Machine vision-based testing action recognition method for robotic testing of mobile application

Tao Zhang¹, Zhengqi Su¹, Jing Cheng², Feng Xue¹ and Shengyu Liu¹

Abstract
The explosive growth and rapid version iteration of various mobile applications have brought enormous workloads to mobile application testing. Robotic testing methods can efficiently handle repetitive testing tasks, which can compensate for the accuracy of manual testing and improve the efficiency of testing work. Vision-based robotic testing identifies the types of test actions by analyzing expert test videos and generates expert imitation test cases. The mobile application expert imitation testing method uses machine learning algorithms to analyze the behavior of experts imitating test videos, generates test cases with high reliability and reusability, and drives robots to execute test cases. However, the difficulty of estimating multi-dimensional gestures in 2D images leads to complex algorithm steps, including tracking, detection, and recognition of dynamic gestures. Hence, this article focuses on the analysis and recognition of test actions in mobile application robot testing. Combined with the improved YOLOv5 algorithm and the ResNet-152 algorithm, a visual modeling method of mobile application test action based on machine vision is proposed. The precise localization of the hand is accomplished by injecting dynamic anchors, attention mechanism, and the weighted boxes fusion in the YOLOv5 algorithm. The improved algorithm recognition accuracy increased from 82.6% to 94.8%. By introducing the pyramid context awareness mechanism into the ResNet-152 algorithm, the accuracy of test action classification is improved. The accuracy of the test action classification was improved from 72.57% to 76.84%. Experiments show that this method can reduce the probability of multiple detections and missed detection of test actions, and improve the accuracy of test action recognition.

Keywords
Mobile application automation testing, expert imitation testing, action recognition, timing behavior detection

Date received: 10 May 2022; accepted: 3 July 2022

Handling Editor: Yanjiao Chen

Introduction
With the development of mobile information technology, the number and penetration rate of mobile phone users are growing rapidly. Massive mobile applications and iterative updates between application versions make mobile application testing an important part of ensuring the quality of mobile applications. Robotic testing for mobile applications is an emerging approach to mobile application testing. Combined with graphical

¹School of Software, Northwestern Polytechnical University, Xi’an, China
²School of Computer Science and Engineering, Xi’an Technological University, Xi’an, China

Corresponding author:
Jing Cheng, School of Computer Science and Engineering, Xi’an Technological University, Xi’an, Shaanxi 710028, China.
Email: chengjing@xatu.edu.cn

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
user interface (GUI) testing, it drives robots to manipulate GUI widgets to simulate user interaction behavior, which reduces the time spent by testers in processing tasks and improves the efficiency of automated testing.\textsuperscript{1-4}

Robotic testing for mobile applications based on machine vision takes interface interaction as the core. It completes repetitive testing tasks through the interaction between the robot and the GUI of the intelligent terminal device. In the current automated testing methods for mobile applications, three types of test action recognition technologies are commonly used\textsuperscript{5}: wearable devices based on gloves, 3D positions based on key points of the hand, and raw visual data. The first method requires wearing an additional device that affects the operational feel of the test. The second method requires additional extraction of hand key points, which is time-consuming, and the quantity of manual tasks is large when adding new action recognition types. The third method requires only one image capture sensor, such as a camera, to achieve acceptable recognition accuracy.

This article adopts the third test action recognition method. Based on machine vision technology, a top-down mobile application testing action recognition process is built. First, the start time and end time of the test action are accurately located, and the type of test operation is identified. Then, a hierarchical mobile application test action model is built. Finally, test cases with high reliability and reusability are generated. Our approach enables highly repeatable regression and compatibility testing in mobile application testing. The contributions are as follows:

1. A network model for extracting testing action is proposed. Image features are extracted in video sequences using the YOLOv5 algorithm to detect the location of smart terminal devices and testers’ hands.
2. A high-accuracy testing action classification network model is trained and generated by spatio-temporal generalization of the ResNet-152 algorithm and introducing the pyramid context awareness mechanism.

The rest of this article is organized as follows. The “Related works” section summarizes the related work involved in mobile application testing. The “Framework of MAEIT” section introduces the framework of the mobile application expert imitation testing (MAEIT) method. The “Mobile application testing action recognition and classification” section details the testing action analysis and recognition methods. The “Experimental analysis and results” section discusses the experimental setup and results. Finally, the “Conclusion” section presents the conclusions.

