Mech-DLK Software Manual

v2.5.0

Table of Contents

1. Getting Started . . . 1
   1.1. About Mech-DLK . . . 1
   1.2. Release Notes . . . 5
   1.3. System Requirements . . . 9
   1.4. Installation Guide . . . 10
   1.5. Terminology . . . 18
   1.6. Train Your First Model . . . 20
2. Use Algorithm Modules . . . 24
   2.1. Train an Instance Segmentation Model . . . 24
      2.1.1. Introduction . . . 24
      2.1.2. Use the Instance Segmentation Module . . . 26
      2.1.3. Introduction to Labeling Tools . . . 29
      2.1.4. Train a High-Quality Model . . . 33
   2.2. Train a Defect Segmentation Model . . . 43
      2.2.1. Introduction . . . 43
      2.2.2. Use the Defect Segmentation Module . . . 44
      2.2.3. Introduction to Labeling Tools . . . 48
      2.2.4. Train a High-Quality Model . . . 52
      2.2.5. Configure Defect Determination Rules . . . 54
   2.3. Train a Classification Model . . . 57
      2.3.1. Introduction . . . 57
      2.3.2. Use the Classification Module . . . 58
      2.3.3. ROI Tool . . . 61
      2.3.4. Train a High-Quality Model . . . 62
   2.4. Train an Object Detection Model . . . 68
      2.4.1. Introduction . . . 68
      2.4.2. Use the Object Detection Module . . . 69
      2.4.3. Introduction to Labeling Tools . . . 73
      2.4.4. Train a High-Quality Model . . . 76
   2.5. Train a Fast Positioning Model . . . 82
      2.5.1. Introduction . . . 82
      2.5.2. Use the Fast Positioning Module . . . 83
      2.5.3. Introduction to Labeling Tools . . . 87
   2.6. Train a Text Detection Model . . . 88
      2.6.1. Introduction . . . 88
      2.6.2. Use the Text Detection Module . . . 89
      2.6.3. Introduction to Labeling Tools . . . 93
      2.6.4. Train a High-Quality Model . . . 96
   2.7. Train a Text Recognition Model . . . 98
      2.7.1. Introduction . . . 98
      2.7.2. Use the Text Recognition Module . . . 100
      2.7.3. Introduction to Labeling Tools . . . 104

2.7.4. Train a High-Quality Model . . . 105
2.8. Train an Unsupervised Segmentation Model . . . 108
2.8.1. Introduction . . . 108
2.8.2. Use the Unsupervised Segmentation Module . . . 109
2.8.3. Introduction to Labeling Tools . . . 113
2.8.4. Train a High-Quality Model . . . 114
2.9. General Parameters . . . 117
2.9.1. Data Processing . . . 117
2.9.2. Model Training . . . 120
2.9.3. Model Validation . . . 123
3. Application Guide of Mech-DLK . . . 125
3.1. Usage Scenarios of Deep Learning . . . 125
3.2. Cascade Modules . . . 140
3.3. Use Models . . . 142
3.4. Model Iteration . . . 149
4. Appendix . . . 150
4.1. Menu Bar . . . 150
4.1.1. Perform Dataset Testing in Batches under Operation Mode . . . 151
4.2. Obtain a Trial Software License . . . 153
4.3. Update Software License . . . 155
4.4. Keyboard Shortcuts . . . 161
4.5. FAQ . . . 163

1. Getting Started

1.1. About Mech-DLK

Mech-DLK is machine vision deep learning software independently developed by Mech-Mind. With a variety of built-in industry-leading deep learning algorithms, it can solve many problems that traditional machine vision cannot handle, such as highly difficult segmentation, positioning, and classification.

Through intuitive and simple UI interactions, even without programming or specialized deep learning knowledge, users can quickly implement model training and validation with Mech-DLK.

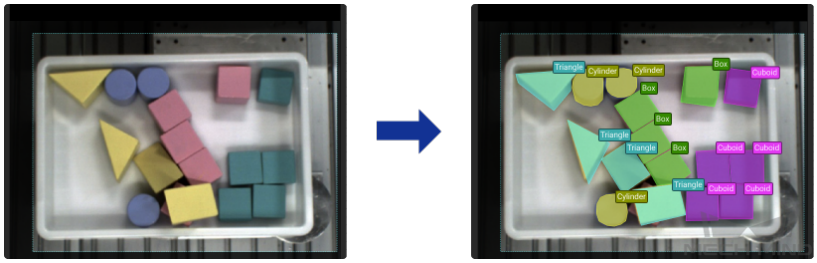
The software includes the following algorithm modules: **Fast Positioning**, **Defect Segmentation**, **Classification**, **Object Detection**, **Instance Segmentation**, **Text Detection**, **Text Recognition**, and **Unsupervised Segmentation**.

Instance Segmentation

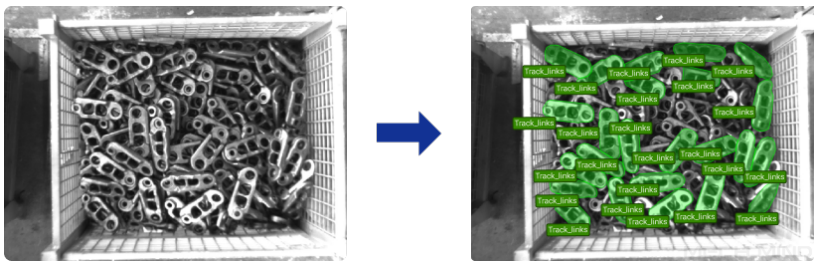
Segments the contour of target objects and outputs the corresponding labels of the classes.

This module can produce more refined segmentation results than the module Object Detection. The module can recognize single or multi-class objects and segment the corresponding contours. It is used for depalletizing, machine tending, piece picking, etc., and it cooperates with Mech-Vision and Mech-Viz to complete object picking.

- Segments blocks of various types.

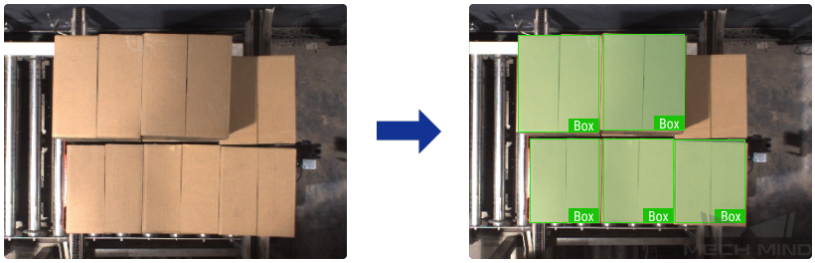


- Segments scattered and overlapping track links.



- Segments cartons closely fitted together.

© 2023 Mech-Mind Robotics Technologies Ltd.

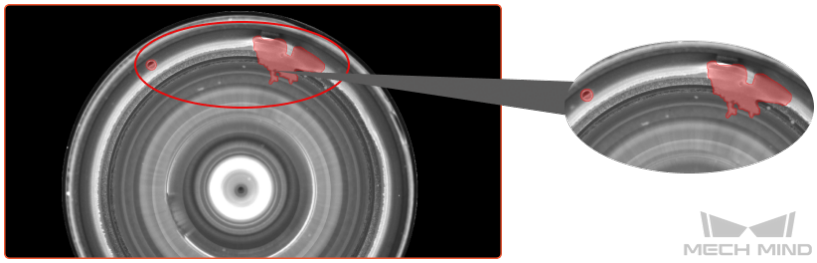


Defect Segmentation

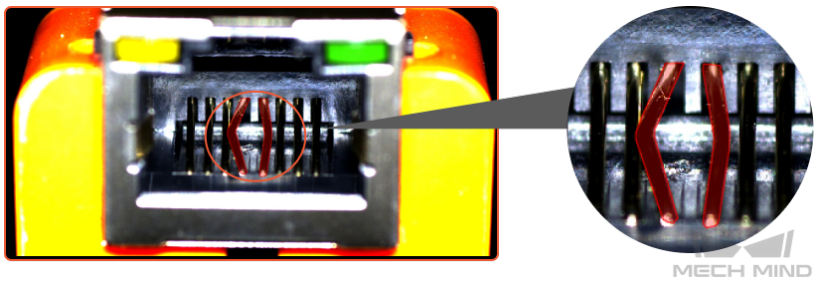
Detects and segments the defect regions in the image.

This module can be used to detect all types of defects. Defects that can be detected include surface defects such as stains, bubbles, and scratches and positional defects such as bending, abnormal shape, and absence. Moreover, this module can be applied even under complex conditions, such as small defects, complicated background, or random object positions.

- Detects air bubbles and glue spills on lenses.



- Detects bending defects of parts.



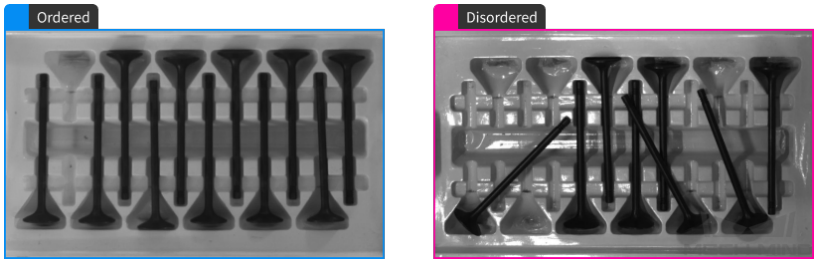
Classification

Recognizes object classes in images.

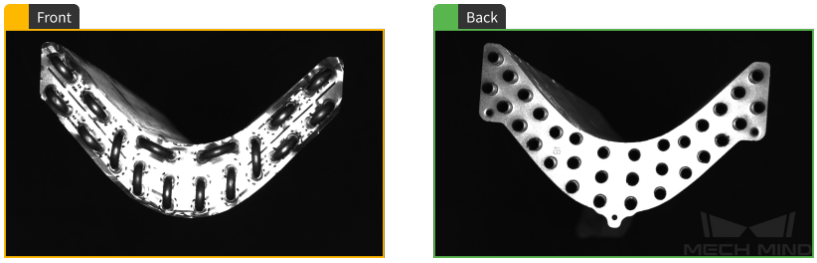
The module is used to recognize workobject front and back faces, workobject orientations, and defect types, and to recognize whether objects are missing, or whether objects are neatly arranged.

- Recognizes whether workobjects are neatly arranged or scattered.

© 2023 Mech-Mind Robotics Technologies Ltd.



- Recognizes the fronts and backs of objects.

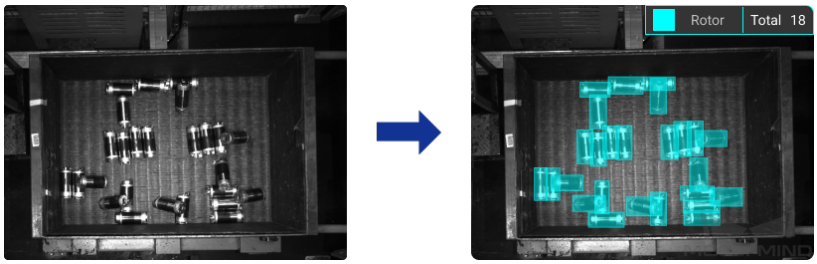


Object Detection

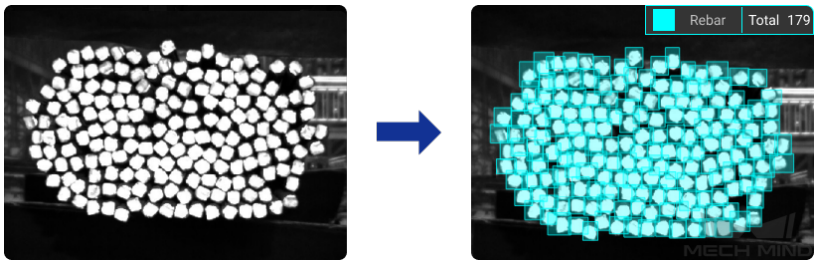
Detects the positions of all target objects and recognizes their categories at the same time.

This module is used to detect the absence of workobjects of fixed position, such as missing components in a PCB; it can also be used for object counting. Even for hundreds or thousands of objects, the module can quickly locate and count them.

- Detects the positions of rotors.



- Counts all rebars.



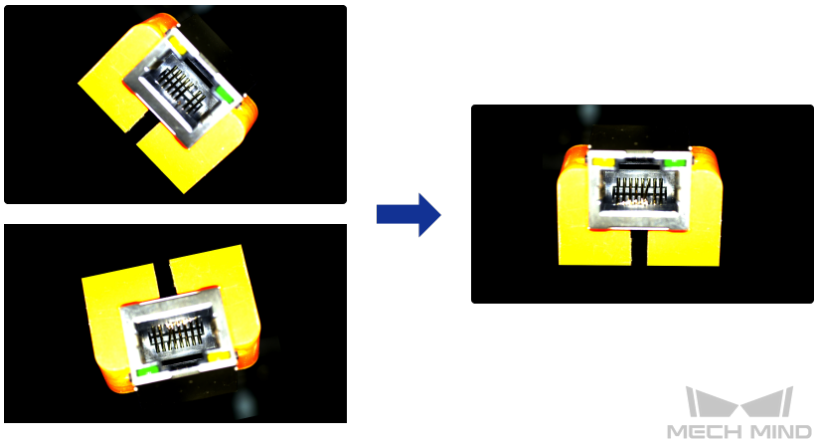
Fast Positioning

Recognizes the object orientation in an image and corrects the image based on the recognition result.

This module is used to detect objects in targeted regions of an image and rotate the image to a specified orientation and position.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Recognizes object orientations in images and rotates these images so that objects in them have a uniform orientation.

