electronics

MDPI

Article

A Non-Intrusive Automated Testing System for Internet of Vehicles App Based on Deep Learning

Yanan Zhang [1], Zhen Guo [2] and Tao Sun [1,*]

[1] School of Mechanical Engineering, Tianjin University, No. 135 Yaguan Road, Haihe Education Park, Jinnan District, Tianjin 300354, China
[2] China Automotive Technology & Research Center Co., Ltd., Dongli District, Tianjin 300300, China; guozhen@catarc.ac.cn
* Correspondence: stao@tju.edu.cn

Abstract: In the non-intrusive automated testing system for Internet of Vehicles (IoV) applications, automatic recognition of text and icons on vehicle central control screens is of paramount importance. However, the detection and recognition of content on vehicle central control screens are inherently complex. Additionally, during non-intrusive vehicle central control screen image testing, there is a deficiency of suitable datasets and detection methods. This deficiency renders information within vehicle application images difficult to be accurately extracted by the detection network. To address this problem, this study first constructs a dataset tailored for text detection and recognition on vehicle screens. This dataset encompasses a variety of vehicle central control images, enabling the generic text detection and recognition network to more effectively identify and interpret text within vehicle screens. Subsequently, this research proposes an enhanced Fully Convolutional Networks for Text Detection (FOTS) method for vehicle central control screen text detection and recognition. This method elevates the semantic expression capabilities of features by sharing vehicle central control screen text detection and recognition features. Furthermore, it improves multi-scale feature processing capabilities through the utilization of a feature transformation module. Validation through visual and quantitative experiments demonstrates that the proposed method can effectively accomplish text detection and recognition tasks on vehicle screens. This achievement bears significant implications for the field of automated testing in IoV applications.

Keywords: non-intrusive testing; deep learning; automated testing

Citation: Zhang, Y.; Guo, Z.; Sun, T. A Non-Intrusive Automated Testing System for Internet of Vehicles App Based on Deep Learning. *Electronics* **2023**, *12*, 2873. https://doi.org/10.3390/electronics12132873

Academic Editor: Dimitra I. Kaklamani

Received: 5 May 2023
Revised: 16 June 2023
Accepted: 25 June 2023
Published: 29 June 2023

1. Introduction

With the relentless advancement of 5G and artificial intelligence technologies, the importance of the Internet of Vehicles (IoV) apps in the realm of smart connected vehicles is increasingly emphasized [1–3]. These IoV apps form an interactive conduit between users and their vehicles via mobile devices, bestowing a multitude of control features and informational services. For instance, through remote control functionalities, users can initiate or shut down their vehicle, lock or unlock doors, and even modify certain settings such as seat adjustments and air conditioning, under certain conditions [4,5]. Companies such as Tesla and Volkswagen equip their latest vehicles with these IoV apps extensively, thereby enhancing the convenience of user-vehicle interactions [6].

To ensure that the vehicle accurately reflects the user's commands during this interaction process, thereby validating the correctness and reliability of the entire IoV app interaction process, the application of automated testing technology becomes essential [7]. This technology autonomously confirms the responses and behaviors of the software being tested. For example, if a user adjusts the internal temperature of the air conditioning in the vehicle via their mobile device, the automated testing software monitoring the vehicle's central control display screen should ascertain whether the temperature displayed on the

screen has been adjusted accurately in accordance with the user's command [8]. In this process, the automated testing software is required to capture screen images, recognize text and numerals on the screen, and juxtapose the results with the expected values preset in the database to determine the success of the recognition process. This form of automated testing falls under the umbrella of GUI testing [9,10].

Automated GUI testing technologies for vehicle center control screens primarily encompass two methodologies: intrusive testing and non-intrusive testing [11]. Intrusive testing involves direct contact with or modification of hardware and software, such as retrieving display information from the vehicle's center control screen through wired connections and the use of Android Studio software. However, this approach could potentially disrupt the system's normal operation or even lead to equipment damage and data loss. Presently, open-source testing tools and platforms, including Appium [12], Selendroid [13], and Robotium [14], are available for IoV app interconnectivity functionality testing. These tools automate application testing, enhancing efficiency and reliability. Robotium [14], for example, is an Android platform-based automation testing tool that assists testers in conducting GUI automation testing for Android applications. This tool can operate on real devices or simulators and supports multiple Android versions.

On the other hand, non-intrusive testing refers to the testing conducted without directly modifying or touching the central control hardware and software [15]. It does not rely on hardware interfaces and employs cameras to capture images of the vehicle's center control screen for evaluation. RoScript [16] employs visual testing scripts to articulate GUI operations on touchscreen applications, subsequently utilizing physical robots to drive the automation of test execution. NiCro [17] incorporates the most advanced GUI widget detectors to identify widgets from GUI images, subsequently analyzing a composite set of information to match widgets across different devices. During the actual development of the IoV App, particularly at stages close to delivery, car manufacturers always block vehicle center control screen hardware module interfaces to ensure system stability, resulting in the exclusive use of non-intrusive methods for GUI testing. Figure 1 presents the non-intrusive GUI testing process for IoV apps.

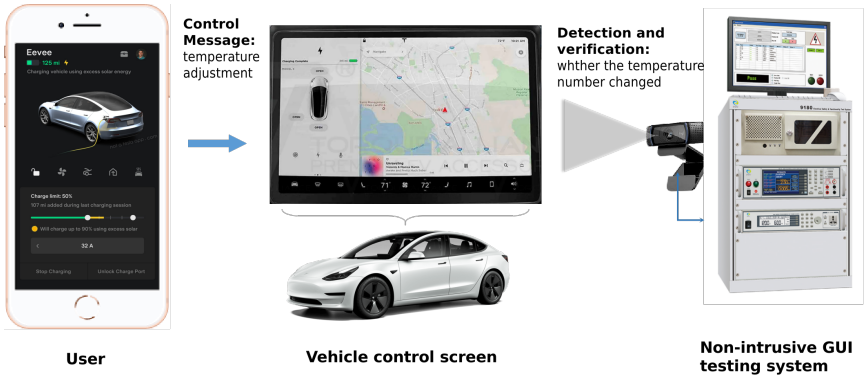


Figure 1. The scheme of non-intrusive GUI testing process for IoV apps.

Taking air conditioning temperature adjustment as an example, the process begins with the user issuing control information containing temperature adjustments or music playback to the vehicle via their mobile device. Subsequently, upon receipt of this message, the vehicle's infotainment center screen adjusts the air conditioning temperature. Finally, the non-intrusive GUI testing system utilizes a camera to examine the vehicle's infotainment center screen, determining whether the air conditioning temperature has been adjusted to the preset number as per the user's command.

While the non-intrusive GUI testing method offers robust adaptability, the quality of the images captured via the camera is significantly lower than screen captures obtained through intrusive methods [18]. To achieve high-quality detection and recognition of text content on the vehicle's central control screen during this process, several challenges must

be addressed. Traditional text recognition methods discern textual meaning within images by analyzing intrinsic characteristics [19], such as structure [20], patterns [21], and unique shapes [22,23]. These approaches lack universality, necessitate manual feature selection to extract textual regions, and require adjustments to multiple parameters for different images, exhibiting inferior robustness [24–27].

Currently, most neural network-based techniques divide this functionality into two components: text detection and text recognition [28,29]. Text detection accurately locates textual areas on the screen, typically employing bounding boxes for annotation, while excluding interfering factors such as distortion and noise [30]. Text detection technology is closely associated with generic object detection technology, and with the progression of deep learning techniques, utilizing convolutional neural networks (CNNs) and recurrent neural networks (RNNs) for in-car central control display text detection has become increasingly prevalent [31]. Neural network technology learns textual region features autonomously through training on copious annotated data [32], thereby accurately detecting and extracting textual information. However, to enhance detection performance and stability, optimization and adjustments are still required with respect to specific vehicle models, screen sizes, and lighting conditions [33,34].

