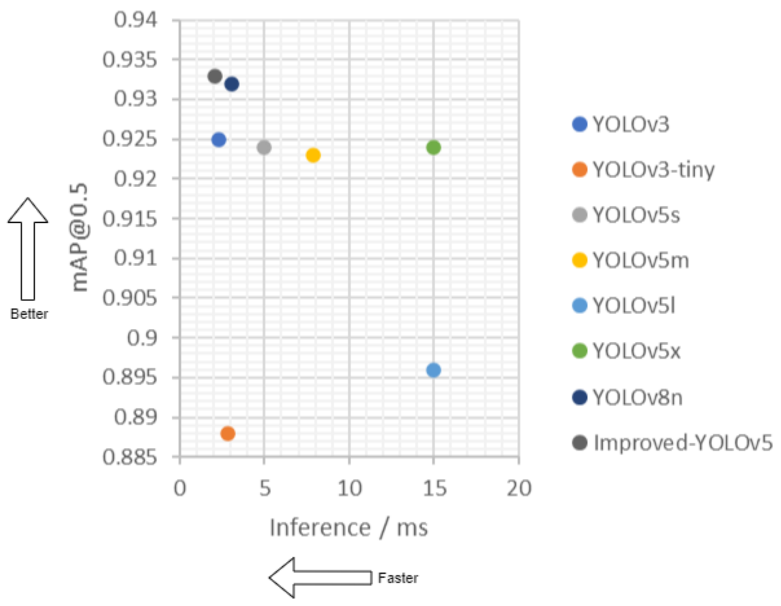


Figure 11. Performance comparison of different models in graphical form.

4. Conclusions

We proposed a lightweight and high-accuracy bus detection model based on a YOLOv5 architecture in this study to address the significant challenges that individuals with visual impairment face while waiting for buses. The Improved-YOLOv5, our proposed model, integrates the GhostConv and C3Ghost Modules into the YOLOv5 network, implements a slim scale detection model, and replaces the SPPF in the backbone with SimSPPF for increased computational efficiency and accurate object detection capabilities. By reducing the number of parameters and FLOPs, our proposed model outperformed the original YOLOv5 model in terms of the precision, recall, and mAP@0.5. The Improved-YOLOv5 model outperformed other object detection models in terms of the accuracy and inference time, as demonstrated by our experimental results. Furthermore, because of its smaller size, lower memory usage, and faster inference time capabilities, the proposed model was more efficient. This makes it a more viable option for mobile devices and real-time applications, especially in resource-constrained scenarios.

The proposed bus detection model has the potential to assist individuals with visual impairment in identifying buses, thereby increasing their independence and quality of life. It can also be integrated into bus detection systems, improving public transportation services, particularly in urban areas. Our proposed model has significant potential benefits for assisting individuals with visual impairment and improving public transportation services, and we believe that our research will inspire further development and research in this area. However, several challenges remain to be addressed, such as improving the model's generalizability across different datasets and environments. More research is needed to investigate the effectiveness of our proposed model under various lighting conditions and environmental factors. Nonetheless, our findings show that object detection models based on the Improved-YOLOv5 have the potential to address real-world problems and contribute to the development of more efficient and accurate bus detection systems, paving the way for future research and development in this field.

Author Contributions: Conceptualization, R.A. and C.W.; methodology, R.A. and C.W.; software, R.A.; validation, R.A. and C.W.; formal analysis, R.A.; investigation, R.A., S.E. and C.W.; resources, R.A. and S.E.; data curation, R.A.; writing—original draft preparation, R.A.; writing—review and editing, R.A. and C.W.; visualization, R.A.; supervision, C.W. All authors have read and agreed to the published version of the manuscript.

Funding: R.A. is supported by a scholarship from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT) scholarship.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are stored in a password-protected PC in the Kyushu Institute of Technology.