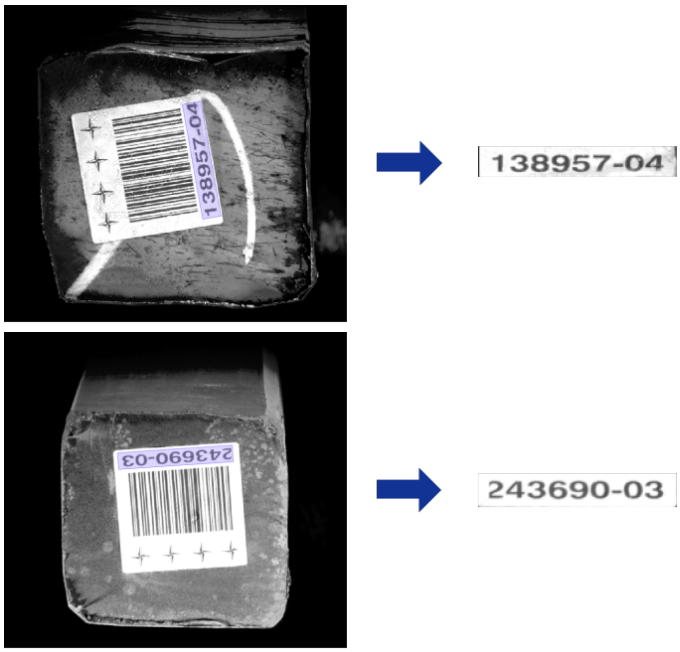


Text Detection

Detects the text areas of an image.

You can use the **Text Detection** module to detect the text area in an image and thus reduce the interference from the background. When the **Text Recognition** module is preceded by a **Text Detection** module, after selecting Import › Import from previous module, make sure the **Rectify image(s)** option is enabled to rectify images to the positive direction (0°).

- Detects the text areas of an image.



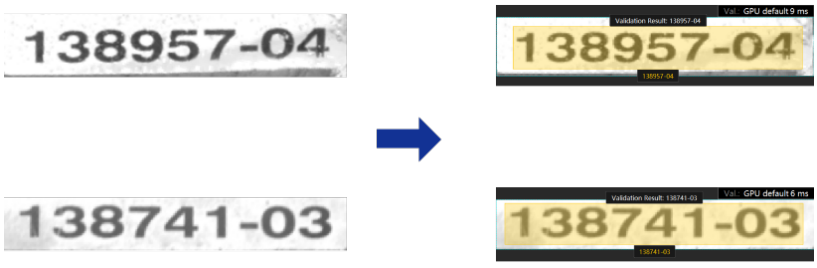
Text Recognition

Recognizes the text in an image, including letters, numbers, and some special symbols.

This module is used to recognize the characters in the text area of an image, often applied in the fast recognition of codes on all kinds of labels.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Recognizes the characters in an image.

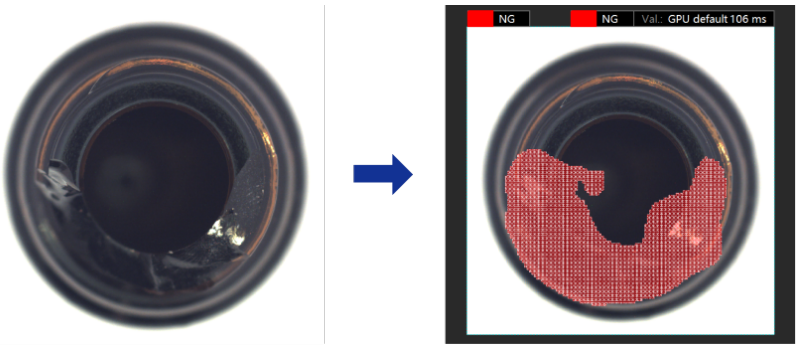


Unsupervised Segmentation

Judges whether an image is OK, NG, or Unknown according to set thresholds and displays the possible areas with defects.

Images should be labeled as OK or NG, and only OK images will be used as the training set for model training. The trained model can help judge whether there is any defect in an image.

- Judges whether an image is OK, NG, or Unknown according to set thresholds.



1.2. Release Notes

Mech-DLK 2.5.0 Release Notes

This section introduces the new features and improvements of Mech-DLK 2.5.0.

New Features

Added Text Detection and Text Recognition Modules

Mech-DLK 2.5.0 now features optical character recognition (OCR) that is realized by cascading the following two algorithm modules:

- Text Detection: the first module, which is used to detect a single line or multiple lines of text in an image.
- Text Recognition: the second module, which is used to recognize the characters in the text area.

This feature is usually employed to detect and recognize tiny characters on electronic components, vehicle-related data against complex backgrounds, and serial numbers and lot numbers displayed on the packaging.

© 2023 Mech-Mind Robotics Technologies Ltd.

Added the Unsupervised Segmentation Module

Mech-DLK 2.5.0 now features the Unsupervised Segmentation module, which only requires OK images in the training set to achieve pixel-level detection of all known and unknown defects. The trained model can be used to judge whether the image of an object is OK, NG, or Unknown on the basis of set thresholds. Moreover, heat maps are available to show the possible areas with defects.

This module is usually used for industrial quality inspection in scenarios where

- obtaining OK images is easy while getting NG images is difficult, and
- defect types are uncertain.

Added Free Rectangle Tool

In the **Text Detection** module, the **Free Rectangle Tool** can be used to draw a rectangle around the text area. It is recommended to select this tool for rectangular text areas.

Added Text Recognition Tool

In the **Text Recognition** module, the **Text Recognition Tool** can be used to define the text recognition range and automatically generate the recognition result.

Added OK/NG Labels

Mech-DLK 2.5.0 now allows to directly label images as OK or NG with the **OK/NG Labels**. Select an image from the image list and click the **OK Label** or **NG Label** on the toolbar. The image will be labeled as OK or NG.

- Both **OK Label** and **NG Label** are available in the **Unsupervised Segmentation** module.
- **OK Label** is available in the **Defect Segmentation** module.

Improvements

Support for NVIDIA GeForce RTX 40 Series GPUs

Mech-DLK 2.5.0 now fully supports NVIDIA GeForce RTX 40 Series GPUs for model training and validation.

Improved the Floating Window of Keyboard Shortcuts

In Mech-DLK 2.5.0, the floating window of keyboard shortcuts displays the shortcuts by modules for easy and fast search.

Mech-DLK 2.4.2 Release Notes

Added region-specific license control. Click Help › About to view the details.

Mech-DLK 2.4.1 Release Notes

New Features

© 2023 Mech-Mind Robotics Technologies Ltd.

Added the Cascade Mode

Mech-DLK 2.4.1 has added a brand-new cascade mode of modules, which enables the combination of modules to solve deep learning problems in complex scenarios. It should be noted that when the **Fast Positioning** module is involved, it must be the first module. For example, when you need to detect the positions of defects and classify these defects, you can first add a **Defect Segmentation** module and then add a **Classification** module. In addition, when importing data from the previous module, you can select images according to actual needs and configure the import of these images in the Import window.

Added the Training Center

The **Training Center** supports the training of models in a queue, which is suitable for scenarios requiring the training of multiple models. With the **Training Center**, the software can train models in sequence, with no need for manual clicking of **Train** repeatedly, which can save a huge amount of time.

Added the Mask Type of Mask Globally and Supported Custom Mask Fill

In the **Defect Segmentation** module, when you select the **Mask Polygon Tool**, the mask type can be **Mask single image** or **Mask globally**. In addition, you can customize the mask color.

- **Mask single image**: The mask is only displayed in the current image. It is only valid in training.
- **Mask globally**: After a mask is drawn in the current image, the mask will be displayed in all images. The mask is valid in both training and validation.

Added the Floating Window of Keyboard Shortcuts

Click in the lower-right corner of the selection region to open the window of keyboard shortcuts.

Added Auxiliary Labeling Lines for Rectangle Tool

In the **Instance Segmentation** and **Object Detection** modules, auxiliary labeling lines are added for **the Rectangle Tool** to assist rectangular selection on images.

Added the Display and Filtering of Validation Result Confidence

In the **Instance Segmentation** and **Object Detection** modules, a confidence filtering function is added for validation results. You can adjust the confidence to filter the validation results and then evaluate the accuracy of models.

Improvements

Optimized the Classification Module

The **Classification** module is optimized, which leads to faster training convergence and a growth rate of 20% in accuracy under complex scenarios.

Optimized Mech-DLK SDK

Mech-DLK SDK is more stable and easier to use after being restructured. Mech-DLK SDK supports the inference based on the cascaded modules and the switching between different operating hardware. In addition, it provides richer samples for reference.

© 2023 Mech-Mind Robotics Technologies Ltd.

Optimized the Setting of Defect Determination Rule

The defect determination rule in the **Defect Segmentation** module is optimized. Click here to view the details.

Enabled the Setting of Translation in the Fast Positioning Module

In the **Image Adjustment** window of the **Fast Positioning** module, you can translate the image along the X and Y axes. After training, images with objects in specified positions and orientations will be generated, which meet the requirements of more application scenarios.

Optimized the Template Tool

In the Instance Segmentation and Object Detection modules, after selecting the Template Tool, press and hold the Shift key and scroll the mouse wheel to adjust the angle of the template. You can also set the Rotation angle to achieve the same purpose.

Release Notes of Previous Versions

▼ *Click here to view the Mech-DLK 2.3.0 release notes*

Mech-DLK 2.3.0 Release Notes

- Graphics Card Driver Requirement

  Before using Mech-DLK 2.3.0, please upgrade the graphics card driver to 472.50 or above.

- Improved the Training Speed

  Optimized the algorithms, and thus significantly improved the speed of model training. Only the optimal model is saved during training, and the training cannot be stopped halfway.

- Added the Smart Labeling Tool

  For modules including Defect Segmentation, Instance Segmentation, and Object Detection, you can do smart labeling by selecting the Smart Labeling Tool, clicking the objects to be labeled, right-clicking to undo the redundant selection, and pressing the Enter key to complete the labeling.

- Added the Function of Adding/Removing Vertices for the Polygon Tool

  For the Instance Segmentation and Object Detection modules, after labeling with the Polygon Tool, if the selection needs to be modified, you can left-click the line segment between two vertices to add a vertex, or right-click a vertex to remove it.

- Added the Template Tool

  For the Instance Segmentation and Object Detection modules, you can use the Template Tool to set the selection as a template. The template can be applied by simply clicking the images. It is suitable for scenarios where there are multiple neatly arranged objects of the same type in an image, and it improves labeling efficiency.

- Added the Function of Preview by Zooming

  Support previewing full images and cropped cell images.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Optimized the Grid Cutting Tool

  Optimized the Grid Cutting Tool. After cutting the image by the grid, you can select a cell image by checking the box in the upper left corner of the cell image, and you can preview the image by clicking on the button in the upper right corner of the cell.

- Optimized the Data Filtering Mechanism

  Added options for filtering results: “Correct results”, “Wrong results”, “False negative”, and “False positive”. Added options for filtering data types: “Labeled as OK” and “Labeled as NG”.

- Built-in Deep Learning Environment

  The deep learning environment is built into the software Mech-DLK, and the models can be trained without a separately installed environment.

▼ *Click here to view Mech-DLK 2.2.1 release notes*

Mech-DLK 2.2.1 Release Notes

- Added the Function of Showing the Class Activation Maps for Module Classification

  After the model is trained, click **[Generate CAM]**. The class activation maps show the weights of the features in the form of heat maps; the model classifies an image into its class according to these features. Image regions with warmer colors have higher weights for classifying the image into its class.

- Supported Validation and Export of CPU Models
  - **Classification and Object Detection**: After training is completed, select the deployment device as CPU or GPU before exporting the model.
  - **Instance Segmentation**: Before training the model, set the training parameters. When exporting a model, select the deployment device as CPU/GPU:
    - CPU lightweight model: Before training the model, set the training parameter **Model type** to **Lite (better with CPU deployment)**. When exporting the model for deployment, set **Deployment device** to **CPU** or **GPU**.
    - GPU standard model: Before training the model, set the training parameter **Model type** to **Normal (better with GPU deployment)**. When exporting the model for deployment, set **Deployment device** to **GPU**.

1.3. System Requirements

It is recommended that the device that runs Mech-DLK should satisfy the following requirements.

Authorized dongle version	Pro-Run	Pro-Train/Standard
Operating system	Windows 10 or above	
CPU	Intel® Core™ i7-6700 or above	
Memory	8GB or above	16GB or above
Graphics card	GeForce GTX 1660 or above	GeForce RTX 3060 or above
Graphics card driver	Version 472.50 or above	

© 2023 Mech-Mind Robotics Technologies Ltd.

The Pro-Run version features Mech-DLK SDK deployment, labeling, and Operation Mode. The Pro-Train version supports all features, including module cascading, labeling, training, validation, and Mech-DLK SDK deployment, while the Standard version supports the features of labeling, training, and validation.

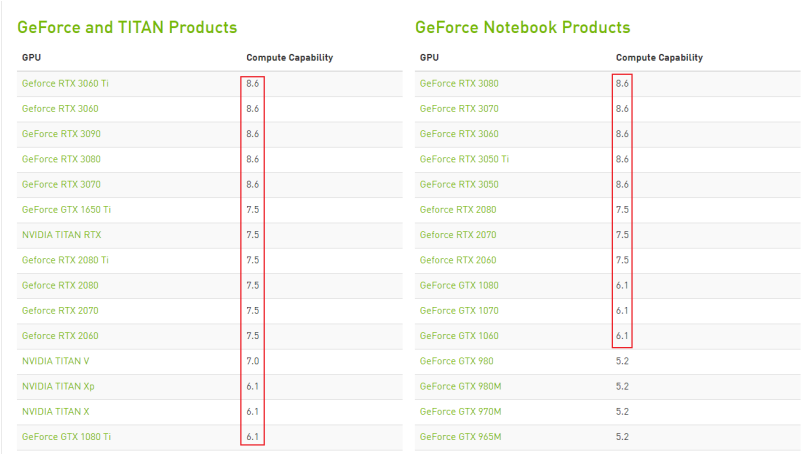
When Mech-DLK is used, the resolution and scale of the screen should follow the correspondence in the table below. If the resolution and scale of your screen do not match those in the table below, some display problems may occur.

If you are using dual monitors, make sure that both have the same resolution and scale.