Following text detection, text recognition must be conducted. Traditional text recognition methods, such as k-nearest neighbors, suffer from low accuracy and high computational complexity, rendering them unsuitable for systems with latency requirements. Neural networks can still be applied to the realm of text recognition, but necessitate adjustments to network architecture. Numerous backbone networks for object recognition [35,36], including VggNet [37,38], ResNet [39,40], and DenseNet [41,42], are also highly applicable to text recognition, elevating the recognition accuracy of individual characters. Dataset training is an indispensable component of neural network training, but in the course of researching vehicular screen text recognition, we discovered an almost complete absence of mature datasets within this field. Consequently, to improve vehicular screen text recognition accuracy and practicality, researchers must develop more advanced, robust text detection and recognition techniques suited to diverse scenarios. This may involve designing novel neural network structures and collecting and annotating a wealth of vehicular screen text datasets to cater to the requirements of different vehicle models and contexts [43,44]. Text detection and recognition for IoV vehicular screens constitute a challenging task. This section will leverage deep learning techniques to achieve efficient and accurate text detection and recognition, providing more intelligent and precise technical support for non-intrusive automated testing systems targeting in-car IoV apps [45–47].

It is important to note that, to the best of our knowledge, no previous work has focused on non-intrusive testing for in-vehicle infotainment screens of IoV applications. Our research represents the pioneering efforts in this field, setting a precedent for future studies and practical applications. The principal contributions of this study are as follows:

1. We initially present the VSTDR-2023 dataset, tailored for the detection and recognition of text on vehicular screens. This dataset encompasses a multitude of in-car control images with varying characteristics, thereby enabling text detection and recognition networks to more effectively discern text within vehicular screens.
2. We introduce an innovative vehicular control screen text detection and recognition approach based on an enhanced FOTS framework, achieving end-to-end single-stage text detection and recognition. This method bolsters feature semantic representation capacity by sharing text detection and recognition stage features, effectively reducing computational consumption while preserving result accuracy.
3. Utilizing a feature transformation module in the enhanced FOTS-based vehicular control screen text detection and recognition approach, we resolve the feature reuse issue in text detection and recognition, augmenting multi-scale feature detection capabilities and ensuring text detection efficacy.

Ultimately, we validate the proposed technique through comparative experiments and analyze the results. Both visual and quantitative analyses demonstrate the proficiency of our method in fulfilling the text detection and recognition tasks within vehicular screens.

The following paper is arranged as follows. The related works are reviewed in Section 2. Section 3 introduces the methods containing dataset constructing principle and enhanced FOTS framework. In Section 4, we verify the proposed methods using detailed experiments. Finally, these conclusions are provided in Section 5.

2. Related Works

Determining whether the control information of the vehicle system works properly is the key function of the vehicle network app automated test system. Furthermore, the key of the test system is to detect the content of the vehicle screen. Currently, manual detection and identification methods are still predominantly used in practical applications. However, these methods are costly and require the expertise of specialists. Moreover, the current update pace of in-vehicle systems is rapid, and the functionality modules are becoming increasingly complex. Examples include navigation systems, multimedia entertainment systems, and dash cams. As a result, it becomes challenging to match the corresponding testing manpower in a timely manner. To address these issues, this article proposes an automated and efficient non-intrusive detection system for connected car applications. Next, we will introduce the related work on non-intrusive GUI testing systems.

2.1. Non-Intrusive GUI Testing System

The GUI is an essential feature of current in-vehicle systems, and testing the GUI is also the most effective method to ensure the quality of the entire connected car app. Automated GUI testing methods and tools greatly assist in reducing testing time and costs.

During the early stages of automated testing tool development, there was significant research conducted in the field of mobile phone testing. This was due to the large number of phones that needed to be tested before releasing applications. Relying solely on manual testing was clearly not feasible. Therefore, Dhanapal et al. proposed a remote automated testing system that eliminates the need for any human involvement in the process [15]. The system only requires the involvement of the system administrator to control the testing scripts, the test manager, and the image processing module. The image processing module includes key functionalities such as Optical Character Recognition (OCR) and text box detection. With such methods, the testing efficiency of mobile terminals can be significantly improved. However, the testing scripts and test manager modules in this approach are considered intrusive automated testing methods. Although technologies such as Appium and Sikuli have been widely applied, they face challenges when it comes to closed or less common systems. Therefore, it is necessary to explore another method—non-intrusive testing methods—to further expand the application scope of testing automation technologies.

In 2016, Yu et al. proposed an image-based GUI automation testing method that utilizes image recognition technology to identify elements in GUI images and simulates input signals from the System Under Test (SUT) using input devices [48]. Subsequently, Ju et al. designed a non-intrusive script-driven robot testing system specifically for automating the testing of touchscreen applications [16]. This method employs visual test scripts to express GUI operations on the touchscreen application and utilizes a physical robot to drive the automated test execution. These two methods mentioned belong to non-intrusive testing methods, and their application results further demonstrate the significant potential of non-intrusive testing.

In recent years, there has been significant development in non-intrusive testing research, particularly due to the rapid advancements in computer vision capabilities. The main reason for this progress is that element detection techniques for GUI can directly benefit from mature methods in the field of computer vision (CV). In particular, Chen et al. have designed a novel GUI element detection method that employs a top-down

coarse-to-fine strategy and combines it with state-of-the-art text detection deep learning models [18].

Overall, at the current stage, non-intrusive research specifically targeting connected car apps is still in its early stages, and there is limited related research work available. Furthermore, the existing non-intrusive GUI testing systems are primarily designed for mobile devices such as smartphones and may not be directly applicable to in-vehicle systems. In our research on non-intrusive GUI detection techniques, we have found that text detection technology plays a crucial role in determining the system's detection results and efficiency. However, in most works focusing on screen text recognition, there has been a lack of in-depth research on this module, and the techniques used are somewhat outdated. Meanwhile, in the field of natural scenes, text detection and recognition technology have made significant progress, achieving high accuracy and efficiency. Therefore, it is natural to consider the migration of text detection and recognition techniques from natural scenes to non-intrusive testing systems for connected car apps. Next, we will conduct in-depth research on the relevant technologies for text detection and recognition.

2.2. Text Detection and Recognition Technology

The research background of text detection can be traced back to the early stages of computer vision [49]. With the development of digital image processing and computer vision technology, people began to realize that automatic detection and recognition of text in images were beneficial for many applications, such as document digitization, image retrieval, autonomous driving, and intelligent security. In the field of computer vision, text detection and recognition are considered critical research areas. Early text detection algorithms were primarily based on traditional computer vision techniques, such as edge detection [50], morphological processing [51], and segmentation. The main limitation of these algorithms is their low robustness to factors such as illumination, noise, and complex backgrounds.

Traditional text detection and recognition techniques include edge-based methods, which typically use edge detection algorithms such as the Canny operator to connect edges and form candidate text regions. This approach is simple and fast, but it may yield poor results for images with unclear edges or noise. Another method is based on image segmentation, where clustering or segmentation algorithms are employed to divide the image into different regions. Heuristic rules are then applied to merge regions that potentially contain text into candidate text regions. This method performs better in detecting text in complex backgrounds, but it requires manual parameter adjustment.

In recent years, with the development of deep learning technology, particularly the widespread application of convolutional neural networks (CNNs), text detection accuracy and robustness have significantly improved [52]. Deep learning technology enables text detection to be used in a wider range of application scenarios, such as natural scene images and videos. However, it also introduces higher computational complexity and greater data demands. Neural network-based text detection and recognition algorithms employ neural network technology to automatically detect and recognize text. This method consists of two parts: text detection, which uses neural networks to segment images and identify text regions, and text recognition, which inputs detected text regions into another neural network for text recognition.

At the beginning of text detection and recognition development, much research focused on transferring generic object detection and recognition neural networks to this field. Commonly used neural network frameworks include CRNN, an end-to-end text recognition algorithm based on recurrent neural networks, and convolutional neural networks. It can directly recognize text in images without the need for text detection. It uses convolutional neural networks to extract features from input images and then employs recurrent neural networks for sequence modeling and text recognition. Faster R-CNN is a recently proposed method for localizing target regions. It introduces a new module called the "Region Proposal Network" to the base of R-CNN, effectively improving text detection

accuracy. YOLOv1-v4 is a series of methods that use deep convolutional neural networks for target position detection and, due to their simpler weights, are often used in systems that require real-time processing. It divides the entire image into several grids and uses a convolutional neural network to predict whether each grid contains text, providing the text's location and category.

Despite the significant advancements in text detection and recognition algorithms, challenges remain. For instance, detecting and recognizing text in low-resolution images, under varying lighting conditions, or in images with complex backgrounds still poses difficulties. Additionally, recognizing highly distorted or deformed text, or text with varying fonts, sizes, and orientations, can be challenging for existing methods.