Related works

This section presents the current research on automated mobile application testing techniques and action recognition. Krieter and Breiter\textsuperscript{6} used video analysis techniques to extract user interaction action events on Android applications. The work focused on playback on the original device and did not provide self-learning capabilities for scripts on new devices. Zhang et al.\textsuperscript{7} proposed a terminable learning model based on deep learning to identify similarities between GUI and recognizable GUI, avoiding redundant test cases and preventing explosion in the number of test cases by merging redundant isomorphic nodes in the GUI model. Xue et al.\textsuperscript{8} proposed a completely black-box learning-playback testing method that combines robotics and vision technologies to enable cross-device and cross-platform record-playback testing. At this stage, action recognition techniques are mostly used for recognizing human actions, identity authentication and other scenarios.

The field of action recognition applications is very broad.\textsuperscript{9-11} Park and Kim\textsuperscript{12} presented an efficient action recognition of 3D images using a convolutional neural network (CNN). Roig et al.\textsuperscript{13} suggested a solution that combines action, scene, object, and audio information to solve multimodal human action recognition tasks in the video by detecting people and using audio and visual information. Yoo et al.\textsuperscript{14} proposed a hybrid hand gesture system, which combines an inertial measurement unit (IMU)-based motion capture system and a vision-based gesture system to increase real-time performance.

The current research on gesture recognition is more active than in other fields. Gesture recognition can be mainly divided into sensor-based gesture recognition and vision-based gesture recognition. Sensor-based gesture recognition uses 1D raw data extracted from wearable sensors, such as IMU without feature extraction. However, this method has higher requirements on the equipment, and the environment configuration is complicated.\textsuperscript{15} Vision-based gesture recognition is more economical and easier to collect data. It analyzes 2D images, 3D images, or videos obtained from optical sensors. Sinha et al.\textsuperscript{16} proposed a hand gesture recognition using the hidden Markov model and trajectory code word. Zhu et al.\textsuperscript{17} proposed a new method to capture shape information for 3D gestures: segmenting the hand shape from a depth image captured by a Kinect sensor with a cluttered background. The action modeling approach based on vision suffers from poorly
defined action boundaries and a large span of temporal dimensions for different action categories, which is not conducive to effective feature extraction and feature fusion for recognition models.

In summary, in the current article, the robotic testing of mobile application still has shortcomings, such as low robot participation and difficulty in building automated scripts. Action recognition is mainly applied in areas, such as human movement and surveillance tracking. However, research on minute gesture recognition remains insufficient. To address these two issues, a method is proposed based on improvements to the YOLOv5 target recognition algorithm and the ResNet-152 algorithm.

Framework of MAEIT

Figure 1 shows the framework of MAEIT. Realizing a non-intrusive mobile application automatic test method and reducing the intervention of the mobile intelligent terminal equipment system when building automatic test cases are proposed. MAEIT is a software and hardware system platform that integrates hardware devices, such as high-resolution industrial cameras and four-degree-of-freedom robots, as well as software modules, such as mobile application GUI layout, control icons, and expert test interaction feature extraction derived from video analysis. Build test cases by analyzing expert test video features. The test case is converted into robot attitude control instructions according to the image information collected by the industrial camera through the coordinate transformation matrix. Accurate execution of mock tests is done across devices.

The first step is to capture a video of the user performing the mobile application test with a high-resolution industrial camera. The received video is then analyzed, and the start time, end time, and operation type of the test action in the video are identified. Then, the GUI and control icons of the mobile smart device are analyzed before and after the test operation is performed, and a test case that integrates the information of the GUI path and the control icon information is generated. Finally, the device-under-test application GUI state is captured by the camera. According to the generated test cases, the robotic arm is driven in real time to complete the imitation test task.

The specific module description of MAEIT is described as follows:

Analysis of action and recognition of expert video

Seven test action types are defined to simulate test case generation, namely, single click, double click, long press, slide up and down, slide left and right, rotate, and zoom. Action recognition obtains information, such as action type and interaction coordinates in user videos. A mobile application test action analysis and recognition method is proposed based on the fusion of the YOLOv5 algorithm and the ResNet-152 algorithm, which completes the test action classification task for recorded videos. A layered architecture is proposed for mobile application testing behavior analysis and identification tasks. The architecture includes a network model for extracting testing action and network model for classifying testing action.

Recognition of GUI element

Much progress has been made in the research of GUI element recognition based on deep learning. According to whether the element is interactive and the element feedback results, GUI elements are divided into three basic elements (text, pictures, and buttons) and two logical controls (list view and grid view). The YOLOv3 model identifies information, such as the type, coordinates, and size of the GUI elements. This module clarifies the layout of the functional areas of the mobile application by analyzing the information of the GUI elements and further forms a functional block structure of the GUI with clear semantics.