Resolution	Scale
1280 × 800 (16:10)	100%
1920 × 1080 (16:9)	100% or 125%
2560 × 1440 (16:9)	125% or 150%
3840 × 2160 (16:9)	150% or 175%

Compute Capability Requirements for the Graphics Card

- The computer graphics card's compute capability should be 6.1 at least.
- Click here to check the compute capability of your GPU:



GeForce and TITAN Products

GPU	Compute Capability
Geforce RTX 3060 Ti	8.6
Geforce RTX 3060	8.6
GeForce RTX 3090	8.6
GeForce RTX 3080	8.6
GeForce RTX 3070	8.6
GeForce GTX 1650 Ti	7.5
NVIDIA TITAN RTX	7.5
Geforce RTX 2080 Ti	7.5
Geforce RTX 2080	7.5
Geforce RTX 2070	7.5
Geforce RTX 2060	7.5
NVIDIA TITAN V	7.0
NVIDIA TITAN Xp	6.1
NVIDIA TITAN X	6.1
GeForce GTX 1080 Ti	6.1

GeForce Notebook Products

GPU	Compute Capability
GeForce RTX 3080	8.6
GeForce RTX 3070	8.6
GeForce RTX 3060	8.6
GeForce RTX 3050 Ti	8.6
GeForce RTX 3050	8.6
Geforce RTX 2080	7.5
Geforce RTX 2070	7.5
Geforce RTX 2060	7.5
GeForce GTX 1080	6.1
GeForce GTX 1070	6.1
GeForce GTX 1060	6.1
GeForce GTX 980	5.2
GeForce GTX 980M	5.2
GeForce GTX 970M	5.2
GeForce GTX 965M	5.2

1.4. Installation Guide

This chapter introduces how to install Mech-DLK and device prerequisites.

Install Mech-DLK

1. You can download the installation package of Mech-DLK from Mech-Mind Download Center.
2. Double-click the `Mech-DLK Installer x.x.x.exe` file and install Mech-DLK following the instructions.

Prerequisites for Using the Software

To make sure Mech-DLK can function properly, please complete the following actions.

© 2023 Mech-Mind Robotics Technologies Ltd.

Check Interfaces and Drivers

Check the following in Windows **Control Panel**:

1. **Network and Internet**: make sure that the network interfaces used to connect to other devices are functioning properly.
2. **Device Manager**: under **[Network adapters]** and **[Display adapters]**, make sure the required drivers for your network interfaces and GPU are installed.

Set up Software License

Mech-Mind uses CodeMeter from Wibu-Systems as the license system for its software. The CodeMeter installer has been included in the software package.

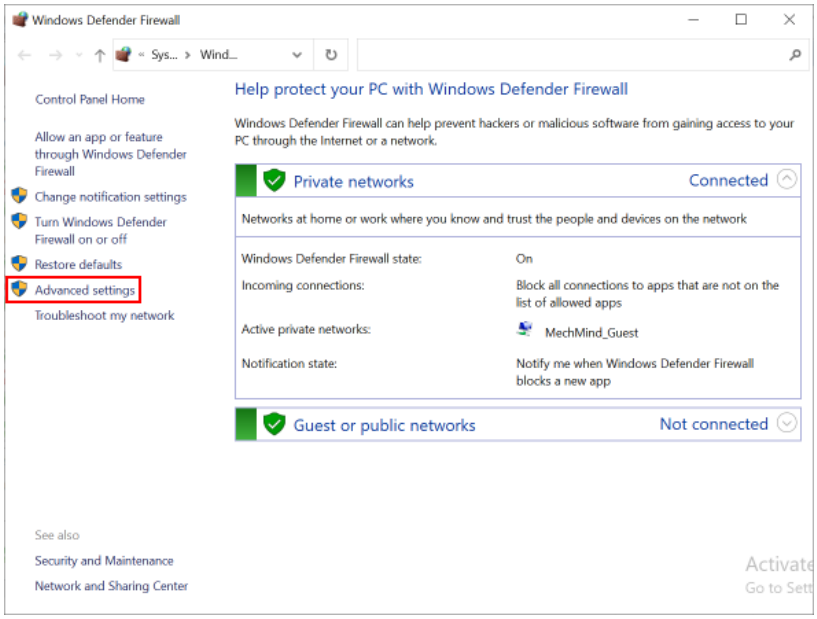
1. Plug the license dongle you received into the IPC.
2. Run the CodeMeter installer to install CodeMeter.
3. Make sure that CodeMeter is running: in the system tray, check if the CodeMeter icon is displayed in the Windows tray.

- If you need to obtain a trial license that does not require a license dongle, please click Obtain a Trial Software License.
- If you need to update an existing license, please click Update Software License.

Turn off Windows Defender Firewall

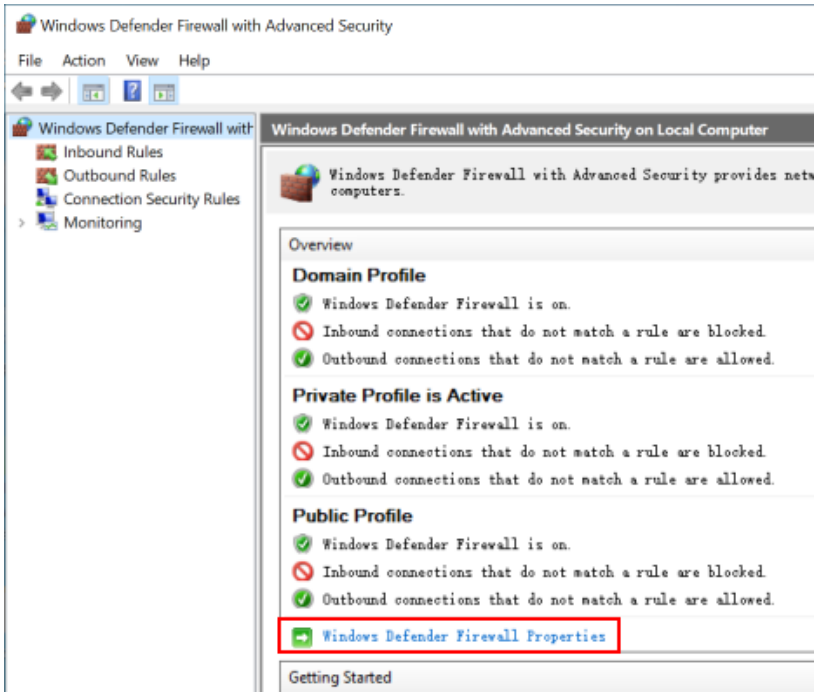
The Windows Defender Firewall might block the normal communication between Mech-Mind software and devices connected to the IPC. Therefore, it is necessary to turn off Windows Defender Firewall's protection for the network interfaces connected to devices that communicate with the software. Follow these steps:

1. Open the control panel, and click System and Security › Windows Defender Firewall › Advanced Settings.

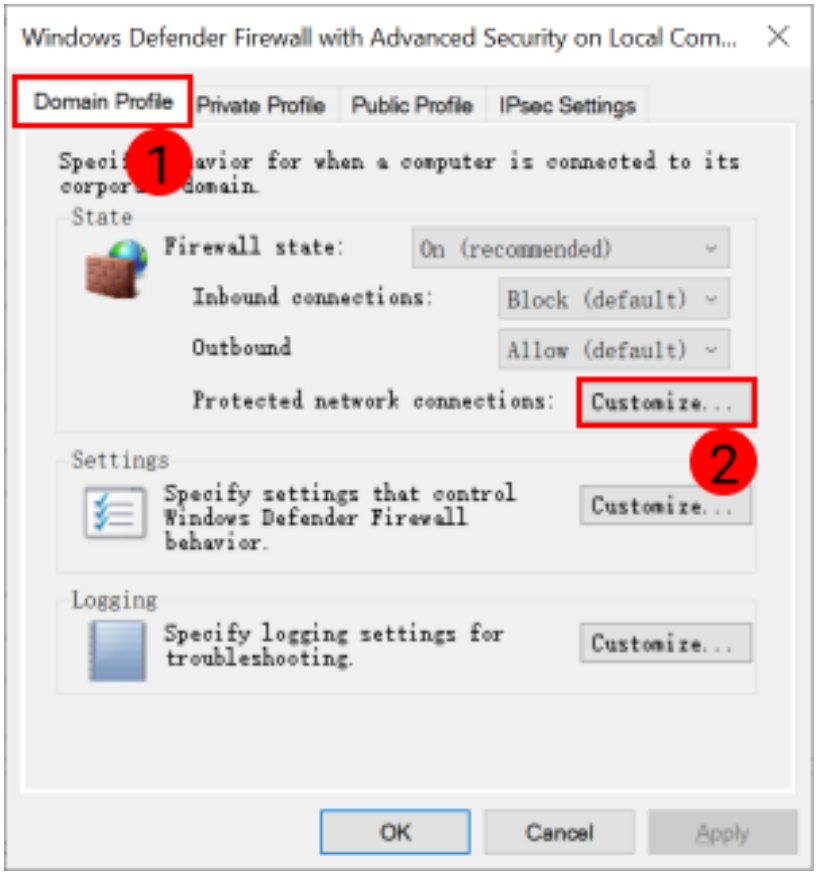


2. In the pop-up window, click **Windows Defender Firewall Properties**.

© 2023 Mech-Mind Robotics Technologies Ltd.

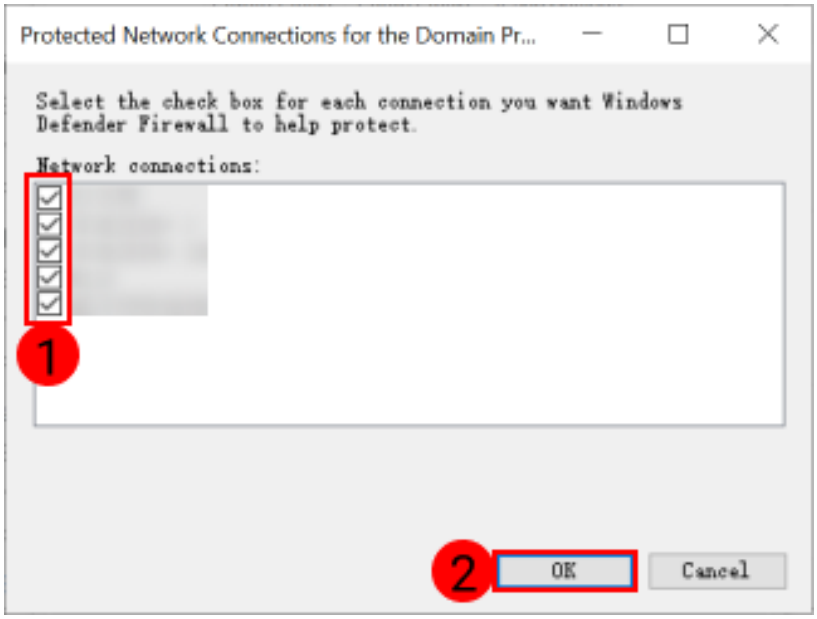


3. Under the **Domain Profile** tab, click **[Customize...]** next to **Protected network connections**.

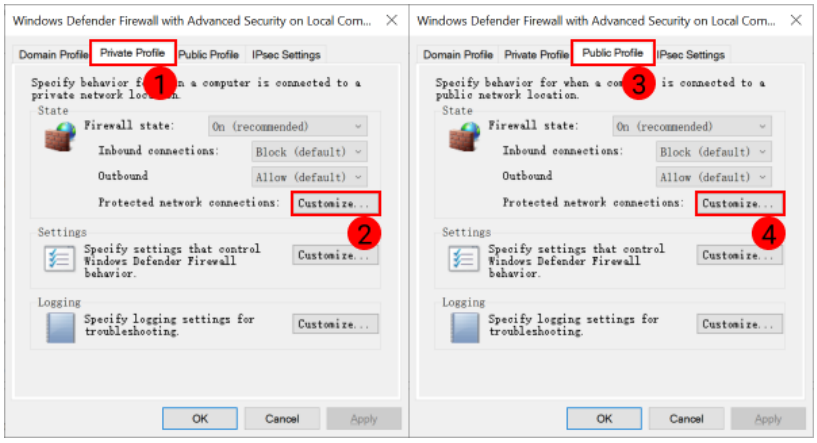


4. In the pop-up window, uncheck all the network interfaces connected to devices that communicate with the software. Then, click **[OK]**.

© 2023 Mech-Mind Robotics Technologies Ltd.



5. Repeat the above two steps for the **Private Profile** and **Public Profile** tabs.



Prevent Windows from Updating (Recommended)

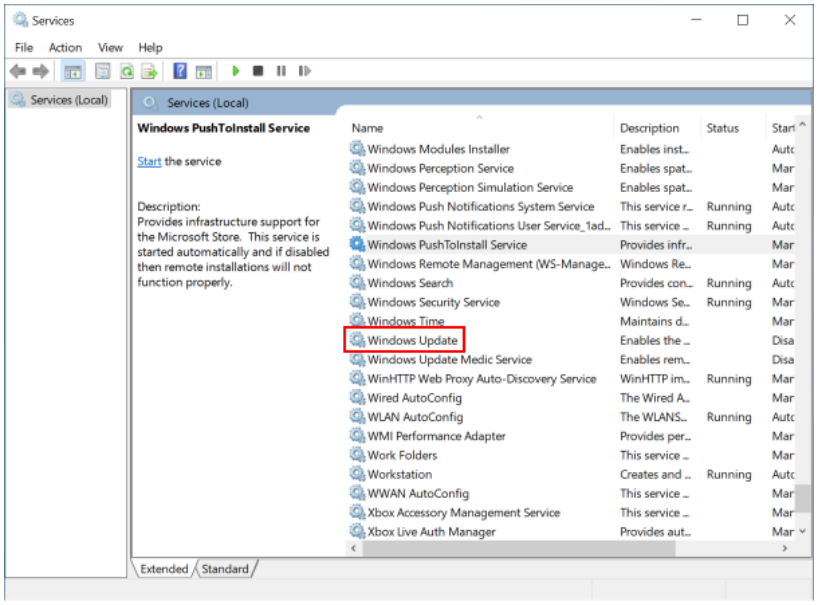
Windows updates might force the IPC to shut down/restart during production in order to complete the updates, which would end the software and thus affect normal production. Therefore, it is highly recommended to prevent Windows from updating to avoid unexpected shutdown.

If you choose to keep Windows Update enabled, please take measures to ensure that IPC shutdown/restart occurs during planned downtime, such as setting active hours for Windows Update.