To address these challenges, researchers continue to explore novel techniques and strategies. For example, incorporating attention mechanisms into neural networks can help focus on relevant areas of an image and improve text recognition performance. Similarly, using unsupervised or semi-supervised learning techniques can help reduce the need for large labeled datasets, making the algorithms more practical for real-world applications. Furthermore, integrating contextual information or a higher-level understanding of the scene can assist in recognizing text in more challenging environments [53].

Based on the research findings on non-intrusive GUI testing systems and text detection and recognition technologies mentioned above, we believe there are two main challenges in the non-intrusive testing of automotive central control screens. The details are as follows:

- Lack of mature central control screen dataset. In the application of deep learning or machine learning, a large and high-quality dataset is crucial. For automotive central control screens, there is a wide variety of vehicle models, brands, and interface designs, resulting in highly diverse data. Additionally, the inclusion of various languages, fonts, colors, graphics, dynamic elements, etc., further complicates the data. Currently, there is a lack of comprehensive and high-quality datasets that encompass these factors, making it challenging to train effective models.
- Lack of deep learning GUI text extraction methods suitable for complex automotive central control screens. Non-intrusive testing often requires parsing the GUI of central control screens to identify various elements such as buttons, text, images, etc. However, current deep learning techniques have certain limitations in this regard. The interface designs of automotive central control screens are typically complex and variable, potentially containing multiple languages, various fonts, sizes, colored text, and even dynamic elements and animations. Existing deep learning techniques may struggle to accurately recognize and extract these elements, presenting a major challenge in the field currently.

These two challenges are the main issues faced in the non-intrusive testing of automotive central control screens. Resolving these problems requires in-depth research and development, as well as a larger amount of data and advanced deep learning techniques.

3. Methods

In this section, the discussion begins with an elucidation of the principles and methodologies utilized in the fabrication of test datasets pertinent to in-vehicle infotainment screens. Subsequently, we introduce the proposed enhanced FOTS methodology specifically adapted for the detection of GUI text within the context of in-vehicle infotainment screens.

3.1. Dataset

An exceptional text recognition and detection network invariably hinges on the backing of a scientifically and rationally constructed dataset. Presently, the sphere of text detection and recognition on vehicle screens lacks a suitable, publicly accessible dataset. A few existing datasets, such as the KAIST Scene Text Database furnished by the Korea Advanced Institute of Science and Technology, do include images pertaining to screen text recognition. However, these predominantly feature text from mobile screens in Korean and English, thus limiting their applicability to the research at hand. Additional datasets,

such as Total-Text, contribute a minuscule quantity of screen text detection data, falling short of the requisite volume for training purposes. Numerous contemporary studies default to the use of natural scene datasets for training. Despite the resultant models being applicable to vehicle screen text recognition scenarios, the constricted scope of their training process creates a bottleneck effect on recognition accuracy. This evidences the urgent and crucial need for the development of a dataset specifically tailored for research into vehicle screen text detection and recognition. In response to this need, we propose the Vehicle Screen Text Detection and Recognition 2023 (VSTDR-2023) dataset. Armed with the VSTDR-2023 dataset, conventional text detection and recognition networks will have an enhanced capacity to detect and recognize text on vehicle screens.

3.1.1. Image Acquisition and Selection

The VSTDR-2023 dataset incorporates an array of images corresponding to disparate categories of vehicular central control. The primary sources of these initial images stem from internet downloads and photographs captured by our laboratory team of extant in-car control screens. During the selection process of the original data, the primary considerations were as follows:

- Diverse car interiors. Variations in car interiors create differing backgrounds for the vehicle's central control system, thereby presenting a significant challenge for neural network training. Consequently, it is imperative to select original data images that capture a broad spectrum of car interiors. Typically, car interior designs echo the overall vehicle style, necessitating consideration of different manufacturers and vehicle types.
- Distinct screen materials: The types of screens employed in car central control systems present considerable diversity. For instance, TFT-LCD screens utilize thin-film transistor technology, ensuring high resolution and color performance. Conversely, OLED screens deploy organic light-emitting diode technology, delivering superior brightness, while AMOLED screens, an evolution of OLED screens, feature enhanced brightness and reduced power consumption, typically found in high-end vehicle dashboards or infotainment screens.
- Various screen sizes: Infotainment screens of 6–8 inches are suitable for compact cars, providing smaller size but superior clarity and response speed. Screens ranging from 9–10 inches are more commonplace, suitable for mid-sized vehicles, while 11–13 inch screens are generally reserved for high-end vehicles, boasting larger sizes, multi-touch support, and high-definition video playback capabilities. Figure 2 illustrates the screen in various sizes. As the vehicle manufacturing industry advances, screens larger than 9 inches have gained popularity within vehicle central control systems. Therefore, the dataset should prioritize the collection of such data during the image acquisition process.

(a) (b)

Figure 2. In-car control screens with various sizes. (**a**) 6–8 and (**b**) 11–13 inch screens.

- Various screen shapes: Square screens, typically designed with a 4:3 or 16:9 aspect ratio, are commonplace, displaying an array of information and user interfaces. Rectangular screens are lengthier, often featuring a 21:9 aspect ratio, while trapezoidal screens innovate with differing angles at the top and bottom, adapting more effectively to

the shape and position of the central control panel. Circular screens are uniquely employed in some high-end vehicle designs, providing drivers with a distinctive visual experience. As square and rectangular screens prevail within the market and circular and trapezoidal screens are typically found in high-end vehicles with a smaller market share, data collection should prioritize square and rectangular screens. Figure 3 illustrates the screen in various shapes.

(a) (b)

Figure 3. In-car control screens with various shapes: (**a**) Horizontal and; (**b**) Vertical screens.

- Different scenarios: The content displayed on screens varies significantly across different scenarios, primarily in terms of text positioning, font size, and background content, all of which can significantly impact recognition results. Several interfaces should generally be included: media interfaces (music, radio, etc.), phone interfaces (calls, dialing screens), basic settings interfaces (infotainment settings, device connections), and basic information interfaces (fuel consumption, oil life, etc.).

Considering these influential factors, the VSTDR-2023 dataset, as proposed in this paper, has amassed 1200 images of vehicle central control screens. The display content on these screens encompasses Chinese text, English alphabets, and numerical characters. Notably, attributes such as color, font, size, aspect ratio, stroke width, and quantity of characters diverge significantly across various types and scenarios of vehicle screens. Additionally, during the image selection process, we took into account external environmental factors, such as diverse angles, backgrounds, brightness levels, distances, and image diversity. This approach ensures a broad spectrum of variations in background, lighting, angle, and distance across the entire dataset. To conclude, the VSTDR-2023 dataset, as compiled in this study, is well-suited for generic vehicle screen text recognition, effectively encapsulating the majority of features prevalent in contemporary vehicle screens.

3.1.2. Data Region and Character Content Annotation

This paper primarily concentrates on the detection and recognition of text within vehicle screens. Throughout the dataset creation process, two aspects are predominantly considered:

- Text region annotation. Text region annotation for vehicle central control screen identification tasks. This entails marking all areas within the vehicle central control screen images that contain text with rectangular boxes and subsequently documenting them. The annotators must ensure that the text regions are comprehensively and accurately encapsulated, while vigilantly preventing the misclassification of non-text regions as text-bearing areas.
- character information Annotation. Annotation of specific character information in vehicle screens. This comprises the creation of annotation information for the images within the dataset. Existing Optical Character Recognition (OCR) tools can be deployed to expedite this process, although manual proofreading is requisite for non-standard, blurred, or distorted texts.

In a move to streamline the data annotation process, this paper employs a universal data annotation format. This approach amalgamates detection and recognition results

in a JSON format within a single file, thereby facilitating the training of diverse types of neural networks.

3.1.3. Dataset Splitting

In terms of splitting the VSTDR-2023 dataset, this study delineates it into training, validation, and testing subsets, adhering to the ratios of 80%, 10%, and 10%, respectively. The paper employs a hybrid approach of random and stratified splitting methods for the VSTDR-2023 dataset.