According to the identification result of the type and position of the elements in the interface, the relative positions of the elements are rearranged to establish the nesting relationship of the elements between the mobile
application GUIs. This kind of relationship is described as a tree structure to form a GUI element tree model. Figure 2 shows the construction of the GUI element tree. As a leaf node of the GUI element tree, the GUI contains information about elements. The jump of the GUI is realized by operating the elements in the interface. Tree (a) is a “GUI control tree” model of interface (1). The interactive elements in the interface are the leaf nodes in the GUI control tree. The internal nodes of the GUI control tree respectively correspond to the recognition results of the view elements of the GUI (1).

**Generation of expert imitation test scripts**

According to the identification of GUI element information and the jump relationship between the interfaces, combined with the test operation type of the tester, the combination relationship between the GUI path and the test action is formed, and the test case is finally generated.

The action recognition and GUI control recognition results are combined to construct an expert imitation test script structure. The script contains information, such as interaction coordinates and action types in the user video. It also uses self-learning features, such as “GUI path” and control icons to realize the self-positioning of the script’s control operation objects under the new device.

**Execution of robot imitation test**

MAEIT drives the robot to perform specific test tasks. When MAEIT provides operation instructions to the robot, the coordinates of the operating controls of the test case should be converted into the executable space coordinates of the robot in the real physical world to control the attitude of the robot. The mobile application robot executes executable use cases to realize the test evaluation of the software on different devices.

In summary, mobile application testing action analysis assists in the generation of expert imitation test cases, which is the key to generating expert imitation test scripts. Precisely locating the time boundary and test action type of the user’s human–computer interaction in the expert test video is necessary.

---

**Figure 2.** The process of building a GUI element tree.
Mobile application testing action recognition and classification

A vision-based approach to mobile application testing action recognition is developed on raw video data. The action extraction network is trained to generate the candidate action segments. Then, the test action recognition network is trained to identify the types of test actions in the candidate action segments.

Mobile application testing action definition

When conducting mobile application testing, the tester needs to perform gesture operations on the touch screen of the mobile smart terminal device. The gestures of the mobile application testing action are described from seven dimensions: number of clicks, trigger timing, time limit, moving direction, number of operating fingers, pressing force, and contact area. Variations in these dimensions produce different gesture variants.

Combined with the seven dimensions of the gesture division of mobile application testing actions, the seven types of mobile application testing actions are described in detail from a mathematical point of view, as shown in Figure 3.

1. **Single click.** Press and lift one finger on the hot spot to trigger the corresponding function

   $V(A) = \{P_{0}(x, y), n_{f}\}$

   where $P_{0}$ refers to the pixel coordinates of the control icon and $n_{f}$ refers to the number of fingers performing the operation.

2. **Double click.** Press and lift one finger at the hot spot, and then press and lift again to trigger the corresponding function

   $V(B) = \{P_{0}(x, y), t_{d}, t_{c}, n_{f}, n_{l}\}$

   where $t_{d}$ denotes the duration of the pressed state, $t_{c}$ denotes the action pause time, and $n_{l}$ denotes the action pause time.

3. **Long press.** Press with one finger on the hotspot and keep it pressed for a specific amount of time

   $V(C) = \{P_{0}(x, y), t_{d}, n_{f}\}$

4. **Slide up and down (left and right).** Press with one finger on the hot spot, keep it pressed, and move the finger from top to bottom or bottom to top (from left to right, or from right to left)

   $V(D) = \{P_{0}(x, y), D_{r}, P_{1}(\Delta x, \Delta y), n_{f}\}$

   where $D_{r}$ denotes the sliding direction and $P_{1}$ denotes the pixel coordinate transformation increment.

5. **Rotate.** Press the two fingers on the hot spot, keep the pressed state, and then turn the wrist at a certain angle to rotate the two fingers

   $V(F) = \{P_{0}(x_{0}, y_{0}), P_{1}(x_{1}, y_{1}), O_{0}(x, y), D_{r}, \theta, n_{f}\}$

   where $P_{0}$ and $P_{1}$ denote the initial pixel coordinates of the thumb and index finger, respectively; $O_{0}$ refers to the pixel coordinates of the center of rotation; $D_{r}$ and $\theta$ refer to the direction and angle of rotation, respectively.

6. **Zoom.** Press down on the hot spot with two fingers, keep it pressed down, and then control the thumb and index finger to contract in or out at the same time

   $V(G) = \{P_{0}(x_{0}, y_{0}), P_{1}(x_{1}, y_{1}), D_{r}, P_{1}(\Delta x, \Delta y), n_{f}\}$

   where $D_{r}$ and $P_{1}$ denote the direction and stroke of the zoom, respectively.