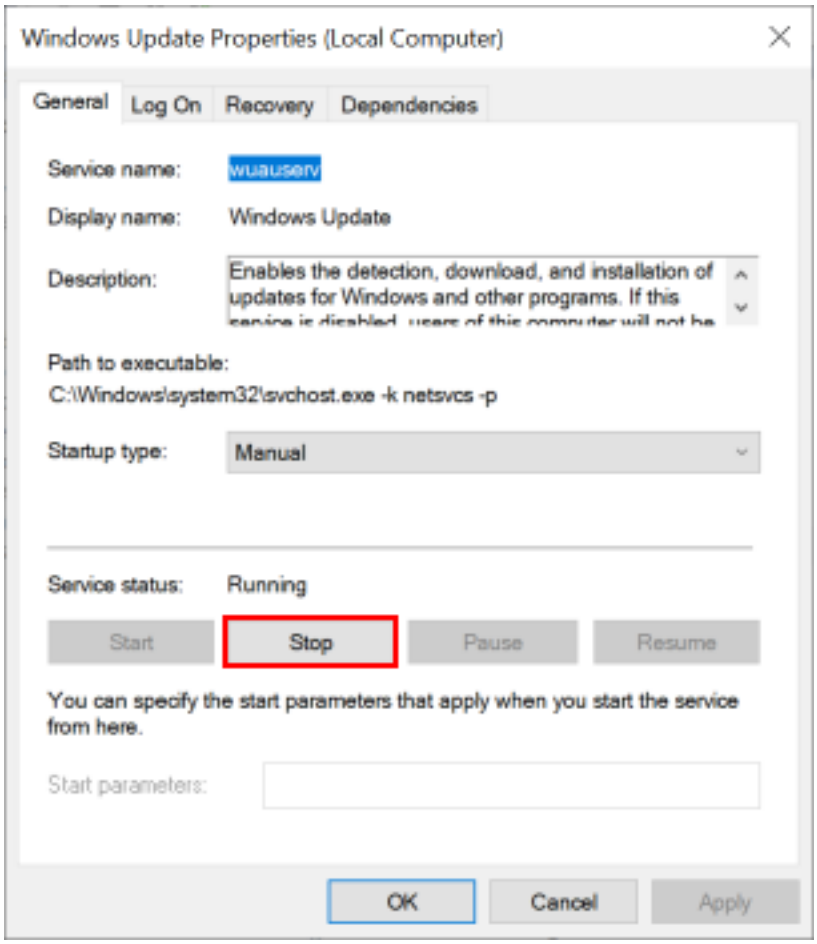
Disable Windows Update

1. Click the magnifying glass icon in the taskbar, and search for **Services**.
2. Click **Services** in the search results to open it, and scroll down to find **Windows Update**.

© 2023 Mech-Mind Robotics Technologies Ltd.

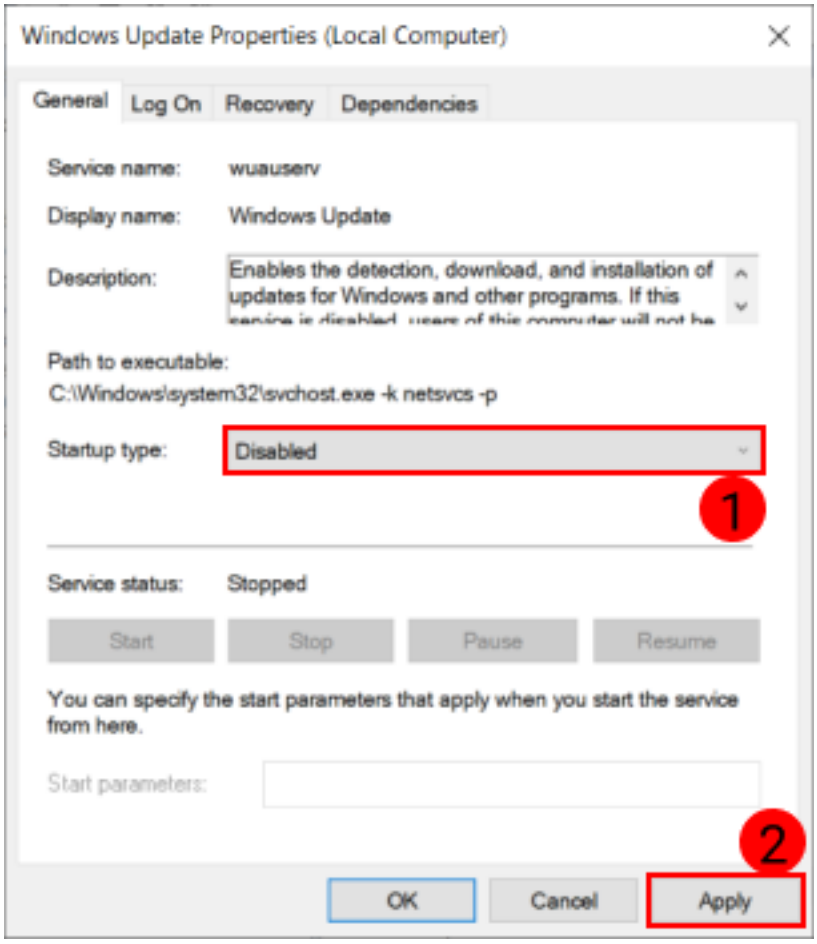


3. Click **[OK]** after clicking the Stop button.



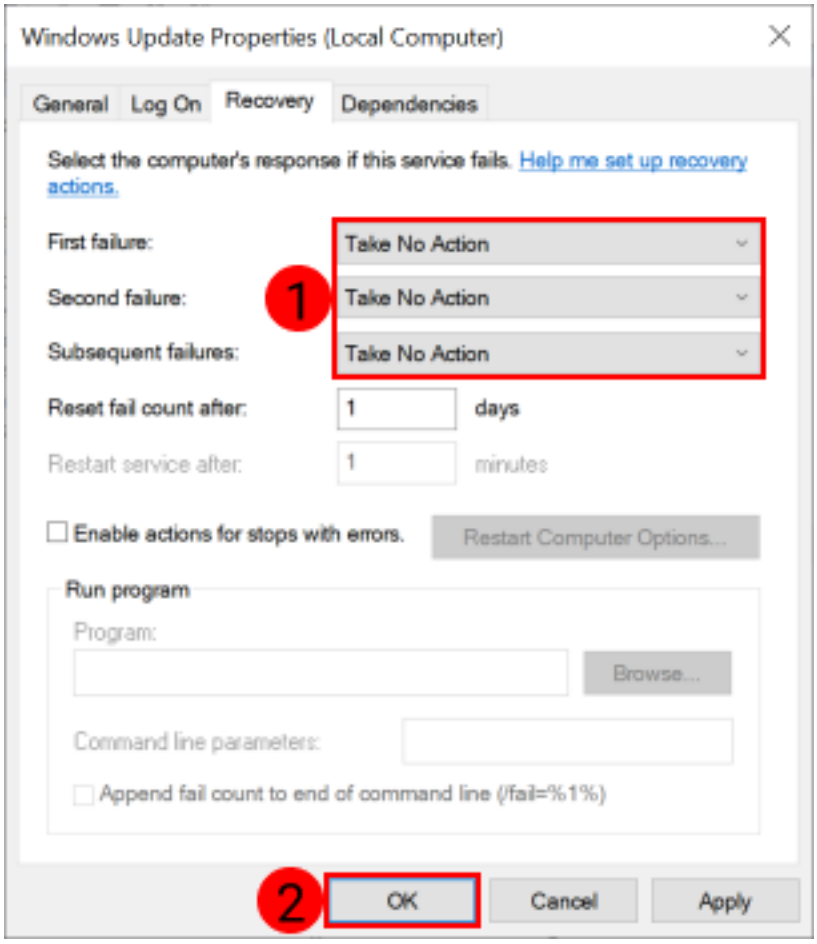
4. In the drop-down menu of **Startup type**, select **Disabled**, and then click **[Apply]**.

© 2023 Mech-Mind Robotics Technologies Ltd.



5. Click the **Recovery** tab, and change the response for all failures to **Take No Action**. Then, click [OK].

© 2023 Mech-Mind Robotics Technologies Ltd.



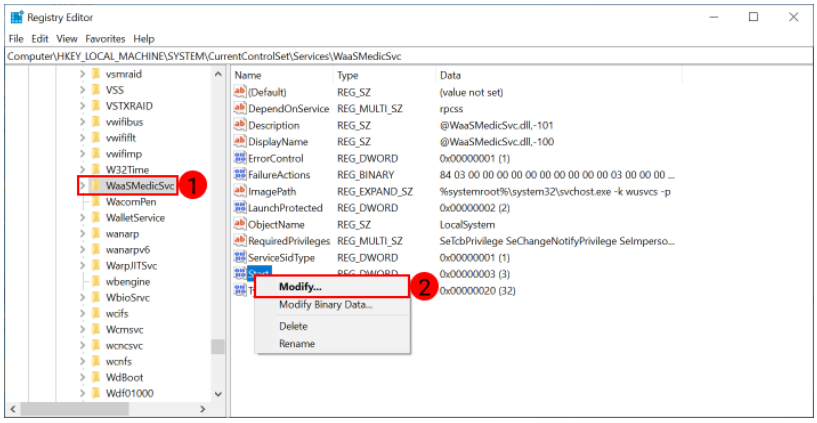
Disable Windows Update Medic Service

Windows Update Medic Service fixes problems in Windows Update and ensures that your computer continues to receive updates. That is, even if you have disabled Windows Update, Windows Update Medic Service will eventually re-enable it. Therefore, it is necessary to disable Windows Update Medic Service as well.

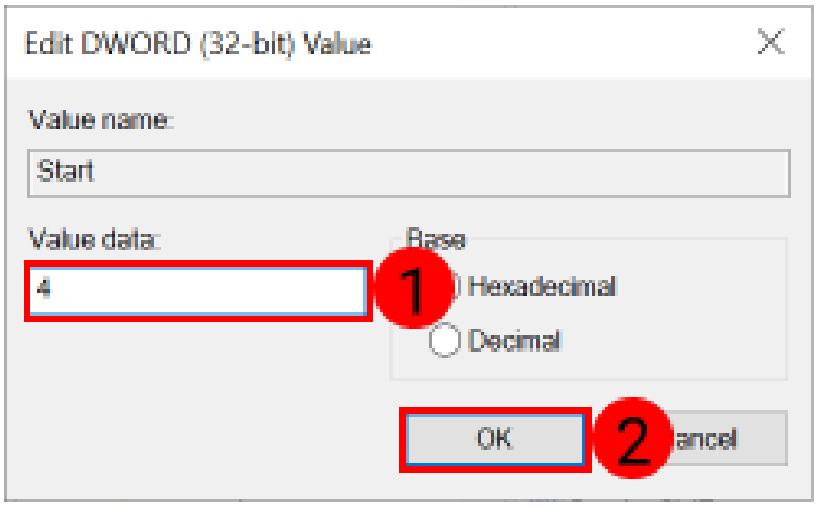
Windows does not allow you to disable Windows Update Medic Service through simple button-clicking. To disable it, you'll need to go to the Registry Editor as instructed below.

1. Click the magnifying glass icon in the taskbar, and search for "regedit". Click **Registry Editor** in the search result to open it.
2. In the left pane, navigate to HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WaaSMedicSvc. Then in the right pane, right-click **Start** and select **Modify**.

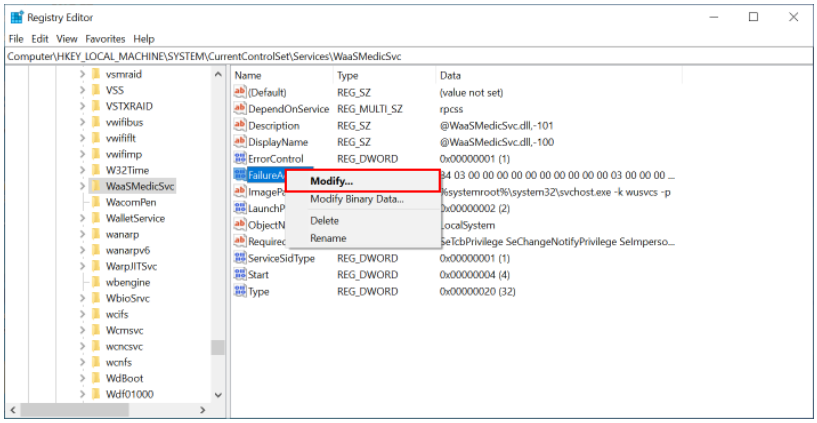
© 2023 Mech-Mind Robotics Technologies Ltd.



3. In the pop-up window, change **Value data** to 4, and then click **[OK]**.

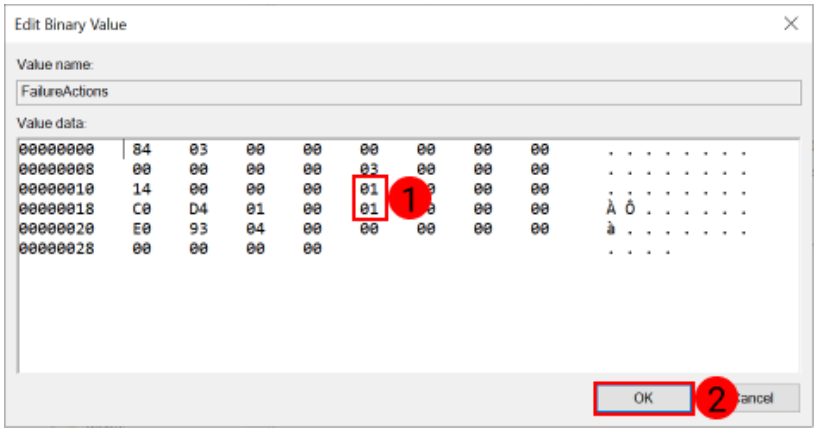


4. Right-click **FailureActions** and select **Modify**.

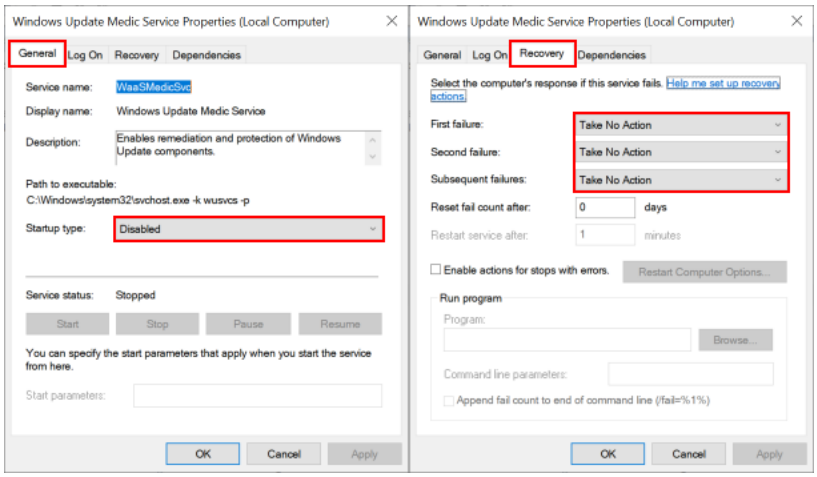


5. In the pop-up window, change the values (whose positions are shown in the middle box of the following figure) to **01**. Then, click **[OK]**.