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

References

1. World Health Organization (WHO). Blindness and Vision Impairment. 2022. Available online: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment (accessed on 22 February 2023).
2. Crudden, A.; McDonnall, M.; Hierholzer, A. Transportation: An electronic survey of persons who are blind or visually impaired. *J. Vis. Impair. Blind.* **2015**, *109*, 445–456. [CrossRef]
3. Azenkot, S.; Prasain, S.; Borning, A.; Fortuna, E.; Ladner, R.E.; Wobbrock, J.O. Enhancing independence and safety for blind and deaf-blind public transit riders. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Vancouver, BC, Canada, 7–12 May 2011; pp. 3247–3256.
4. El Alamy, L.; Lhaddad, S.; Maalal, S.; Taybi, Y.; Salih-Alj, Y. Bus identification system for visually impaired person. In Proceedings of the 2012 Sixth International Conference on Next Generation Mobile Applications, Services and Technologies, Paris, France, 12–14 September 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 13–17.
5. Oudah, A. RFID-based automatic bus ticketing: Features and trends. *IOP Conf. Ser. Mater. Sci. Eng.* **2016**, *114*, 012146. [CrossRef]
6. Al Kalbani, J.; Suwailam, R.B.; Al Yafai, A.; Al Abri, D.; Awadalla, M. Bus detection system for blind people using RFID. In Proceedings of the 2015 IEEE 8th GCC Conference & Exhibition, Muscat, Oman, 1–4 February 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–6.
7. Abdullah, S.; Noor, N.M.; Ghazali, M.Z. Mobility recognition system for the visually impaired. In Proceedings of the 2014 IEEE 2nd International Symposium on Telecommunication Technologies (ISTT), Langkawi, Malaysia, 24–26 November 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 362–367.
8. Raj, J.T.; Sankar, J. IoT based smart school bus monitoring and notification system. In Proceedings of the 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Dhaka, Bangladesh, 21–23 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 89–92.
9. Yelamarthi, K.; Haas, D.; Nielsen, D.; Mothersell, S. RFID and GPS integrated navigation system for the visually impaired. In Proceedings of the 2010 53rd IEEE International Midwest Symposium on Circuits and Systems, Seattle, WA, USA, 1–4 August 2010; pp. 1149–1152. [CrossRef]
10. Shah, S.; Singh, B. RFID based school bus tracking and security system. In Proceedings of the 2016 International Conference on Communication and Signal Processing (ICCSP), Melmaruvathur, India, 6–8 April 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1481–1485.
11. Wang, Y.; Zheng, Y. Modeling RFID signal reflection for contact-free activity recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2018**, *2*, 1–22. [CrossRef]
12. Liang, X.; Zhang, Y.; Wang, G.; Xu, S. A Deep Learning Model for Transportation Mode Detection Based on Smartphone Sensing Data. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 5223–5235. [CrossRef]
13. Espinosa, J.E.; Velastin, S.A.; Branch, J.W. Vehicle detection using alex net and faster R-CNN deep learning models: A comparative study. In Proceedings of the Advances in Visual Informatics: 5th International Visual Informatics Conference, IVIC 2017, Bangi, Malaysia, 28–30 November 2017; Proceedings 5; Springer: Berlin/Heidelberg, Germany, 2017; pp. 3–15.
14. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv* **2013**. [CrossRef]
15. Girshick, R. Fast R-CNN. *arXiv* **2015**. [CrossRef]

16. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv* **2015**. [CrossRef]
17. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef]
18. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. *arXiv* **2015**. [CrossRef]
19. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; Springer International Publishing: Berlin/Heidelberg, Germany, 2016; pp. 21–37. [CrossRef]
20. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. *arXiv* **2017**. [CrossRef]
21. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. *arXiv* **2019**. [CrossRef]
22. Dong, X.; Yan, S.; Duan, C. A Lightweight Vehicles Detection Network Model Based on YOLOv5. *Eng. Appl. Artif. Intell.* **2022**, *113*, 104914. [CrossRef]
23. Nikkath Bushra, S.; Shobana, G.; Uma Maheswari, K.; Subramanian, N. Smart Video Survillance Based Weapon Identification Using Yolov5. In Proceedings of the 2022 International Conference on Electronic Systems and Intelligent Computing (ICESIC), Chennai, India, 22–23 April 2022; pp. 351–357. [CrossRef]
24. Zhaoxin, G.; Han, L.; Zhijiang, Z.; Libo, P. Design a Robot System for Tomato Picking Based on YOLO v5. In Proceedings of the 16th IFAC Symposium on Large Scale Complex Systems: Theory and Applications LSS 2022, Xi'an, China, 22–24 April 2022; IFAC-PapersOnLine; Volume 55, pp. 166–171. [CrossRef]
25. Mahaur, B.; Mishra, K. Small-object detection based on YOLOv5 in autonomous driving systems. *Pattern Recognit. Lett.* **2023**, *168*, 115–122. [CrossRef]
26. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. GhostNet: More Features from Cheap Operations. *arXiv* **2019**. [CrossRef]
27. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *arXiv* **2019**. [CrossRef]
28. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *arXiv* **2016**. [CrossRef]
29. Wang, W.; Xie, E.; Song, X.; Zang, Y.; Wang, W.; Lu, T.; Yu, G.; Shen, C. Efficient and Accurate Arbitrary-Shaped Text Detection with Pixel Aggregation Network. *arXiv* **2019**. [CrossRef]
30. 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**. [CrossRef]
31. Hosang, J.; Benenson, R.; Schiele, B. Learning non-maximum suppression. *arXiv* **2017**. [CrossRef]
32. Li, C.; Li, L.; Geng, Y.; Jiang, H.; Cheng, M.; Zhang, B.; Ke, Z.; Xu, X.; Chu, X. YOLOv6 v3.0: A Full-Scale Reloading. *arXiv* **2023**. [CrossRef]
33. Zhou, Q.; Liu, H.; Qiu, Y.; Zheng, W. Object Detection for Construction Waste Based on an Improved YOLOv5 Model. *Sustainability* **2023**, *15*, 681. [CrossRef]