Algorithm flow for testing action recognition and classification

As shown in Figure 4, the flow of the algorithm can be roughly summarized into two steps: behavior extraction and behavior classification. The specific process of mobile application testing behavior analysis and identification algorithm is as follows:

Use a high-resolution camera to record professional movements of your mobile app to create expert testing videos.

1. Use a high-resolution camera to record professional movements of your mobile app to create expert test videos.
2. Image features are extracted using the YOLOv5 object detection algorithm. When both the mobile phone and the tester’s hand are detected in a certain frame of image, the position information of the intelligent terminal device and the mobile application tester’s hand bounding box in the image target detection result is analyzed. Then determine the start and end time of the test action, and generate a sequence of behavior proposals.
3. Determine whether to activate the test action classification network according to the detection result of the test action extraction network. When the test action is not detected, it returns to the video sequence to continue the detection operation. Otherwise, the test action classification network is activated, identifying the class of the test action in the candidate segment.

4. The test action recognition network constructed based on ResNet-152 enables it to process video data through the spatio-temporal generalization of the network structure. And the pyramid context awareness mechanism is introduced to increase the receptive field during feature extraction. When testing the action classification network to classify test action candidate segments, we mainly identify the seven actions in the “Mobile application testing action definition” section.

**Network model for extracting testing action**

The network model for extracting testing action distinguishes between gesture classes and no gesture classes by running an object detection algorithm on a sequence of images. We consider using the YOLOv5 object detection algorithm and improve it. The advantage of YOLOv5 is the anchor box mechanism included in the target detection network structure. This mechanism can detect multiple objects present in the image at the same time, and can even deal with inconsistent object proportions or a small amount of occlusion caused by distance.

**Boundaries for start and end actions.** During the frame-by-frame disassembly of the long video, the target category and bounding box position information are recorded by identifying the intelligent terminal equipment and the tester’s hand in the image. The tester performs a touch operation on the GUI of the intelligent terminal device to generate the test action. During this process, the smart terminal device in the video and the tester’s hand inevitably overlap, as shown in Figure 5. The video clips in which the terminal device and the tester’s hands are stably coincident are defined as the test action extraction clips.

**Improved YOLOv5 algorithm.** The YOLO target detection algorithm uses the K-means clustering algorithm in the anchor box generation stage. Hence, the YOLOv5 algorithm inherits the defects of the K-means algorithm in the anchor generation stage: the anchor boxes generated by the network under the same input show
variability. Moreover, YOLOv5 regards the anchor frame as a constant during the training, which ultimately affects the calculation result of the model loss function during the training and increases the difficulty of model convergence. In response to these two problems, an improved method for the YOLOv5 algorithm is proposed.

Improving dynamic anchor boxes. A dynamic anchor box feature selection (DAFS) structure is added to the YOLOv5 model. After generating the predefined anchor box information, the detection network is used as the input of the anchor refinement module (ARM). It is connected to the backbone network of YOLOv5 through bidirectional feature fusion (BFF) to generate a priori box for model training.

Figure 6 shows that the ARM method cleans the negative sample data of the native anchor box. Then, the coordinates and size of the original anchor box are modified according to the real position of the target in the original image, and the prior information of the target detection network is generated. The BFF connects adjacent feature maps of different scales, connects to the backbone network of the YOLOv5 target detection network model, and finely adjusts the feature points on the feature sequence before outputting the prior frame. In subsequent iterations, the update of the feature map continues to serve as the basis for the ARM module to fine-tune the anchor box.

Introducing attention mechanism. Adding a visual attention mechanism to the backbone network of YOLOv5 can improve the extraction efficiency of the model for detecting object features in the input image.

The SENet network structure realizes the processing of the global information of the image by performing the squeeze operation and the excitation operation on the feature sequence extracted from the image. Its specific structure is shown in Figure 7.

The operation of squeeze is shown in equation (7)

$$z_c = F_{sq}(U_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j)$$  \hspace{1cm} (7)$$

where $z_c$ denotes the $c$th element. $H$ and $W$ refer to height and width, respectively. And $u_c(i, j)$ represents the $(i, j)$th element of the $c$th channel of $u$.

The squeeze operation describes the features of the entire image through the global average pooling method. The extraction operations of the global features are as follows

$$s = F_{ex}(z, W) = \sigma(W_2 \delta(W_1 z))$$

$$\delta_{sc} = F_{sc}(u_c, s_c) = s_c u_c$$  \hspace{1cm} (8)$$

When the attention model outputs, the degree of attention to different channels is differentiated utilizing
additional weights: The key channels are given higher weight values, whereas the unconcerned channels have lower weight values. In this article, attention modules of spatial attention and mixed channel attention are added to the lower layer of the YOLOv5 backbone network. Global average pooling layers and global max-pooling layers are used in the channel attention module.