© 2023 Mech-Mind Robotics Technologies Ltd.



6. Return to **Services**, find **Windows Update Medic Service** and double-click to open it.
7. The Startup type in the General tab should be Disabled, and all responses for failures in the Recovery tab should be Take No Action.



Now, you can start to use Mech-DLK.

1.5. Terminology

- ROI

  A region of interest (ROI) is a region selected from an image. The region is the focus of the analyses of the images. Selecting regions of interest helps reduce processing time and improve accuracy.

- Labeling

  Labeling refers to the process of selecting object features or contours from images and adding labels indicating features or defects to the selections, and the process of adding labels to individual images, thus telling the model what contents it should learn.

- Dataset

  A dataset contains the original data and labels. In Mech-DLK, datasets are saved in dlkdb files.

- Unlabeled data

© 2023 Mech-Mind Robotics Technologies Ltd.

Original data without labels.

- Training set

  The part of the dataset allocated for model training.

- Validation set

  The part of the dataset allocated for model validation.

- OK image

  In defect defection, an OK image is an image that contains no defects.

- NG image

  In defect detection, an NG image is an image that contains any defects.

- Training

  Training refers to the process of letting the model learn on the training set.

- Validation

  Validation refers to the process of verifying the trained model on the validation set.

- Accuracy

  Accuracy refers to the ratio of the number of correctly predicted samples to the total number of samples in the validation set when validating the trained model.

- Loss

  Loss is a measure of the inconsistency between the model's predictions on the validation set and the ground truth of the validation set.

- Epochs

  The number of times the model goes through all data in the training set when training. The model completes one epoch when it has gone through all training data once.

- False positive (FP)

  The situation where an image not containing defects is predicted as an image containing defects.

- False negative (FN)

  The situation where an image containing defects is predicted as an image not containing defects.

- Defect confidence

  The probability that there are defects in an image.

- Super model

© 2023 Mech-Mind Robotics Technologies Ltd.

General-purpose models provided by Mech-Mind, which are used to recognize cartons or sacks. If any one of them works poorly, Mech-DLK can be used to fine-tune the model.

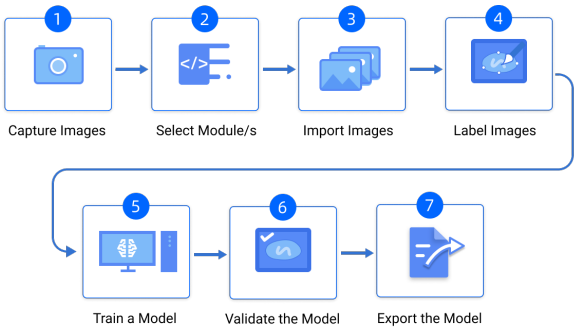
1.6. Train Your First Model

This section shows how to train and export an example deep learning model that can be used for defect segmentation. The data used for training is from an image dataset of Ethernet ports.

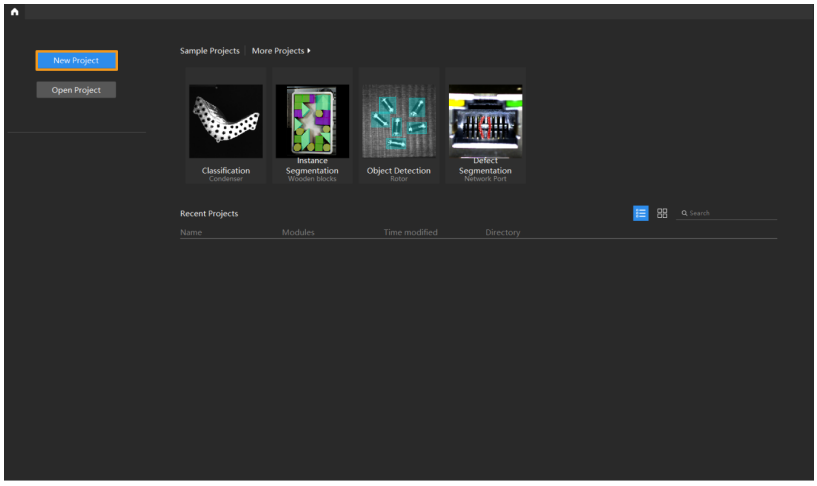
Prerequisites

Click here to download the image data and unzip the file.

Training Process



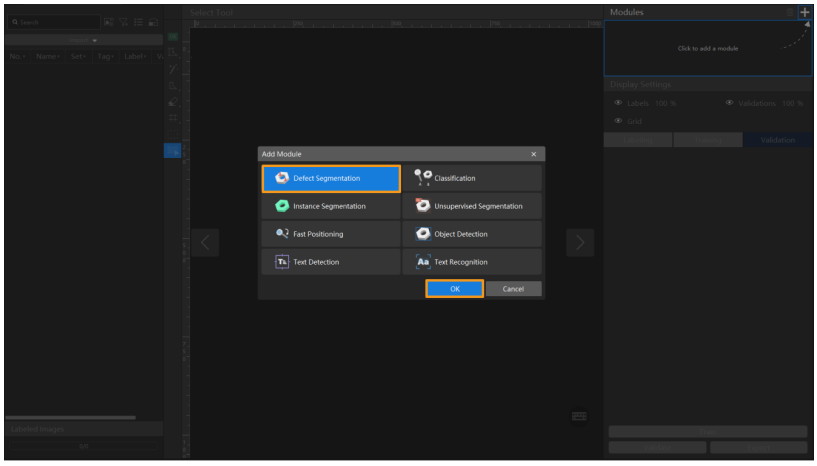
1. **Create a New Project**: Click **[New Project]** in the interface, name the project, and select a directory to save the project.



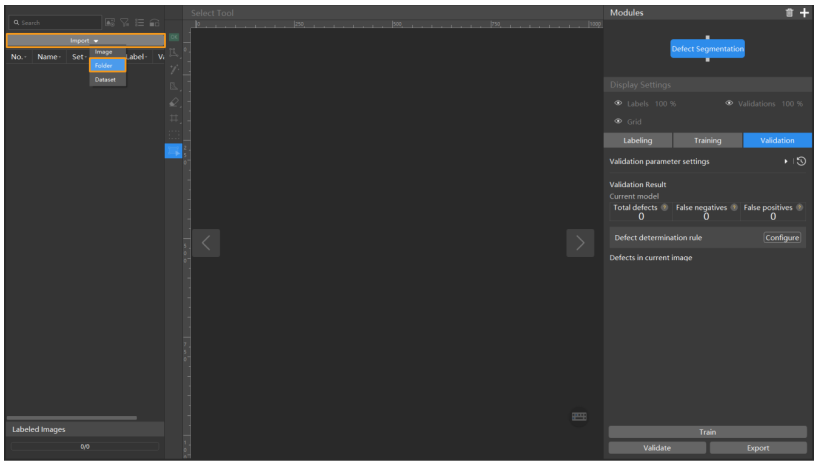
Do NOT include Chinese characters in the path name.

2. **Add the Defect Segmentation Module**: Click + to add a module. Select **[Defect Segmentation]** and then click the OK button.

© 2023 Mech-Mind Robotics Technologies Ltd.

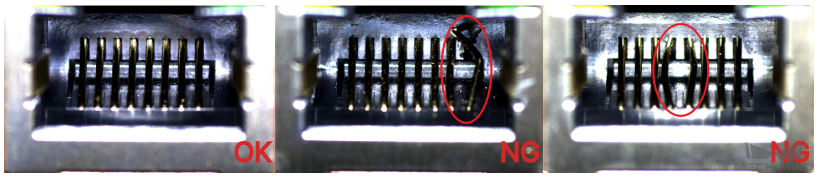


3. **Import Image Data**: Click **[Import]** in the upper-left corner, select **[Folder]** and import the image dataset you have downloaded.



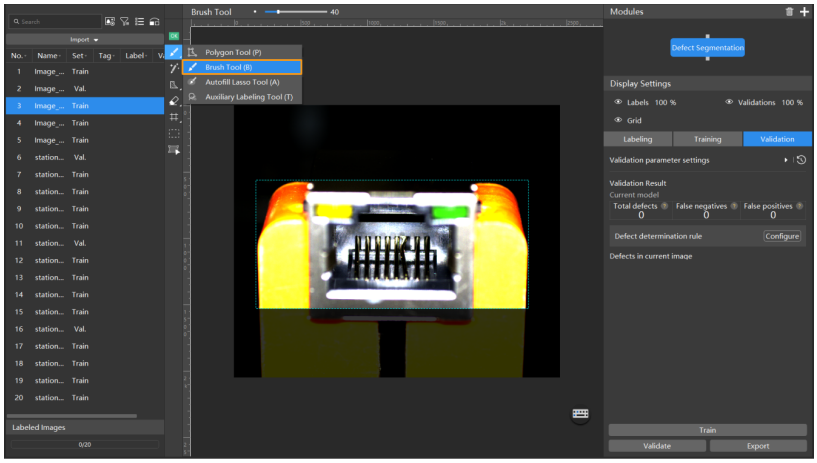
- Image: Import image(s).
- Folder: Import all images from the selected folder.
- Dataset: A dataset contains images and corresponding labels. You can generate one by clicking File › Export Dataset.

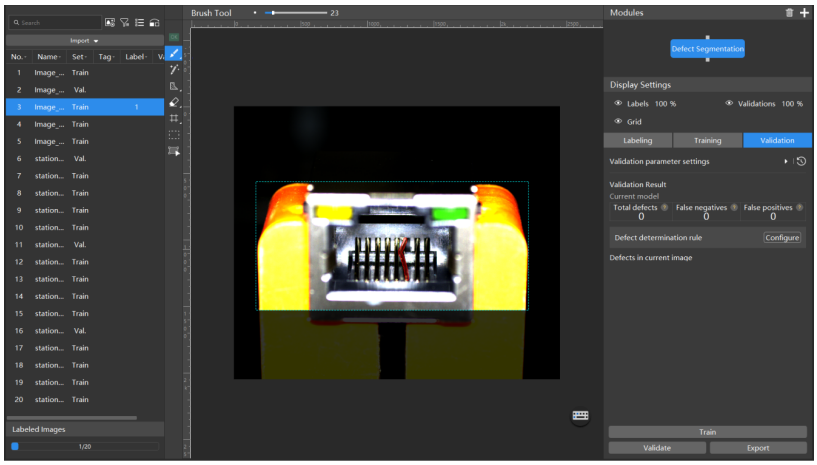
4. **Labeling**: In this example, you will need to label the OK images and NG images in each dataset. OK means that the connectors meet quality requirements and NG means that there are defects such as deformations and fractures on the Ethernet ports. Labeling is to provide the information required by deep learning training.



For NG images, long press the left mouse button or right-click on the toolbar and then use the Brush Tool to select the regions with defects.

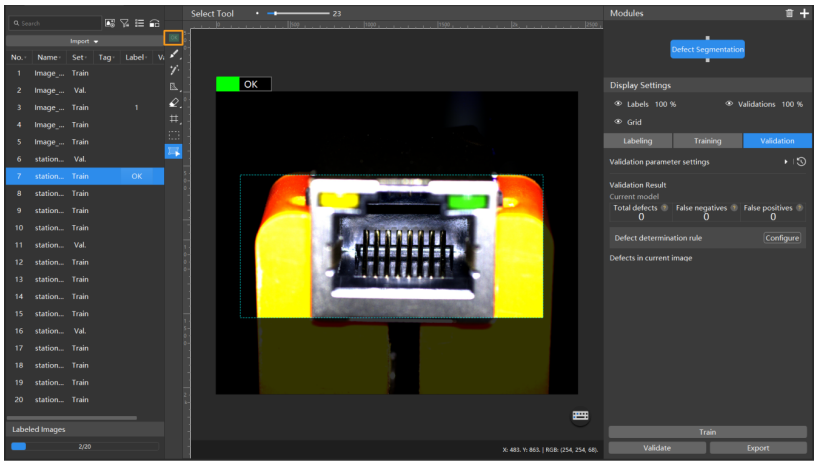
© 2023 Mech-Mind Robotics Technologies Ltd.





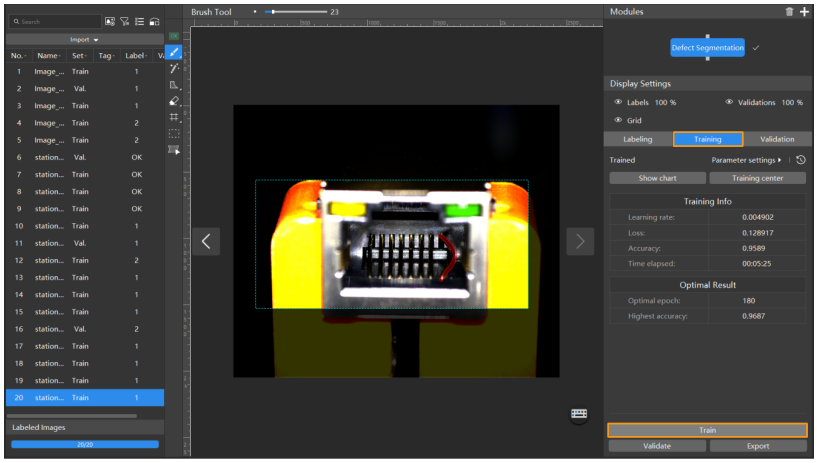
Click to use the eraser tool to remove the labeled region.

For OK images, select an image and click the **OK Label** on the labeling toolbar, or right-click the image and select **[Set to OK]**. The image will be labeled as OK. Please make sure that there is at least one OK image in each dataset.