Initially, stratification is executed on central control images from diverse scenarios such as vehicle status, entertainment features, and vehicle settings. Despite these scenes being recurrent, enhancing recognition capabilities across different scenarios is paramount during network training. Thus, this paper stratifies images from various scenarios within the dataset. Subsequently, following the principle of random splitting, data from each stratum is selected, ensuring images from different scenes are equitably divided into distinct subsets, thereby reinforcing the robustness of the trained neural network. Ultimately, the dataset is partitioned into 720 training images, 240 testing images, and 240 validation images.

3.1.4. Data Augmentation

In the domain of deep learning, data augmentation is a technique that generates novel training data through the application of random transformations to the original dataset. This technique is commonly used to address challenges such as insufficient training data or overfitting, thereby enhancing the model's generalizability and robustness. By augmenting the data, the size of the training dataset is increased, thereby augmenting the volume of training samples and reducing the risk of overfitting. Moreover, data augmentation assists the model in learning greater invariance and resilience. For instance, in image classification tasks, operations such as rotation, cropping, and scaling imbue the model with increased robustness in classifying images of varying angles and sizes. It is crucial to select and amalgamate data augmentation techniques based on the specific requirements of the task and the characteristics of the data, integrating random combinations and applications during the training process. Furthermore, to prevent data drift issues arising from excessive augmentation, the scope of augmentation in terms of magnitude and diversity should be judiciously regulated.

In the context of text recognition and detection on in-car control screens, we utilize various data augmentation techniques. Random cropping involves choosing a random area within the original image for cropping, subsequently scaling the cropped region to a fixed size, thereby simulating diverse angles and proportions of vehicular screens. Random adjustments to parameters such as brightness, contrast, and saturation are performed on images, thereby training the model to recognize text under various lighting conditions. Random noise, such as Gaussian or salt-and-pepper noise, is superimposed on images, thereby simulating the conditions of vehicular screens amidst vibrations or noisy environments. These data augmentation strategies effectively strengthen the model's training performance and generalization capabilities.

3.2. Enhanced FOTS

Given the research context, a single-stage FOTS approach is suitable for automotive screen GUI detection and recognition scenarios, as it offers accurate detection rates and reduced computational complexity. Consequently, this paper introduces an enhanced FOTS-based method for the detection and identification of text within vehicle display GUIs. This approach simultaneously addresses text detection and recognition technologies, introducing an end-to-end, single-stage text detection and identification algorithm. Notably, feature extraction convolutions in most neural networks are computationally intensive. This method effectively lessens the computational burden during the detection and recognition stages. However, efficiently sharing features between text detection and recognition has consistently posed challenges. Therefore, this paper incorporates the RoIRotate module,

which introduces text detection features into the text recognition framework, enabling shared feature extraction network components for both text detection and recognition tasks. The scheme of the enhanced FOTS is depicted in Figure 4.

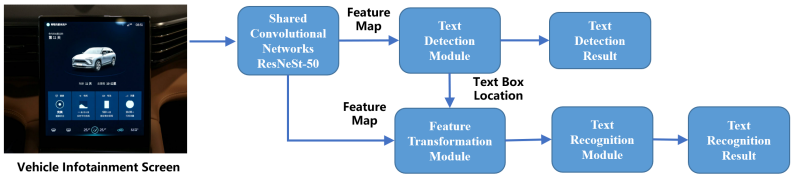


Figure 4. The scheme of enhanced FOTS.

3.2.1. Shared Feature Extraction Layer

As illustrated in Figure 4, the raw data initially undergoes feature extraction via shared convolutional layers upon entering the network. Drawing inspiration from the EAST network model, this paper employs a Fully Convolutional Network (FCN) for comprehensive feature extraction. Additionally, a ResNet-50-based FCN is utilized as the backbone. ResNet-50, a deep convolutional neural network, is built on the principles of residual learning, addressing the vanishing gradient problem by creating shortcuts between inputs and outputs to train deeper networks. The network primarily comprises residual modules, each containing two convolutional layers and a shortcut connection. This shortcut connection facilitates rapid information transfer to output modules, enabling efficient residual information transmission. Moreover, ResNet introduces the Bottleneck structure, designed to decrease parameter count and computational complexity while preserving network depth. Bottleneck residual modules consist of 1×1 and 3×3 convolutions, as well as 1×1 convolutions for channel number reduction and expansion operations.

Simultaneously, to extract character features of varying scales, this method incorporates the concept of the Feature Pyramid Network (FPN), linking low-dimensional and high-dimensional feature maps. Traditional Convolutional Neural Networks (CNNs) typically necessitate multiple convolutions and pooling operations on input images for tasks such as object detection or semantic segmentation, generating a series of progressively smaller feature maps. While shallower feature maps provide more detailed information, deeper feature maps offer higher-level semantic information. The core concept of FPN involves enhancing the semantic expressiveness of feature maps through cross-layer connections while maintaining feature map detail. Given the significant variation in text size in automotive systems, this method employs two FPN structure components for bottom-up feature extraction and top-down feature fusion.

Although ResNet attains satisfactory detection and recognition accuracy, its simplistic structure can result in less-than-optimal precision. Therefore, this paper proposes the use of a split attention module, enabling cross-group feature recognition. Termed ResNeSt, this network retains the primary architecture of ResNet, permitting direct application to downstream tasks without incurring additional computational costs. The ResNeSt model outperforms other networks of comparable complexity.

3.2.2. Text Detection

In this stage, the shared feature extraction layer previously described performs the crucial function of generating features for each pixel. During this process, the feature map size undergoes an expansion from $1/32$ to $1/4$. Post feature extraction, the text detection branch discerns whether each pixel signifies a character and forecasts the coordinates of the corresponding character bounding box.

Figure 5 illustrates the composition of the text detection branch, which comprises multiple convolutional neural network layers. Initially, a convolutional layer computes the probability of each pixel functioning as a positive sample. Subsequently, for every pixel that qualifies as a positive sample, quadruple channel predictions are initiated to estimate the distances to the top, bottom, left, and right boundaries of the predicted bounding box.

Ultimately, a specific channel is tasked with predicting the orientation of the associated bounding box.

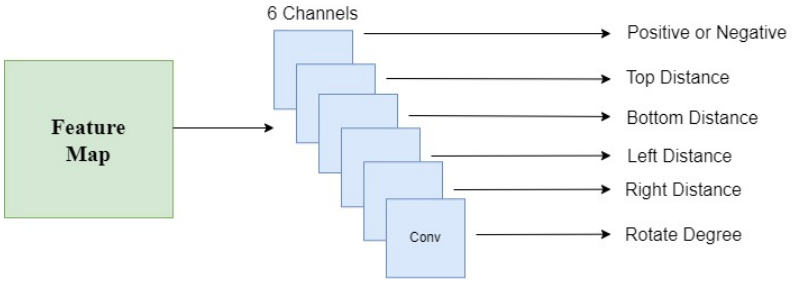


Figure 5. Input and output of text detection.

The loss function of the text detection branch constitutes two distinct components:

The other component is the regression term, which consists of two parts: IoU loss and rotation angle loss. These two regression losses ensure the final predicted bounding box is robust in terms of shape, scale, and orientation. The specific expressions are as follows :

$$L_{reg} = \frac{1}{|\Omega|} \sum_{x \in \Omega} \mathrm{IoU}(\mathbf{R}_x, \mathbf{R}_x^*) + \lambda_\theta (1 - \cos(\theta_x, \theta_x^*)) \tag{1}$$

In the formula, the first term $\mathrm{IoU}(\mathbf{R}_x, \mathbf{R}_x)$ represents the IoU loss value between the predicted bounding box and the ground truth bounding box. The second term is the rotation angle loss, where θ_x is the predicted angle, and θ_x is the true angle. Here, λ_θ is a hyperparameter for adjusting the weight between boundary prediction and rotation angle prediction, typically set to 10.

Thus, the loss calculation formula for the text detection branch is organized as follows:

$$L_{\text{detect}} = L_{\text{cls}} + \lambda_{\text{reg}} L_{\text{reg}} \tag{2}$$

In the equation, λ_{reg} is also a hyperparameter, mainly used to adjust the balance between the two loss values, typically set to 1.

3.2.3. Feature Transformation Module

To utilize the results of the shared feature extraction layer for text recognition, it is vital to adjust the orientation of the feature map. Consequently, this method employs the RoIRotate module to execute a directional transformation on the feature region, generating a feature map that is parallel to the coordinate axes. The diagram of the feature transformation module is depicted in Figure 6

Figure 6. Diagram of feature transformation module.