Figure 8 shows that the “CSP × S + A” module is added after the feature map that has undergone the convolution operation. The CSPdarknet53 network structure is the backbone of the YOLOv5 object detection network structure, and is used as the backbone of the mobile application test action proposal network to participate in the training of the model. The “Conv + BN + LeakyReLU” module is a key step in the realization of the channel attention mechanism. It can enhance the acquisition efficiency of the network model’s attention information. Furthermore, the speed and accuracy of the network’s behavior detection are effectively promoted.

Correcting bounding regression box. The deviation of the anchor box coverage position filtered by YOLOv5 is improved by correcting the bounding regression box. Weighted boxes fusion (WBF) is a method of bounding box regression. WBF sets different weights for all predicted bounding boxes according to the difference in bounding box confidence and generates a new predicted bounding box through the weighted fusion method.

When dividing the positive and negative samples of the video training set, the prior frame generated by the video is judged according to the corresponding rules. The first thing to do is to identify when the mobile app testing behavior occurs. If the area delineated by the prior box has the highest overlap ratio with the time period when the real behavior occurs, the prior box is considered as a positive sample. In particular, for each real action, at least one positive sample prior box must be selected corresponding to it. If there is a certain a priori box in the remaining a priori box that overlaps with the time period when the real behavior occurs in the time dimension greater than 0.7, it is counted as a positive sample. For a priori box whose overlap ratio is less than 0.3, it is recorded as a negative sample.

The WBF algorithm not only retains the anchor box information with high confidence but also collects the anchor box information with low confidence. Most of the redundant predicted bounding boxes through WBF are fused and destroyed to generate fewer predicted bounding boxes with high accuracy, as shown in Figure 9.

Network model for classifying testing action

When the network for extracting test action detects a gesture and generates a candidate action clip, the network for classifying testing action is activated and uses the candidate action clip as input to determine the type of test action in the video.

Spatio-temporal generalization of 2D CNN structures. The 2D ResNet is transformed into a 3D ResNet with a
wider feature receptive field and the ability to process video data by adding a processing module for temporal information. In terms of network structure, the 3D ResNet has a high similarity with the 2D ResNet. In addition to the minor adjustments made to the convolutional dimension of each layer, the increase in the temporal dimension requires appropriate improvements to the moving step of the convolution kernel in the first layer of the network structure. Table 1 shows the structure of the network before and after the extension.

The first to fourth layers of the 3D ResNet-152 network are used to extract the video features of mobile application test action candidate clips. The output of the fourth-layer network is used as the feature map for the input of the mobile application test action type recognition network, establishing a mapping between the original video and all points within the feature map. Then, the feature maps pass through a 3D averaging pooling layer and are adjusted to the same scale features sequences as the input of the fifth-layer network, which is then subjected to deeper feature extraction. Finally, the window size of the averaging pooling layer is set to the size of the feature map by global averaging pooling to eliminate the black-box features in the fully connected layer. The above completes the mobile application testing action classification.

**Pyramid context awareness mechanism.** The pyramid context awareness mechanism processes video clips through several temporal dilated convolution modules to explore the contextual information deeply in the video. Dependencies between temporal information and video units in long videos are established to obtain features with great receptive field and multiscale information.

When a deep learning network extracts video features in the convolutional layer, it expands the receptive

---

**Table 1.** Comparison between 3D ResNet-152 network and 2D network structure.

| Layer name | Output size | ResNet-152 | 3D ResNet-152 |
|------------|-------------|------------|---------------|
| Conv1      | 112 × 112   | 7 × 7, 64, stride = 2 | 7 × 7 × 7, 64, stride = 1 × 2 × 2 |
| Conv2_x    | 56 × 56     | 3 × 3, max pool, stride = 2 | 3 × 3 × 3, 3D max pool, stride = 2 |
| Conv3_x    | 28 × 28     | 1 × 1, 64, 3 × 3, 128, 8 | 1 × 1, 64, 3 × 3, 128, 8 |
| Conv4_x    | 14 × 14     | 1 × 1, 256, 3 × 3, 256, 36 | 1 × 1, 256, 3 × 3, 256, 36 |
| Conv5_x    | 7 × 7       | 1 × 1, 512, 3 × 3, 512, 3 | 1 × 1, 512, 3 × 3, 512, 3 |
|            | 1 × 1       | Average pool | 3D average pool |
|            |             | FC-1000 softmax | Softmax |

---

**Figure 9.** The effect of weighted fusion of WBF target box.

---

**Zhang et al.**
field of the feature map directly by increasing the gap between convolution kernels. In the time-series data processing, combined with dilated convolution, the model can obtain multiscale information while keeping the time-series length unchanged. Figure 10 shows the performance of dilated convolution on spatial convolution and temporal convolution.