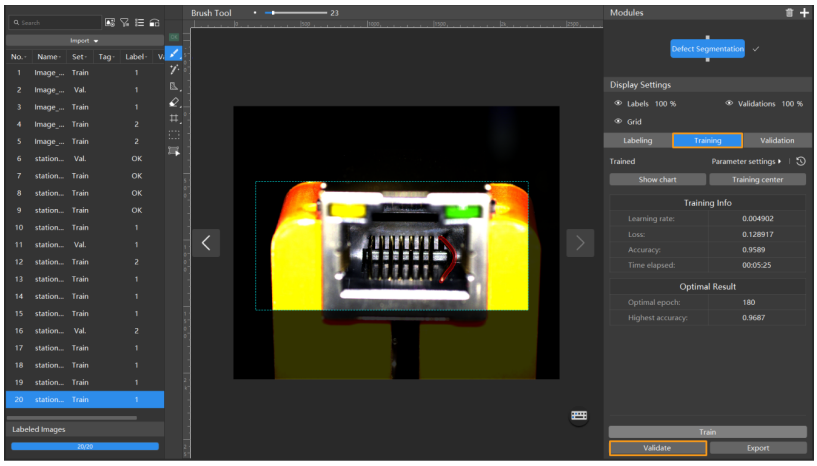


5. **Train the Model**: On the **Training** tab,click **[Train]** in the lower-right corner of the interface to start training the model.

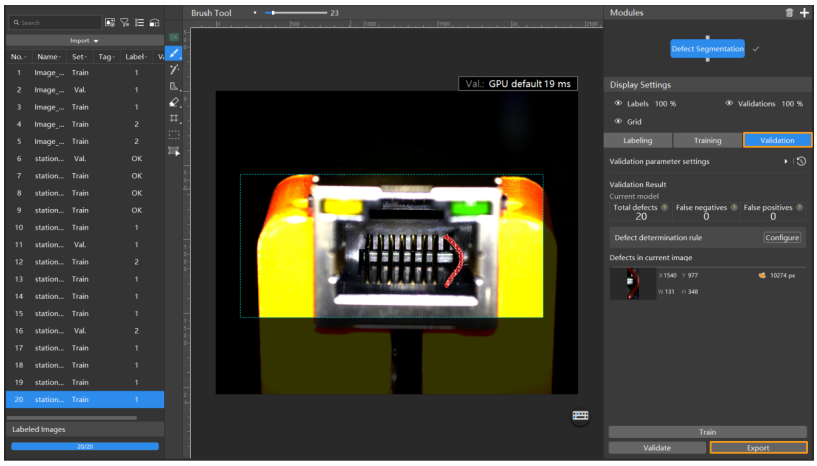
© 2023 Mech-Mind Robotics Technologies Ltd.



6. **Validate the Model**: After the training is completed, click **[Validate]** to validate the model and check the results.



7. **Export the Model**: Click **[Export]** and select a directory to save the exported model (with file extension of .dlkpack).



Now, you have completed the training of the first model. Continue your reading to learn all the details of modules.

© 2023 Mech-Mind Robotics Technologies Ltd.

2. Use Algorithm Modules

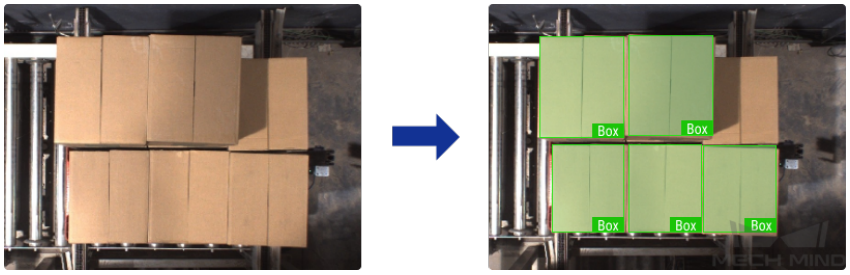
2.1. Train an Instance Segmentation Model

2.1.1. Introduction

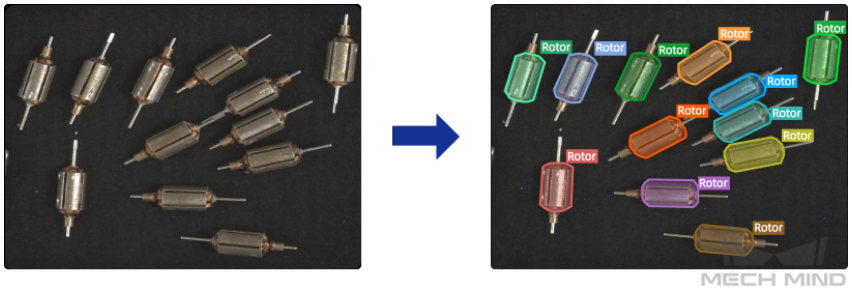
This module is used to segment the contour of target objects and output the corresponding labels of the classes.

Application Scenarios

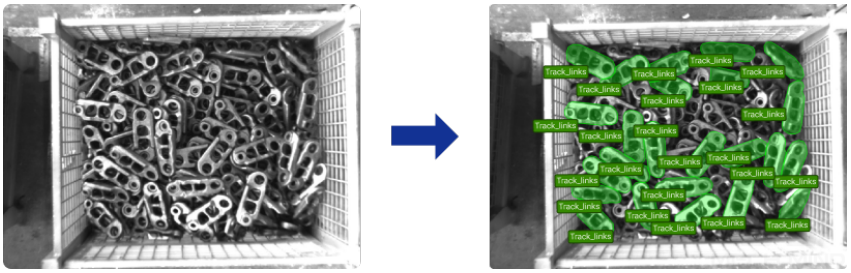
Depalletizing: Applicable to scenarios where cartons, boxes, or sacks need to be depalletized from the pallets and then placed elsewhere (such as a bag break station or conveyor belt).



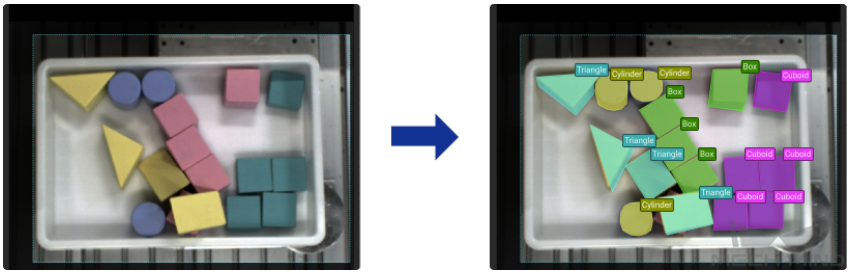
Machine tending: Applicable to machine tending of complex workobjects, structural parts, and irregular parts in automobile, steel, machinery, and other industries.



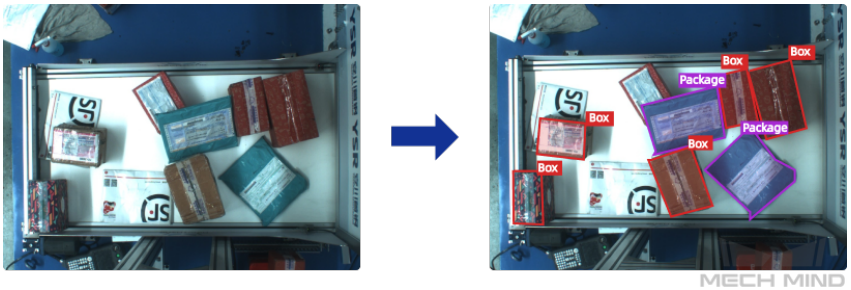
© 2023 Mech-Mind Robotics Technologies Ltd.



Product picking: Applicable to batch picking, discrete order picking, etc., in the e-commerce industry. This module is capable of processing various product packaging such as airbags, transparent packaging, aluminum cans, packaging of irregular shapes, etc.

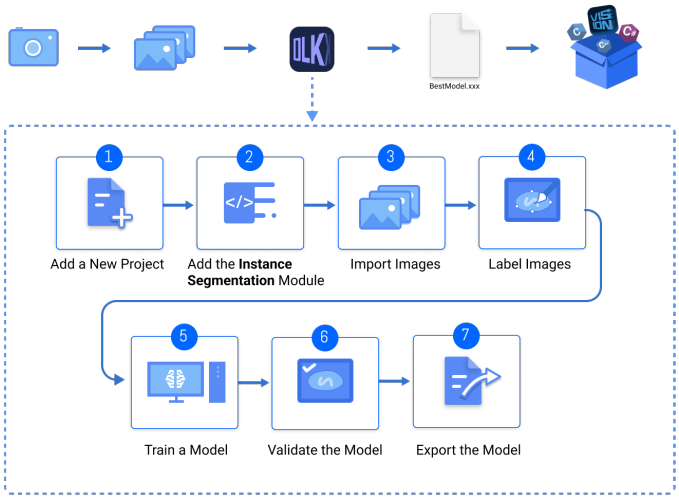


Express parcels: Suitable for detecting parcels of different shapes such as soft packages, postal envelopes, express cartons, padded envelopes, etc.



General Workflow

© 2023 Mech-Mind Robotics Technologies Ltd.

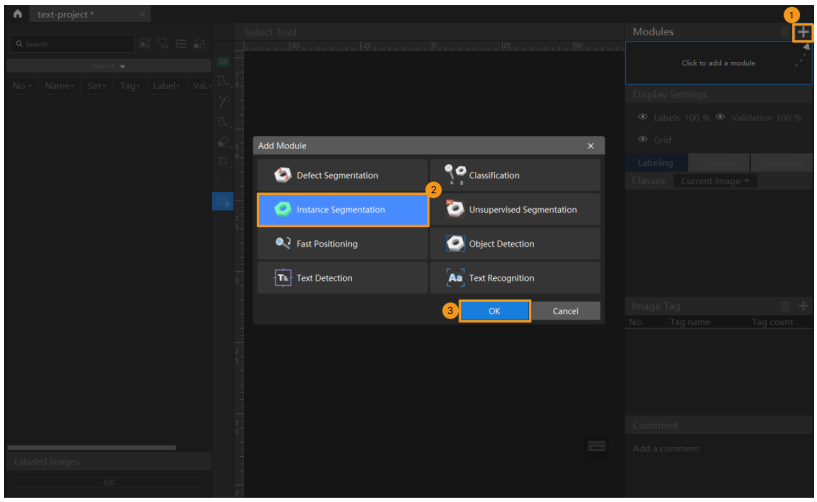


2.1.2. Use the Instance Segmentation Module

Please click here to download an image dataset of wooden blocks, an example project provided by Mech-DLK. In this section, we will use an Instance Segmentation module and train a model to segment different types of wooden blocks and export the corresponding class labels.

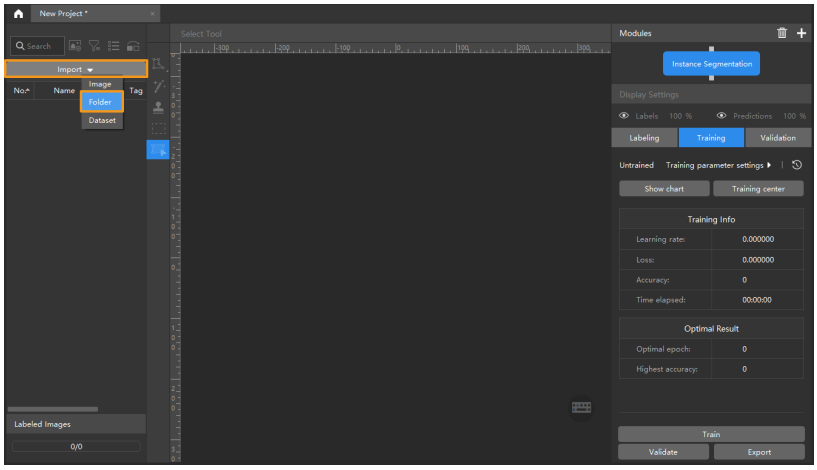
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Instance Segmentation module**: Click **[New Project]** in the interface, name the project, and select a directory to save the project. Click + in the upper right corner of the Modules panel and add the Instance Segmentation module.

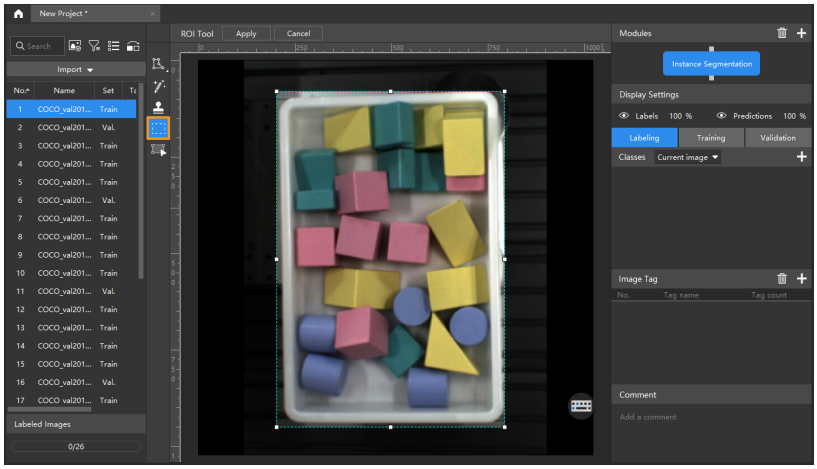


2. **Import the image dataset of wooden blocks**: Unzip the downloaded dataset file. Click the **[Import]** button in the upper left corner, select Folder, and import the image dataset. The wooden blocks in the images are of four different shapes and colors.

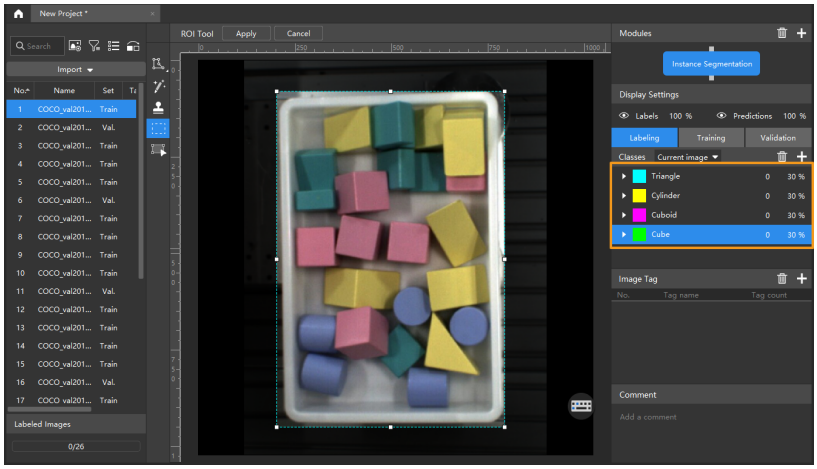
© 2023 Mech-Mind Robotics Technologies Ltd.