Specifically, the RoIRotate module rotates the input feature map by an angle θ and carries out the corresponding cropping and padding operations to ensure that the output feature map's size matches that of the input feature map. The rotation and scaling opera-

tions necessitate pixel-level interpolation, which is achieved using bilinear interpolation, thereby allowing for more precise pixel values. The rotation angle θ is predicted by the text detection network, enabling RoIRotate to adapt to text in any orientation. Throughout the transformation process, the height and aspect ratio of the text feature remain consistent, while the length of the text can be adjusted, making it more amenable to the vehicle screen text recognition scenario targeted in this study.

The specific computations within the RoIRotate module are as follows:

- Calculate affine transformation coefficients. Calculate the affine transformation coefficients using the predicted or true text box coordinates:

$$t_x = l * \cos\theta - t * \sin\theta - x \tag{3}$$

$$t_y = t * \cos\theta + l * \sin\theta - y \tag{4}$$

$$s = \frac{h_t}{t + b} \tag{5}$$

$$w_t = s * (l + r) \tag{6}$$

$$\mathbf{M} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$= s \begin{bmatrix} \cos\theta & -\sin\theta & t_x\cos\theta - t_y\sin\theta \\ \sin\theta & \cos\theta & t_x\sin\theta + t_y\cos\theta \\ 0 & 0 & \frac{1}{s} \end{bmatrix} \tag{8}$$

  In this context, $\mathbf{M}$ refers to the affine transformation matrix, h_t and w_t represent the height and width of the feature map after affine transformation, respectively. (x, y) denotes the coordinates of a point in the shared feature map, (t, b, l, r) represent the distance of the pixel to the top, bottom, left, and right sides of the text box, and θ indicates directionality. These parameters can be either actual label data or predicted data attributes.
- Perform affine transformations. Affine transformations are performed on the shared feature maps of each region to attain standardized horizontal text region feature maps. This can be achieved through the following equation:

$$\begin{pmatrix} x_i^s \\ y_i^s \\ 1 \end{pmatrix} = \mathbf{M}^{-1} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \tag{9}$$

  Affine transformation parameters are calculated using the ground truth coordinates, and these transformations are then applied to the shared feature maps of each candidate region. This results in standardized horizontal feature maps of the text regions. During the network training process, the actual text regions are utilized to train the network. In the testing process, thresholding and Non-Maximum Suppression (NMS) are employed to filter the predicted text regions. Once the data has been processed through the RoIRotate module, the transformed feature maps are fed into the text recognition branch.

3.2.4. Text Recognition

Considering the lengthiness of the label sequence in the text region, this study employs an approach that expands the features input to the Long Short-Term Memory (LSTM)

network along the width axis to half of the original image's size (1/4 at input). This strategy is advantageous in distinguishing features within compact text regions, particularly narrow character features. The text recognition branch is structured with a VGG-type convolutional network, a bidirectional LSTM [16,42], a fully connected network, and lastly, a Connectionist Temporal Classification (CTC) decoder. Table 1 provides a detailed representation of the text recognition structure.

Table 1. Text recognition neural network.

Layer	Convolution Kernel Size [Scale, Step-Size]	Output Channels
Convolutional layers	[3, 1]	64
Convolutional layers	[3, 1]	64
Height dimension pooling	[(2, 1), (2, 1)]	64
Convolutional layers	[3, 1]	128
Convolutional layers	[3, 1]	128
height dimension pooling	[(2, 1), (2, 1)]	128
Convolutional layers	[3, 1]	256
Convolutional layers	[3, 1]	256
height dimension pooling	[(2, 1), (2, 1)]	256
Bidirectional LSTM	[3, 1]	256

Next, we will elaborate on the processes in the text recognition algorithm and the design of the associated Loss function.

The processes for the text recognition algorithm are delineated as follows:

1. Initially, transformed feature maps are input, passing through multiple linear convolution and pooling layers. This process reduces the height dimension of the spatial feature map while extracting features of higher dimensionality. Subsequent to this step, we derive the feature vectors to be recognized, forming them into a feature sequence. For simplification, we present the convolution parameters of the VGG network in Table 1.
2. Subsequently, the high-dimensional feature sequence is encoded within an RNN network, facilitating the prediction of the label distribution for each feature vector within the sequence. Here, we employ a bidirectional LSTM and designate 256 output channels for each direction. This step yields a probability list of predicted results.
3. Following this, the hidden states are computed at each time point in both directions, and the results are summed. These outcomes are then input into a fully connected network, resulting in the distribution of each state within the character classification. Through this step, we obtain the probability distribution of all characters, represented as a vector with a length equivalent to the number of character categories. To circumvent overfitting during training, a dropout layer is inserted prior to full connection.
4. Lastly, the probability vector is input into the Connectionist Temporal Classification (CTC) layer. This stage transforms the predictions rendered by each feature vector into a label sequence, addressing the redundancy issue that can occur within the RNN network.

Then, we will elaborate on the design of the loss function for the text recognition branch. With regard to the loss function of the text recognition branch, it is pertinent to note that the predicted conditional probabilities x_t of distinct characters for each hidden state vary. Let $\mathbf{y}$ represent the actual label sequence. Hence, the conditional probability of $\mathbf{y}$ is the sum of the probabilities of all paths π.

$$p(\mathbf{y}^*|\mathbf{x}) = \sum_{\pi \in B^{-1}(\mathbf{y}^*)} p(\pi|\mathbf{x}) \tag{10}$$

B represents a many-to-one mapping, which corresponds to a label set with partial blank and duplicate labels to y^*. The training goal of the text recognition branch network is to maximize the sum of the log-likelihoods in the above expression.

$$L_{\text{recog}} = -\frac{1}{N}\sum_{n=1}^{N} \log p(\mathbf{y}_n^*|\mathbf{x}) \tag{11}$$

N is the number of text regions in the input image.

Now, we can obtain the overall loss function for the entire text detection and recognition network, as follows:

$$L = L_{\text{detect}} + \lambda_{\text{recog}} L_{\text{recog}} \tag{12}$$

λ_{recog} is a hyperparameter for balancing the two losses. In experiments, λ_{recog} is usually set to 1.

4. Experimental Results

4.1. Experimental Setup

The initial phase of training typically necessitates the selection of an appropriate learning rate, supplemented by learning rate scheduling strategies that progressively diminish the learning rate, circumventing overfitting in the later training stages. In this study, the initial learning rate is established at 1×10^{-3} , with a decay to 1×10^{-4} after 5000 iterations.

All experimental procedures delineated in this section are conducted on a computer equipped with an NVIDIA RTX 2060 GPU, 32GB RAM, and an Intel Core i7-8700k CPU. The neural network methodology proposed herein is operationalized using the PyTorch framework, an instrument that permits developers to freely define, train, and deploy neural network models. The inherent diversity of datasets and complexity of tasks in text detection and recognition necessitate flexible tools for model building and debugging, a requirement satisfactorily met by PyTorch. As PyTorch's core language is Python, the algorithmic development is also conducted in Python, with Ubuntu 18 as the development system and PyCharm as the development platform.

During the neural network training process, batch size configuration demands consideration of various factors, including hardware limitations, model size, and dataset size. Generally, the batch size should balance available hardware resources and model size. To ascertain an optimal value, tests were performed with batch sizes of 32, 64, 128, and 256. The experimental dataset employed is the VSTDR-2023, as proposed in Section 2.

As evidenced in Figure 7, an increase in batch size decelerates the network's convergence speed and amplifies the fluctuations in loss value. Consequently, the batch size value cannot be increased indefinitely. Pursuant to these findings, this study adopts a batch size of 32 for subsequent training experiments.

4.2. Evaluation Metrics

To verify the accuracy of text detection and recognition, relevant indicators are defined for quantitative evaluation. Commonly used indicators include recall, which represents the proportion of correctly detected text boxes to all text boxes. Detection precision is the proportion of the correct text boxes detected to the total number of text boxes. Character Error Rate (CER) is the proportion of different characters between the recognition result and the true label. The F1 score is a measure of the performance of a classification model, a comprehensive index of precision and recall, representing the accuracy and coverage of the model in predicting positive classes (e.g., correct recognition in text recognition).