**Experimental analysis and results**

A comparative experiment on the accuracy of the improved YOLOv5 algorithm is conducted, and the effect of adding a pyramid context awareness mechanism on the accuracy of mobile application testing action recognition is verified.

**Dataset preparation**

The experiments use picture datasets and video datasets. The static picture dataset is used to train the YOLOv5 network to recognize smart terminal devices and testers’ hands, whereas the dynamic video dataset is used to train the temporally generalized ResNet-152 network to recognize mobile application testing action. Both datasets are recorded in a laboratory environment with a resolution of 1920 × 1080 for images and videos, and the acquisition device is Vizion Smart Industrial Camera MV-EM500C.

The integrated dataset includes 546 static images and 1848 dynamic videos, of which the dynamic mobile application testing action videos include click, double click, long press, slide up and down, slide left and right, rotate, and zoom operations. To achieve better detection results for the proposed network and classification network of mobile application testing action, 1200 training images and videos are recorded, including seven action categories. The test set part contains 648 video clips with the same total of seven action categories. Table 2 shows the composition of the datasets for each category.

### Table 2. The composition of the datasets.

| Datasets                  | Category                | Number |
|--------------------------|-------------------------|--------|
| Static picture dataset   | Smart terminal devices  | 546    |
|                          | Testers’ hands          | 513    |
| Dynamic video dataset    | Click                   | 315    |
|                          | Double click            | 162    |
|                          | Long press              | 228    |
|                          | Slide up and down       | 436    |
|                          | Slide left and right    | 355    |
|                          | Rotate                  | 164    |
|                          | Zoom                    | 188    |

**Results and discussion**

First, the location of the smart terminal device and the tester’s hand in the video are identified by the improved YOLOv5 algorithm. A geometric measure is used to dichotomize the operation of the GUI, determining whether the testing action is occurring. This step results in the positioning of video action clips and the generation of proposal clips. Then, the 3D ResNet-152 mobile application testing action recognition network with pyramid context awareness mechanism is used to predict the type of testing action.

**Experiment of improved YOLOv5 algorithm.** The YOLOv5 object detection model is configured using TensorFlow framework, and the network is trained. The YOLOv5 + DAFS + attention + WBF improvement algorithm is compared with several YOLOv5 object detection algorithms in a controlled experiment on an expert test video dataset, and then the experimental results are derived and analyzed. Table 3 shows the results of the experiments.

### Table 3. Comparative experimental results of improved YOLOv5 algorithm.

| Method                   | mAP (%) |
|--------------------------|---------|
| YOLOv5                   | 84.6    |
| YOLOv5 + DAFS            | 87.5    |
| YOLOv5 + attention       | 86.3    |
| YOLOv5 + WBF             | 88.4    |
| YOLOv5 + DAFS + attention | 92.8    |

DAFS: dynamic anchor box feature selection; WBF: weighted boxes fusion.
to more than 90%. It shows that the improved YOLOv5 target detection algorithm can also achieve
good performance in real scenes. Compared with the
mainstream target detection algorithm, the improved
algorithm maintains certain advantages in detection
accuracy.

3D ResNet-152 pyramid context awareness mechanism comparison experiment. A 3D ResNet-152 network with pyr-
amid context awareness mechanism constructed by
temporal dilated convolution is compared with a 3D
ResNet-152 network obtained by spatio-temporal gen-
eralization only for mobile application testing action
recognition and classification. The experimental results
are shown in Table 4.

The mobile application action recognition network
adds a pyramid context awareness mechanism module
and a temporal dilated convolution module that uses
dense junctions for connectivity, making the structure
of the deep learning network more complex. The
extracted multiscale video features can be better reused
and fused, enabling the mobile application action rec-
ognition network to achieve higher recall rates for can-
didate action segments. The accuracy of 3D ResNet-
152 in mobile application action recognition is
improved by introducing the pyramid context aware-
ness mechanism.

In summary, the non-end-to-end trained mobile
application testing action localization and recognition
network proposed in this article can accurately locate
testing action in long unprocessed videos. The larger
receptive field in the feature extraction network facil-
itates the extraction of varying lengths action fragment
features, improving the accuracy of testing action loca-
lization and type recognition.