3. **Select an ROI**: Click the ROI Tool button and adjust the frame to select the bin containing wooden blocks in the image as an ROI, and click **[Apply]** to save the settings. Setting the ROI can avoid interferences from the background and reduce processing time.



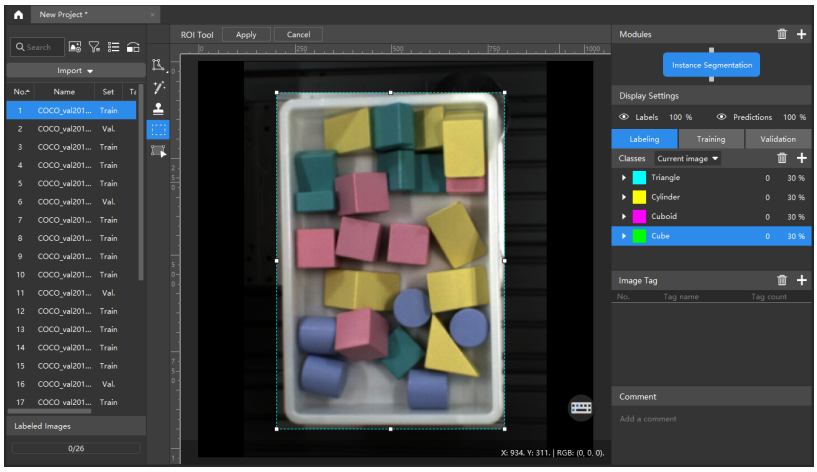
4. **Create Labels**: Select Labeling and click the + button in the Classes panel to create labels based on the type or feature of different objects. In this example, the labels are named after the different shapes of the wooden blocks. You can also name the labels according to different colors.



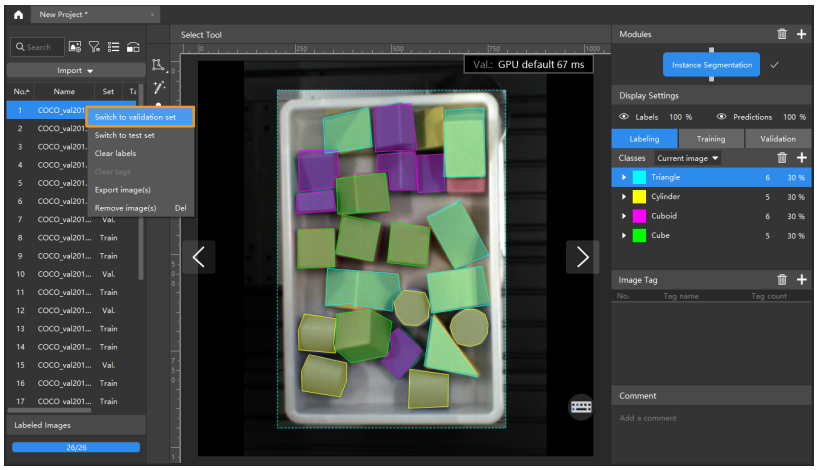
5. **Label images**: Right-click the button and select a suitable tool to label the image. In this example project, the contours of the wooden blocks need to be outlined for segmentation. In

© 2023 Mech-Mind Robotics Technologies Ltd.

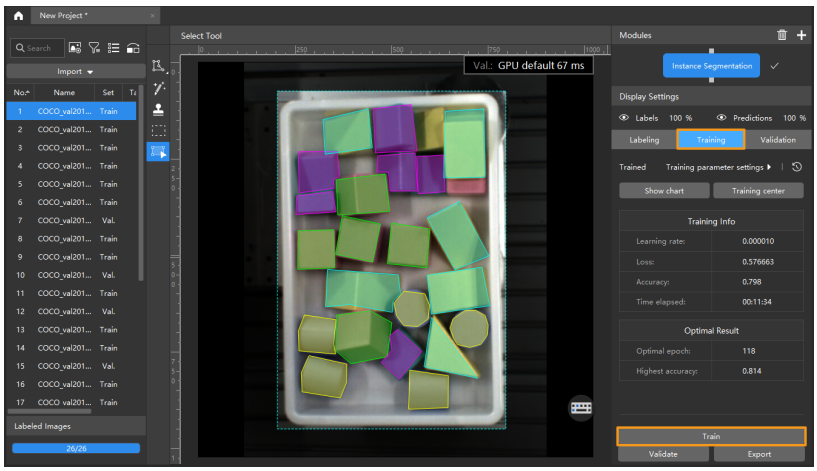
addition, please make sure that the different shapes of wooden blocks have been labeled correctly. Click here to view how to use labeling tools.



6. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. You can click and drag the slider to adjust the proportion. Please make sure that both the training set and validation set include objects of all classes to be segmented. If not, select the image name and then right-click it for adjustment.

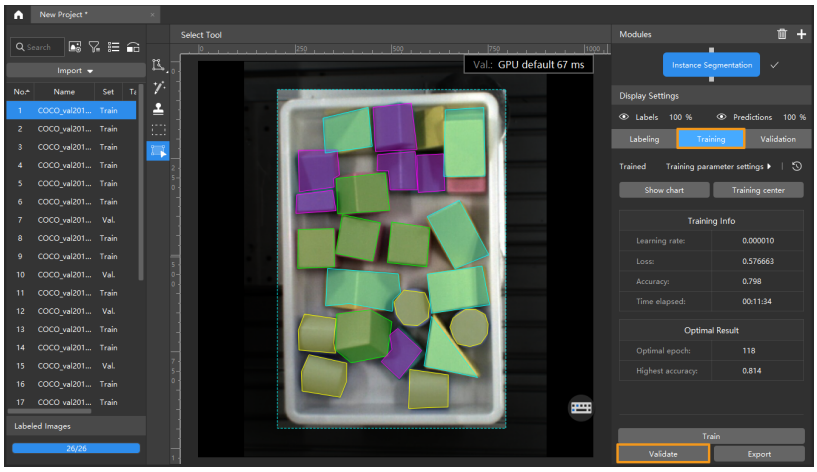


7. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.



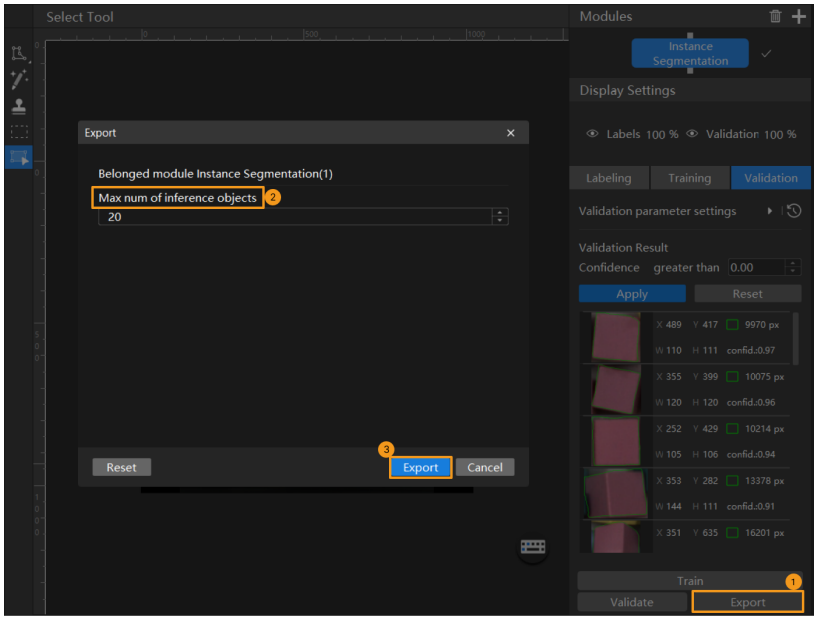
© 2023 Mech-Mind Robotics Technologies Ltd.

8. **Validate the Model**: After the training is completed, click **[Validate]** to validate the model and check the results.



9. **Export the model**: Click **[Export]**. Then, set **Max num of inference objects** in the pop-up dialog box, click **[Export]**, and select a directory to save the exported model.

The maximum number of objects during a round of inference, which is 100 by default.



The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.1.3. Introduction to Labeling Tools

You can use labeling tools to label the images and hence provide data for deep learning training.

You can choose among the following labeling tools built in the software according to actual needs.

Please create labels of corresponding classes according to the needs of the current project before using labeling tools.

© 2023 Mech-Mind Robotics Technologies Ltd.

Polygon Tool

The **Polygon Tool** can draw polygon labels with more vertices, which is suitable for objects of complex shapes.

1. Click (or press P on the keyboard).
2. Click the first position (vertex) in the selection region, then click the second one, third one, etc., to draw the labels, and right-click to finish. (At least three vertices are required.)
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

After labeling, use the **Select Tool** to select a label and adjust the label by the following methods.

- Click the label edges to increase the number of vertices.
- Right-click a vertex to delete it.
- Long press the left mouse button and drag the vertex in any direction to modify the label shape.

Ellipse Tool

Use more vertices to make elliptical selections. This tool is suitable for elliptical objects.

1. Right-click and then click (or press L on the keyboard).
2. Click the first position (vertex) in the selection region, and then continue clicking. An elliptical label should have at least five vertices.
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

After labeling, use the **Select Tool** to select the label and then long press the left mouse button to drag the vertex in any direction and thus modify the label shape.

Rectangle Tool

The **Rectangle Tool** can be used to draw rectangular labels, which is suitable for rectangular objects.

1. Right-click and then click (or press R on the keyboard).
2. Long press the left mouse button in the selection region, move it in any direction, and then release the left mouse button to finish the rectangular selection.
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

Smart Labeling Tool

The **Smart Labeling Tool** can be used to automatically select the objects in the image.

When multiple objects in an image have large color differences and are scattered, you can use the **Smart Labeling Tool** to conveniently label the objects in the image.

1. Click (or press M on the keyboard).
2. Move the cursor in the selection region and then click the object to be labeled.

© 2023 Mech-Mind Robotics Technologies Ltd.

- If the selection cannot completely cover the object, click the uncovered part to expand the selection area.
- If the selection covers the areas outside the object, right-click these areas to reduce the selection area.

3. Click **[Apply]** in the upper-left corner of the selection region.

You can use the **Select Tool** to fine-tune the labeled contour by the following steps:

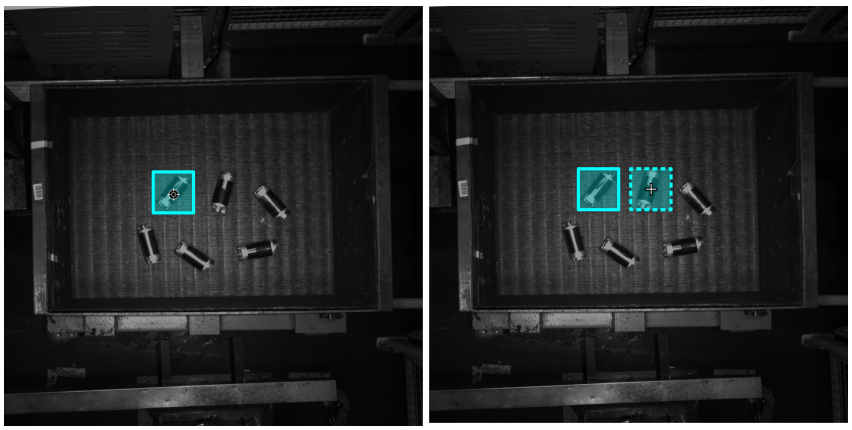
1. Use the **Select Tool** to select the label to be adjusted.
2. Adjust the contour in one of the following three ways according to actual situation. Please ensure that the selected area closely aligns with the object contour.
   a. Place the mouse cursor on a vertex of the contour. When the cursor turns into , long-press the left mouse button and drag the vertex to adjust the contour.
   b. Place the mouse cursor on a vertex of the contour. When the cursor turns into , click the right mouse button to delete the vertex.
   c. Place the mouse cursor on the contour. When the cursor turns into , click the left mouse button to add a vertex.

When the objects differ greatly in colors and have clear contours, it is recommended that you label multiple objects at a time and click **[Apply]**. If the objects are not obviously distinct, it is recommended to label one at a time.

Template Tool

You can use the **Template Tool** to set an existing selection as a template. After setting, you can use this template to rapidly label contours and objects with the same pose.

It is suitable for scenarios where there are multiple neatly arranged objects of the same type in an image, which can improve labeling efficiency.



1. Click (or press C on the keyboard).
2. Click the region that needs to be set as the template.
3. Move the template to the to-be-selected object, adjust the angle of the template to make it fit the object, and then click it.
   - Coarse adjustment: Press and hold the `Shift` key and then scroll the mouse wheel.
   - Fine adjustment: Adjust the **Rotation angle** parameter.

During labeling, press and hold the `Ctrl` key and click the selection to switch the template.

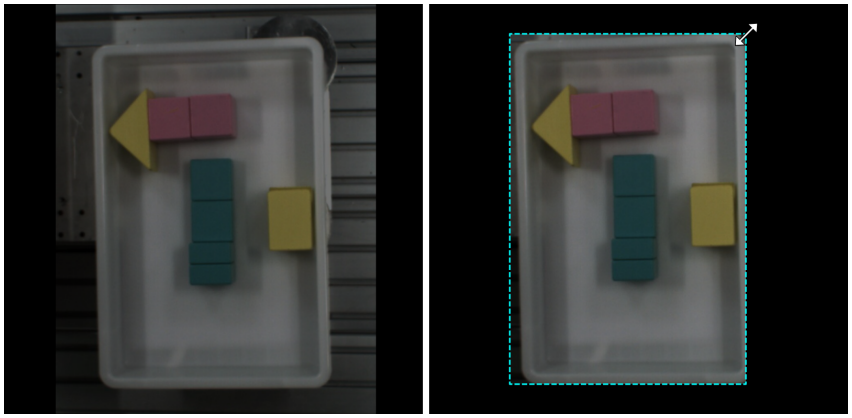
© 2023 Mech-Mind Robotics Technologies Ltd.