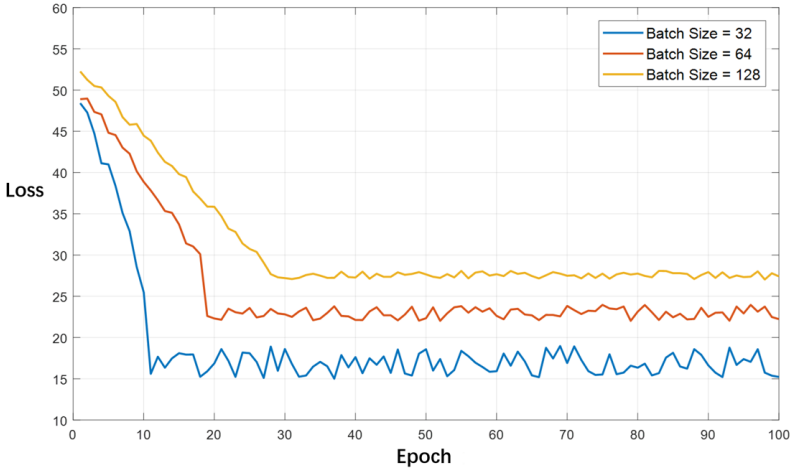


Figure 7. Comparison of the results of different Batch Size experiments.

4.3. Experimental Results and Analysis

This section will verify the effectiveness of each component of the proposed method on the VSTDR-2023 dataset. Table 2 shows the comparison results.

Table 2. The effect of text detection module and feature transformation module on VSTDR-2023.

Text Detection Module	Feature Transformation Module	Precision (%)	Recall (%)	F Score (%)
		67.23	63.21	65.16
✓		79.52	75.58	77.52
	✓	70.12	66.43	68.26
✓	✓	95.77	91.27	93.75

An ablation study is conducted on the VSTDR-2023 dataset to quantify the relative contributions of the text detection module and the feature transformation module. These ablation experiments are conducted within the same framework on identical hardware, with results depicted in Table 2.

Initial observations indicate that in the absence of both the text detection module and the feature transformation module, the recognition accuracy is significantly low, at 67.23%, accompanied by a correspondingly low F-score of 65.16%. Upon the integration of only the text detection module or the feature transformation module, we observe substantial improvements in precision, alongside increases in both recall rate and F-score. It is particularly noteworthy that the text detection module contributes more substantially to these improvements. Upon the concurrent inclusion of the text detection module and the feature transformation module, there is an overall enhancement across all performance metrics.

Table 3 shows the quantitative results on the VSTDR-2023 dataset. The method proposed in this section achieves 95.77%, 91.23%, and 93.75% in precision, recall, and F-measure, respectively. Compared to the method proposed by Shi et al., the F-measure of our method is improved by about 5%.

Table 3. Text detection and recognition results.

Method	Precision (%)	Recall (%)	F Score (%)
Shi et al. [54]	91.13	83.37	87.73
Panhwar et al. [55]	92.14	85.76	88.13
Xiao et al. [56]	93.83	91.35	92.33
Chen et al. [57]	93.64	89.95	92.31
Our method	95.77	91.23	93.75

The end-to-end text recognition algorithm proposed in this section notably outperforms other algorithms in terms of comprehensive performance and processing speed. The superior results obtained through our proposed algorithm can be attributed to two primary factors: First, the shared feature extraction layer that efficiently extracts text features from images plays a crucial role. Second, the implementation of the RoIRotate module within the feature transformation module significantly aids in applying the features to text recognition. Consequently, the method proposed herein can achieve a considerable gain in performance.

The gradient descent algorithm is used here to optimize the network model, so the change in loss value during the training process is not smooth. As can be seen from Figures 8 and 9, with the increase in iterations, the network loss value gradually decreases and eventually stabilizes at a constant value. At the same time, the classification accuracy also increases with the increase in iterations. After about 2000 iterations, the classification accuracy finally stabilizes at around 0.95.

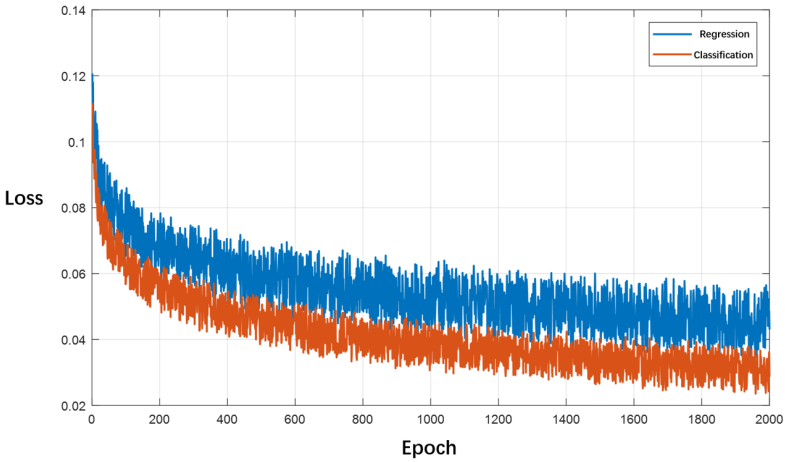


Figure 8. The regression and classification loss value curve in VSTDR-2023 dataset.

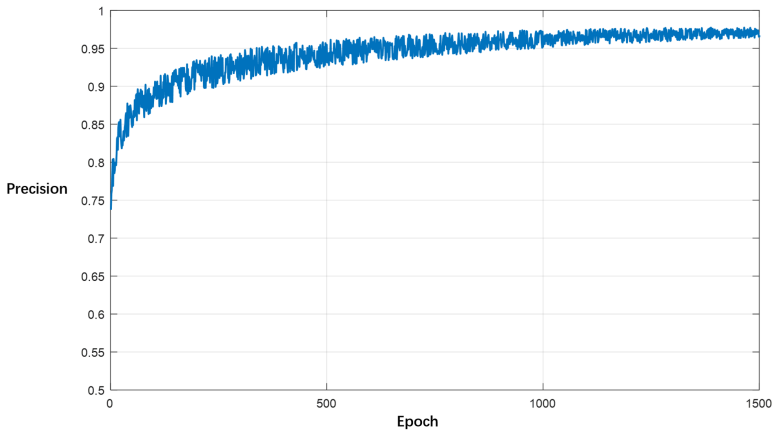
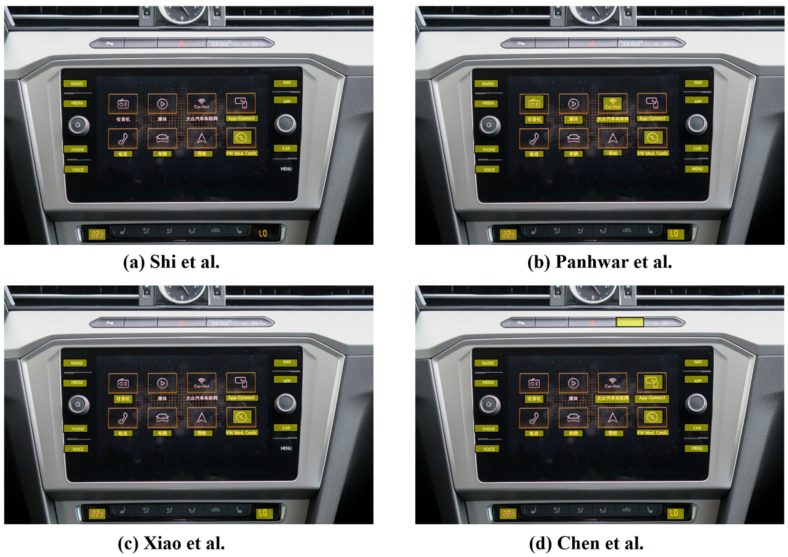


Figure 9. The precision value curve in VSTDR-2023 dataset.

Experimental results on the VSTDR-2023 dataset, illustrated by visual outcomes of state-of-the-art methods and our proposed approach in Figures 10 and 11, provide essential insights. In qualitative comparisons, the methods proposed by Shi et al., Xiao et al., and Panhwar et al. struggle with text detection in areas characterized by multiple bright spots and an uneven black-and-white distribution at the screen center, thereby leading to missed detections and compromised text detection accuracy. The techniques put forward by Panhwar et al. and Chen et al. tend to inaccurately detect small text and individual characters. For instance, as seen in Figure 10b, certain text, such as the Wi-Fi signal, is misidentified as incorrect text regions.