Conclusion

This article proposes a MAEIT method that generates
test cases executable by the mobile test robot by
analyzing the video of the human–computer interaction
and guides the robot to complete the test task. Importantly, methods for analyzing and identifying test
actions are examined. Based on the design logic of object
detection algorithms, a visual learning modeling algorithm
for mobile application testing action based on the fusion
of YOLOv5 and the ResNet models is proposed. It locates
the boundaries of testing action effectively, improves the
accuracy of proposal classification, and reduces the rate of
missing mobile application testing action. It provides a
feasible manner to achieve cross-platform and cross-
operating system testing tasks while providing new ideas
for temporal action detection tasks in other fields.

Some limitations remain in this work. First, in the
actual scene, errors are inevitably caused in the test
action extraction stage due to the inclination of the
object placement angle and other reasons. Second, the
mobile application testing behavior analysis and recog-
nition algorithm model proposed in this article is still
complex. Although it belongs to the same type of tim-
ing behavior detection algorithm, it can realize the
extraction of video information. However, in combina-
tion with the current state of technological develop-
ment, the implementation of the project is still difficult.
In the future, we will continue to improve the detection
efficiency of the method and promote its integration
with the commercial mobile apps testing process.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with
respect to the research, authorship, and/or publication of this
article.

Funding

The author(s) received no financial support for the research,
authorship, and/or publication of this article.

ORCID iD

Tao Zhang  
https://orcid.org/0000-0003-3695-8328

| Algorithm       | 3D ResNet-152 | 3D ResNet-152 + pyramidal context-aware mechanism |
|-----------------|---------------|--------------------------------------------------|
|                 | mAP (%)       | Recall (%)                                       | mAP (%)       | Recall (%)                                       |
| Click           | 72.36         | 69.56                                            | 77.48         | 75.42                                            |
| Double click    | 69.43         | 68.27                                            | 72.33         | 70.32                                            |
| Long press      | 71.45         | 68.74                                            | 73.54         | 70.96                                            |
| Slide up and down | 72.14     | 70.32                                            | 75.13         | 72.68                                            |
| Slide left and right | 70.83  | 69.96                                            | 76.25         | 73.45                                            |
| Rotate          | 76.85         | 74.57                                            | 80.68         | 76.63                                            |
| Zoom            | 74.93         | 73.42                                            | 82.46         | 79.94                                            |

Table 4. Experimental results comparing 3D ResNet-152 before and after the introduction of a pyramid context awareness mechanism.
References

1. Yatskiv S, Voytyuk I, Yatskiv N, et al. Improved method of software automation testing based on the robotic process automation technology. In: Proceedings of the 2019 9th international conference on advanced computer information technologies (ACIT), Ceske Budejovice, 5–7 June 2019, pp.293–296. New York: IEEE.

2. Ma YW, Lin DP, Chen SJ, et al. System design and development for robotic process automation. In: Proceedings of the 2019 IEEE international conference on smart cloud, Tokyo, Japan, 10–12 December 2019. New York: IEEE.

3. Maalla A. Development prospect and application feasibility analysis of robotic process automation. In: Proceedings of the 2019 IEEE 4th advanced information technology, electronic and automation control conference (IAEAC), Chengdu, China, 20–22 December 2019. New York: IEEE.

4. Yatskiv N, Yatskiv S and Vasylyk A. Method of robotic process automation in software testing using artificial intelligence. In: Proceedings of the 2020 10th international conference on advanced computer information technologies (ACIT), Deggendorf, 16–18 September 2020. New York: IEEE.

5. Abhishek KS, Qubeley LCF and Ho D. Glove-based hand gesture recognition sign language translator using capacitive touch sensor. In: Proceedings of the 2016 IEEE international conference on electron devices and solid-state circuits (EDSSC), Hong Kong, China, 3–5 August 2016, pp.334–337. New York: IEEE.

6. Krieter P and Breiter A. Analyzing mobile application usage: generating log files from mobile screen recordings. In: Proceedings of the 20th international conference on human-computer interaction with mobile devices and services, Barcelona, 3–6 September 2018. New York: ACM.

7. Zhang T, Liu Y, Gao J, et al. Deep learning-based mobile application isomorphic GUI identification for automated robotic testing. IEEE Softw 2020; 37(4): 67–74.

8. Xue F, Wu J and Zhang T. Learning-replay based automated robotic testing for mobile app. Mob Inform Syst 2022; 2022: 1084602.

9. Li J, Shi H and Hwang KS. Using fuzzy logic to learn abstract policies in large-scale multi-agent reinforcement learning. IEEE Trans Fuzz Syst. Epub ahead of print 27 April 2022. DOI: 10.1109/TFUZZ.2022.3170646.