(a) Shi et al. (b) Panhwar et al.

(c) Xiao et al. (d) Chen et al.

Figure 10. The state-of-the-art methods text detection and recognition result of in-car control screen [54–57].

Figure 11. The enhanced FOTS text detection and recognition result of in-car control screen.

However, the text detection and recognition scheme proposed in this study effectively identifies and interprets text on in-vehicle screens, addressing a significant majority of the challenges encountered when applying classic text detection neural networks to this scenario. In summary, the novel text detection scheme presented in this paper demonstrates promising results for detecting text on vehicle-mounted screens, resolving most problems associated with directly applying classical text detection neural networks to such situations.

5. Conclusions

This research primarily addresses the lack of adequate training datasets for text detection and recognition within non-intrusive vehicle infotainment testing systems. We designed and created a dataset specifically tailored for in-vehicle screen text detection and recognition. Following this, we proposed an enhanced FOTS-based method for detecting and recognizing graphical user interfaces (GUIs) on in-vehicle screens. This method employs shared feature modules and feature transformation modules to precisely automate the recognition and operation of icons and text within vehicle infotainment systems. When compared to the generic text detection and recognition method introduced by Shi et al., the

detection and recognition accuracy of the proposed algorithm has shown an approximate improvement of 3.6%.

In the future, as vehicle networking technology advances and application scenarios continue to expand, in-vehicle GUI testing will become increasingly complex. To enhance the functionality testing of infotainment screens, attention mechanisms can be considered to ensure recognition capabilities for icons and multilingual text in complex backgrounds. Attention mechanisms can aid GUI testing image recognition models in better focusing on crucial regions, thereby improving their accuracy and performance. For instance, attention mechanisms of various scales can be designed for targets of different sizes and proportions, enabling the model to adaptively recognize targets of diverse dimensions. For multi-channel images in GUI testing, utilizing channel attention mechanisms can allow the model to better distinguish features between different channels, consequently enhancing the model's accuracy and performance.

Author Contributions: Conceptualization, Y.Z.; Methodology, Y.Z.; Validation, Y.Z.; Resources, Y.Z.; Writing—original draft, Y.Z.; Writing—review & editing, Z.G. and T.S.; Supervision, T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is unavailable due to company privacy.

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

References

1. Yang, F.; Wang, S.; Li, J.; Liu, Z.; Sun, Q. An overview of internet of vehicles. *China Commun.* **2014**, *11*, 1–15. [CrossRef]
2. Elmoiz Alatabani, L.; Sayed Ali, E.; Mokhtar, R.A.; Saeed, R.A.; Alhumyani, H.; Kamrul Hasan, M. Deep and Reinforcement Learning Technologies on Internet of Vehicle (IoV) Applications: Current Issues and Future Trends. *J. Adv. Transp.* **2022**, *2022*, 1947886. [CrossRef]
3. Hamid, U.Z.A.; Zamzuri, H.; Limbu, D.K. Internet of vehicle (IoV) applications in expediting the implementation of smart highway of autonomous vehicle: A survey. In *Performability in Internet of Things*; Springer: Cham, Switzerland, 2019; pp. 137–157.
4. Svangren, M.K.; Skov, M.B.; Kjeldskov, J. The connected car: An empirical study of electric cars as mobile digital devices. In Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services, Vienna, Austria, 4–7 September 2017; pp. 1–12.
5. Schipor, O.A.; Vatavu, R.D.; Vanderdonckt, J. Euphoria: A Scalable, event-driven architecture for designing interactions across heterogeneous devices in smart environments. *Inf. Softw. Technol.* **2019**, *109*, 43–59. [CrossRef]
6. Rohm, A.J.; Gao, T.T.; Sultan, F.; Pagani, M. Brand in the hand: A cross-market investigation of consumer acceptance of mobile marketing. *Bus. Horiz.* **2012**, *55*, 485–493. [CrossRef]
7. Menzel, T.; Bagschik, G.; Maurer, M. Scenarios for development, test and validation of automated vehicles. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 1821–1827.
8. Banerjee, I.; Nguyen, B.; Garousi, V.; Memon, A. Graphical user interface (GUI) testing: Systematic mapping and repository. *Inf. Softw. Technol.* **2013**, *55*, 1679–1694. [CrossRef]
9. Chang, T.H.; Yeh, T.; Miller, R.C. GUI testing using computer vision. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Atlanta, GA, USA, 10–15 April 2010; pp. 1535–1544.
10. Borjesson, E.; Feldt, R. Automated system testing using visual gui testing tools: A comparative study in industry. In Proceedings of the 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation, Montreal, QC, Canada, 17–21 April 2012; pp. 350–359.
11. Herbold, S.; Grabowski, J.; Waack, S.; Bünting, U. Improved bug reporting and reproduction through non-intrusive gui usage monitoring and automated replaying. In Proceedings of the 2011 IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops, Berlin, Germany, 21–25 March 2011; pp. 232–241.
12. Singh, S.; Gadgil, R.; Chudgor, A. Automated testing of mobile applications using scripting technique: A study on appium. *Int. J. Curr. Eng. Technol. (IJCET)* **2014**, *4*, 3627–3630.
13. Li, A.; Li, C. Research on the Automated Testing Framework for Android Applications. In Proceedings of the 12th International Conference on Computer Engineering and Networks, Haikou, China, 4–7 November 2022; pp. 1056–1064.
14. Zadgaonkar, H. *Robotium Automated Testing for Android*; Packt Publishing: Birmingham, UK, 2013.
15. Dhanapal, K.B.; Deepak, K.S.; Sharma, S.; Joglekar, S.P.; Narang, A.; Vashistha, A.; Salunkhe, P.; Rai, H.G.; Somasundara, A.A.; Paul, S. An Innovative System for Remote and Automated Testing of Mobile Phone Applications. In Proceedings of the 2012 Annual SRII Global Conference, San Jose, CA, USA, 24–27 July 2012; pp. 44–54. [CrossRef]

16. Qian, J.; Shang, Z.; Yan, S.; Wang, Y.; Chen, L. RoScript: A Visual Script Driven Truly Non-Intrusive Robotic Testing System for Touch Screen Applications. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE '20), Seoul, Republic of Korea, 27 June–19 July 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 297–308. [CrossRef]
17. Xie, M.; Ye, J.; Xing, Z.; Ma, L. NiCro: Purely Vision-based, Non-intrusive Cross-Device and Cross-Platform GUI Testing. *arXiv* **2023**, arXiv:2305.14611.
18. Chen, J.; Xie, M.; Xing, Z.; Chen, C.; Xu, X.; Zhu, L.; Li, G. *Object Detection for Graphical User Interface: Old Fashioned or Deep Learning or a Combination?* Association for Computing Machinery: New York, NY, USA, 2020. [CrossRef]
19. Zuo, L.Q.; Sun, H.M.; Mao, Q.C.; Qi, R.; Jia, R.S. Natural scene text recognition based on encoder-decoder framework. *IEEE Access* **2019**, *7*, 62616–62623. [CrossRef]
20. Zhang, C.; Xu, Y.; Cheng, Z.; Pu, S.; Niu, Y.; Wu, F.; Zou, F. SPIN: Structure-preserving inner offset network for scene text recognition. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 2–9 February 2021; Volume 35, pp. 3305–3314.
21. Chen, X.; Jin, L.; Zhu, Y.; Luo, C.; Wang, T. Text recognition in the wild: A survey. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–35. [CrossRef]
22. Wang, Y.; Xie, H.; Zha, Z.J.; Xing, M.; Fu, Z.; Zhang, Y. Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11753–11762.
23. Tian, Z.; Shu, M.; Lyu, P.; Li, R.; Zhou, C.; Shen, X.; Jia, J. Learning shape-aware embedding for scene text detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4234–4243.
24. Odeh, A.; Odeh, M.; Odeh, H.; Odeh, N. *Hand-Written Text Recognition Methods: Review Study*; International Information and Engineering Technology Association: Washington, DC, USA, 2022.
25. Hwang, W.Y.; Nguyen, V.G.; Purba, S.W.D. Systematic survey of anything-to-text recognition and constructing its framework in language learning. *Educ. Inf. Technol.* **2022**, *27*, 12273–12299. [CrossRef]
26. Liu, X.; Meng, G.; Pan, C. Scene text detection and recognition with advances in deep learning: A survey. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2019**, *22*, 143–162. [CrossRef]
27. Long, S.; He, X.; Yao, C. Scene text detection and recognition: The deep learning era. *Int. J. Comput. Vis.* **2021**, *129*, 161–184. [CrossRef]
28. Liao, M.; Wan, Z.; Yao, C.; Chen, K.; Bai, X. Real-time scene text detection with differentiable binarization. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11474–11481.
29. Aberdam, A.; Litman, R.; Tsiper, S.; Anschel, O.; Slossberg, R.; Mazor, S.; Manmatha, R.; Perona, P. Sequence-to-sequence contrastive learning for text recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–20 June 2021; pp. 15302–15312.
30. Alsamadony, K.L.; Yildirim, E.U.; Glatz, G.; Waheed, U.B.; Hanafy, S.M. Deep learning driven noise reduction for reduced flux computed tomography. *Sensors* **2021**, *21*, 1921. [CrossRef] [PubMed]
31. Abdallah, A.; Hamada, M.; Nurseitov, D. Attention-based fully gated CNN-BGRU for Russian handwritten text. *J. Imaging* **2020**, *6*, 141. [CrossRef] [PubMed]
32. Kim, H.H.; Jo, J.H.; Teng, Z.; Kang, D.J. Text detection with deep neural network system based on overlapped labels and a hierarchical segmentation of feature maps. *Int. J. Control. Autom. Syst.* **2019**, *17*, 1599–1610. [CrossRef]
33. Hozhabr Pour, H.; Li, F.; Wegmeth, L.; Trense, C.; Doniec, R.; Grzegorzek, M.; Wismüller, R. A machine learning framework for automated accident detection based on multimodal sensors in cars. *Sensors* **2022**, *22*, 3634. [CrossRef]
34. Celaya-Padilla, J.M.; Galván-Tejada, C.E.; Lozano-Aguilar, J.S.A.; Zanella-Calzada, L.A.; Luna-García, H.; Galván-Tejada, J.I.; Gamboa-Rosales, N.K.; Velez Rodriguez, A.; Gamboa-Rosales, H. "Texting & Driving" detection using deep convolutional neural networks. *Appl. Sci.* **2019**, *9*, 2962.
35. Hardalaç, F.; Uysal, F.; Peker, O.; Çiçeklidağ, M.; Tolunay, T.; Tokgöz, N.; Kutbay, U.; Demirciler, B.; Mert, F. Fracture detection in wrist X-ray images using deep learning-based object detection models. *Sensors* **2022**, *22*, 1285. [CrossRef]
36. Zhang, R.; Shao, Z.; Huang, X.; Wang, J.; Li, D. Object detection in UAV images via global density fused convolutional network. *Remote Sens.* **2020**, *12*, 3140. [CrossRef]
37. Shanmugavel, A.B.; Ellappan, V.; Mahendran, A.; Subramanian, M.; Lakshmanan, R.; Mazzara, M. A Novel Ensemble Based Reduced Overfitting Model with Convolutional Neural Network for Traffic Sign Recognition System. *Electronics* **2023**, *12*, 926. [CrossRef]
38. Liu, M.; Li, B.; Zhang, W. Research on Small Acceptance Domain Text Detection Algorithm Based on Attention Mechanism and Hybrid Feature Pyramid. *Electronics* **2022**, *11*, 3559. [CrossRef]
39. Ganesan, J.; Azar, A.T.; Alsenan, S.; Kamal, N.A.; Qureshi, B.; Hassanien, A.E. Deep Learning Reader for Visually Impaired. *Electronics* **2022**, *11*, 3335. [CrossRef]
40. Peng, H.; Yu, J.; Nie, Y. Efficient Neural Network for Text Recognition in Natural Scenes Based on End-to-End Multi-Scale Attention Mechanism. *Electronics* **2023**, *12*, 1395. [CrossRef]

41. Gong, H.; Liu, T.; Luo, T.; Guo, J.; Feng, R.; Li, J.; Ma, X.; Mu, Y.; Hu, T.; Sun, Y.; et al. Based on FCN and DenseNet Framework for the Research of Rice Pest Identification Methods. *Agronomy* **2023**, *13*, 410. [CrossRef]
42. Akhtar, M.J.; Mahum, R.; Butt, F.S.; Amin, R.; El-Sherbeeny, A.M.; Lee, S.M.; Shaikh, S. A Robust Framework for Object Detection in a Traffic Surveillance System. *Electronics* **2022**, *11*, 3425. [CrossRef]
43. Lee, H.J.; Ullah, I.; Wan, W.; Gao, Y.; Fang, Z. Real-time vehicle make and model recognition with the residual SqueezeNet architecture. *Sensors* **2019**, *19*, 982. [CrossRef] [PubMed]
44. Tang, T.; Li, L.; Wu, X.; Chen, R.; Li, H.; Lu, G.; Cheng, L. TSA-SCC: Text semantic-aware screen content coding with ultra low bitrate. *IEEE Trans. Image Process.* **2022**, *31*, 2463–2477. [CrossRef] [PubMed]
45. Nagy, V.; Kovács, G.; Földesi, P.; Kurhan, D.; Sysyn, M.; Szalai, S.; Fischer, S. Testing Road Vehicle User Interfaces Concerning the Driver's Cognitive Load. *Infrastructures* **2023**, *8*, 49. [CrossRef]
46. Gao, Y.; Feng, J.; Liu, F.; Liu, Z. Effects of Organic Vehicle on the Rheological and Screen-Printing Characteristics of Silver Paste for LTCC Thick Film Electrodes. *Materials* **2022**, *15*, 1953. [CrossRef]
47. Xue, H.; Zhang, Q.; Zhang, X. Research on the Applicability of Touchscreens in Manned/Unmanned Aerial Vehicle Cooperative Missions. *Sensors* **2022**, *22*, 8435. [CrossRef]
48. Yu, Z.; Xiao, P.; Wu, Y.; Liu, B.; Wu, L. A Novel Automated GUI Testing Echnology Based on Image Recognition. In Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, Australia, 12–14 December 2016; pp. 144–149. [CrossRef]
49. Bhatt, D.; Patel, C.; Talsania, H.; Patel, J.; Vaghela, R.; Pandya, S.; Modi, K.; Ghayvat, H. CNN variants for computer vision: History, architecture, application, challenges and future scope. *Electronics* **2021**, *10*, 2470. [CrossRef]
50. You, N.; Han, L.; Zhu, D.; Song, W. Research on image denoising in edge detection based on wavelet transform. *Appl. Sci.* **2023**, *13*, 1837. [CrossRef]
51. Tadic, V. Study on Automatic Electric Vehicle Charging Socket Detection Using ZED 2i Depth Sensor. *Electronics* **2023**, *12*, 912. [CrossRef]
52. Alaskar, H.; Hussain, A.; Al-Aseem, N.; Liatsis, P.; Al-Jumeily, D. Application of convolutional neural networks for automated ulcer detection in wireless capsule endoscopy images. *Sensors* **2019**, *19*, 1265. [CrossRef]
53. Xu, C.; Zhu, G.; Shu, J. A combination of lie group machine learning and deep learning for remote sensing scene classification using multi-layer heterogeneous feature extraction and fusion. *Remote Sens.* **2022**, *14*, 1445. [CrossRef]
54. Shi, B.; Bai, X.; Yao, C. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 2298–2304. [CrossRef]
55. Panhwar, M.A.; Memon, K.A.; Abro, A.; Zhongliang, D.; Khuhro, S.A.; Memon, S. Signboard detection and text recognition using artificial neural networks. In Proceedings of the 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 12–14 July 2019; pp. 16–19.
56. Xiao, Y.; Xue, M.; Lu, T.; Wu, Y.; Palaiahnakote, S. A text-context-aware CNN network for multi-oriented and multi-language scene text detection. In Proceedings of the 2019 International Conference on Document Analysis and Recognition (ICDAR), Sydney, NSW, Australia, 20–25 September 2019; pp. 695–700.
57. Chen, X.; Lv, Z.; Zhu, D.; Yu, C. Ticket Text Detection and Recognition Based on Deep Learning. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 3922–3926.