10. Shi H, Li J, Mao J, et al. Lateral transfer learning for multiagent reinforcement learning. IEEE Trans Cybern. Epub ahead of print 10 September 2021. DOI: 10.1109/TCYB.2021.3108237.

11. Shi H, Li X, Hwang KS, et al. Decoupled visual servoing with fuzzy Q-learning. IEEE Trans Ind Inform 2018; 14(1): 241–252.

12. Park S and Kim D. Study on 3D action recognition based on deep neural network. In: Proceedings of the 2019 international conference on electronics, information, and communication (ICEIC), Auckland, New Zealand, 22–25 January 2019; pp.1–3. New York: IEEE.

13. Roig C, Sarmiento M, Varas D, et al. Multi-modal pyramid feature combination for human action recognition. In: Proceedings of the 2019 IEEE/CVF international conference on computer vision workshop (ICCVW), Seoul, South Korea, 27–28 October 2019, pp.3742–3746. New York: IEEE.

14. Yoo M, Na Y, Song H, et al. Motion estimation and hand gesture recognition-based human–UAV interaction approach in real time. Sensors 2022; 22(7): 2513.

15. Muezzinoglu T and Karakose M. Wearable glove based approach for human-UAV interaction. In: Proceedings of the 2020 IEEE international symposium on systems engineering (ISSE), Vienna, 12 October–12 November 2020, pp.1–6. New York: IEEE.

16. Sinha K, Kumari R, Priya A, et al. A computer vision-based gesture recognition using hidden Markov model. In: Chattopadhyay J, Singh R and Bhattacherjee V. (eds) Innovations in soft computing and information technology. Singapore: Springer, 2019, pp.55–67.

17. Zhu C, Yang J, Shao Z, et al. Vision based hand gesture recognition using 3D shape context. In: Proceedings of the 2018 IEEE international conference on robotics and biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018, pp.784–789. New York: IEEE.

18. Liu C and Szirányi T. Gesture recognition for UAV-based rescue operation based on deep learning, 2021, pp.180–187, https://pdfs.semanticscholar.org/b38c/115443b12d8ab3f65ecf948fceb52e7399.pdf?_ga=2.48293408.702853629.1658460100-1792531526.1619760732.

19. Human interface guidelines, 2022, https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/

20. Material design, 2022, https://material.io/design/interaction/gestures.html

21. Zhang C, Shi T, Ai J, et al. Construction of GUI elements recognition model for AI testing based on deep learning. In: Proceedings of the 2021 8th international conference on dependable systems and their applications (DSA), Yinchuan, China, 5–6 August 2021, pp.508–515. New York: IEEE.

22. Cheng J, Li J and Chen J. Research on recognition method of interface elements based on machine learning. In: Proceedings of the 2021 international conference on intelligent computing, automation and applications (ICAA), Nanjing, China, 25–27 June 2021, pp.233–236. New York: IEEE.

23. Baek YM and Bae DH. Automated model-based Android GUI testing using multi-level GUI comparison criteria. In: Proceedings of the 2016 31st IEEE/ACM international conference on automated software engineering (ASE), Singapore, 3–7 September 2016, pp.238–249. New York: IEEE.

24. Gu T, Sun C, Ma X, et al. Practical GUI testing of Android applications via model abstraction and refinement. In: Proceedings of the 2019 IEEE/ACM 41st international conference on software engineering (ICSE), Montreal, QC, Canada, 25–31 May 2019, pp.269–280. New York: IEEE.

25. Frister D, Oberweis A and Goranov A. Automated testing of mobile applications using a robotic arm. In: Proceedings of the 2020 international conference on computational science and computational intelligence (CSCI), Las Vegas, NV, 16–18 December 2020, pp.1729–1735. New York: IEEE.
26. Mao K, Harman M and Jia Y. Robotic testing of mobile apps for truly black-box automation. *IEEE Softw* 2017; 34(2): 11–16.

27. Ling L, Tao J and Wu G. Research on gesture recognition based on YOLOv5. In: *Proceedings of the 2021 33rd Chinese control and decision conference (CCDC)*, Kunming, China, 22–24 May 2021, pp.801–806. New York: IEEE.

28. Menghi C, Tsigkanos C, Pelliccione P, et al. Specification patterns for robotic missions, 2019, https://arxiv.org/pdf/1901.02077.pdf

29. Qin Z, Tang Y, Novak E, et al. MobiPlay: a remote execution based record-and-replay tool for mobile applications. In: *Proceedings of the 38th international conference on software engineering (ICSE)*, Austin, TX, 14–22 May 2016, pp.571–582. New York: IEEE.