> You can also click **[Replace template]** and then click the selection to achieve the same purpose.

ROI Tool

You can use the **ROI Tool** to set the region of interest.

Setting the ROI can avoid interferences from the background.

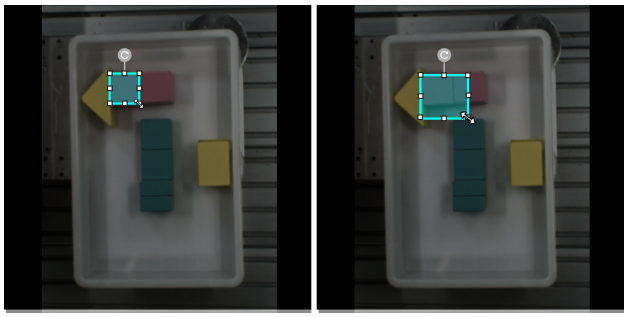
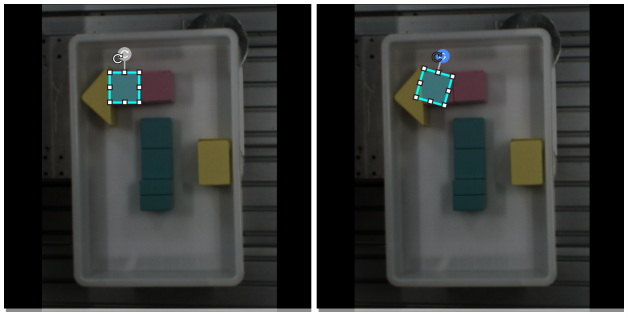


1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.
3. Click **[Apply]** in the upper left corner of the selection region.

Select Tool

You can use the **Select Tool** to select, move, and adjust the selections.

1. Click (or press S on the keyboard).
2. Move the cursor in the selection region and then click the selection to be processed. Select multiple selections by pressing and holding the `Ctrl` key.

Adjust size	
Rotation	

© 2023 Mech-Mind Robotics Technologies Ltd.

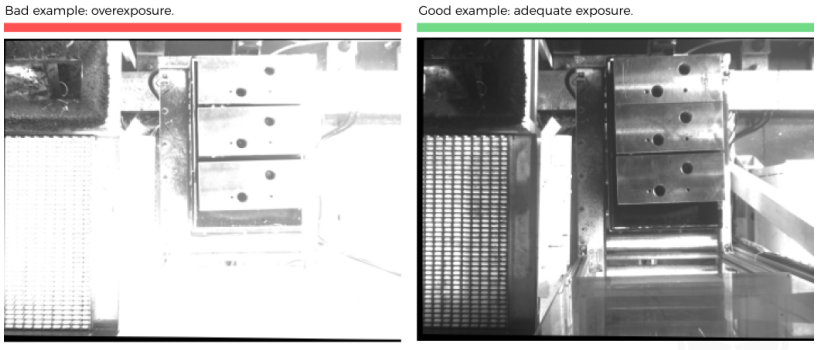
Move	

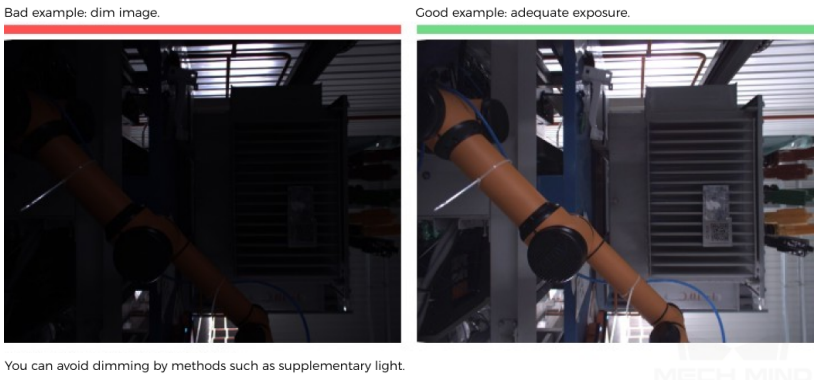
2.1.4. Train a High-Quality Model

This section introduces the factors that most affect the model quality and how to train a high-quality instance segmentation model.

Ensure Image Quality

1. Avoid **overexposure, dimming, color distortion, blur, occlusion**, etc. These conditions can lead to the loss of features that the deep learning model relies on, which will affect the model training effect.

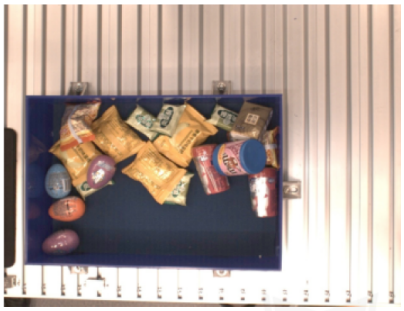




© 2023 Mech-Mind Robotics Technologies Ltd.

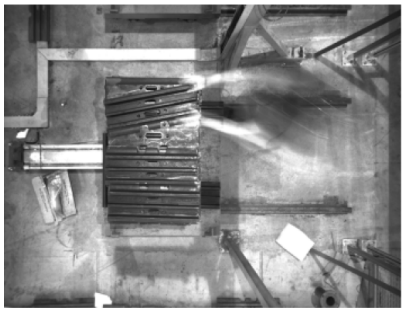
Bad example: color distortion.

Good example: normal color.



Color distortion can be avoided by adjusting the white balance.

Bad example: blur.

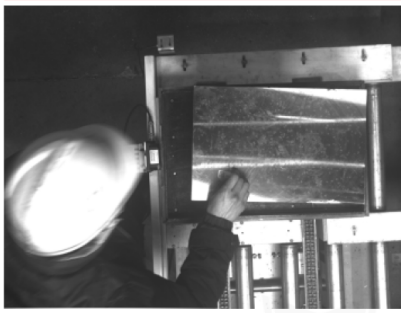


Good example: clear.

Please avoid capturing images when the camera or the objects are still moving.

Bad example: occluded by the robot arm.

Bad example: occluded by a human.



Please make sure there is no robot or human in the way from the camera to the objects.

2. Ensure that the **background, perspective, and height** of the image-capturing process are consistent with the actual application. Any inconsistency can reduce the effect of deep learning in practical applications. In severe cases, data must be re-collected. Please confirm the conditions of the actual application in advance.

Bad example: The background in the training data (left) is different from the background in the actual application (right).

Please make sure the background stays the same when capturing the training data and when deploying the project.

© 2023 Mech-Mind Robotics Technologies Ltd.

Bad example: The field of view and perspective in the training data (left) are different from that in the actual application (right).

Please make sure the field of view and perspective stay the same when capturing the training data and when deploying the project.

Bad example: The camera height in the training data (left) is different from the background in the actual application (right).

Please make sure the camera height stays the same when capturing the training data and when deploying the project.

Ensure Data Quality

The Instance Segmentation module obtains a model by learning the features of existing images and applies what is learned to the actual application. Therefore, to train a high-quality model, the conditions of the collected and selected dataset must be consistent with those of the actual applications.

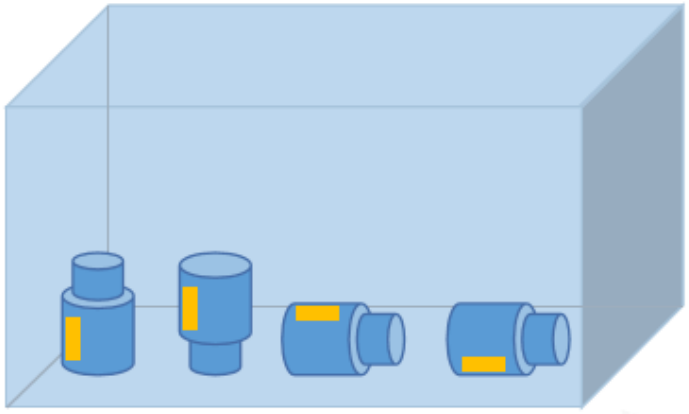
Collect Data

Various placement conditions need to be properly allocated. For example, if there are horizontal and vertical incoming materials in actual production, but only the data of horizontal incoming materials are collected for training, the classification effect of vertical incoming materials cannot be guaranteed. Therefore, when collecting data, it is necessary to **consider various conditions** of the actual application, including the following:

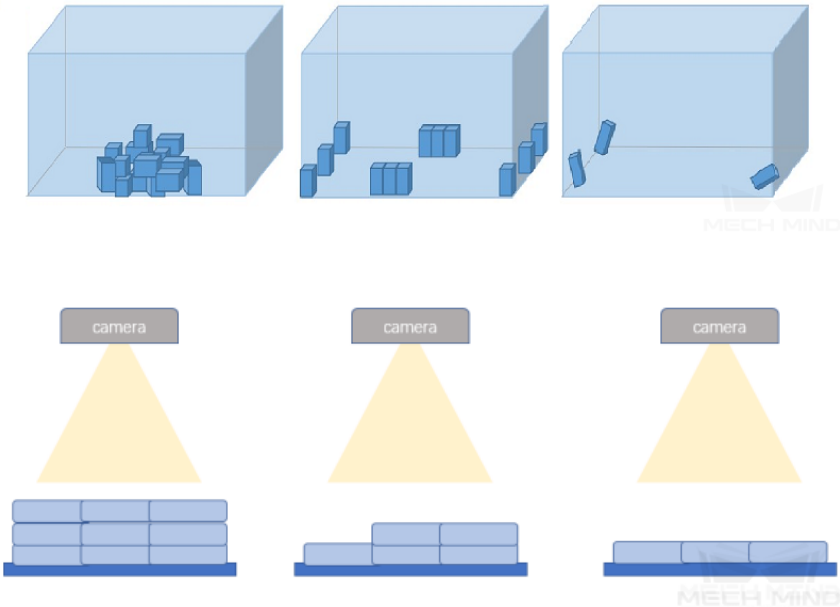
- Ensure that the collected dataset includes all possible **object placement orientations** in actual applications.
- Ensure that the collected dataset includes all possible **object positions** in actual applications.
- Ensure that the collected dataset includes all possible **positional relationships between objects** in actual applications.

If any of the representations among the above three is missed in the dataset, the deep learning model will not be able to learn the features properly and therefore cannot recognize the objects correctly. A dataset with sufficient samples will reduce errors.

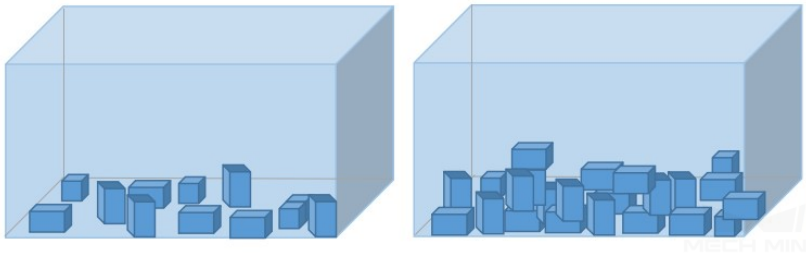
Object placement orientations



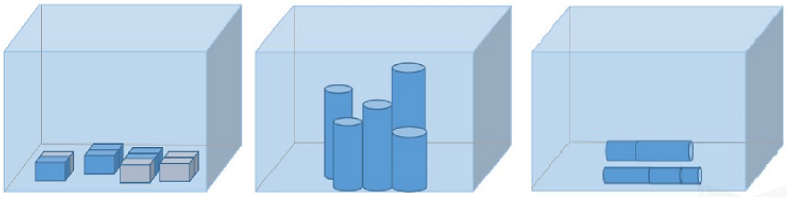
Object positions



Positional relationships between objects

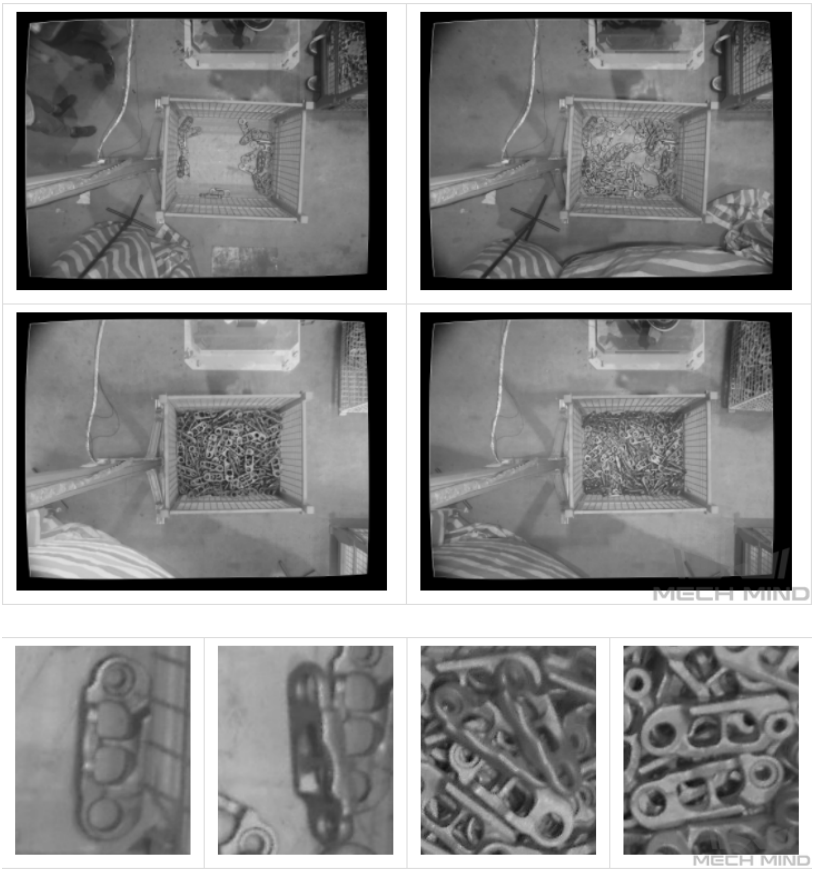


© 2023 Mech-Mind Robotics Technologies Ltd.



Data Collection Examples

1. A metal piece project involves objects of a single class, and thus 50 images were collected. Object placement conditions of lying down and standing on the side need to be considered. Object positions at the bin center, edges, corners, and at different heights need to be considered. Object positional relationships of overlapping and parallel arrangement need to be considered. Samples of the collected images are as follows:



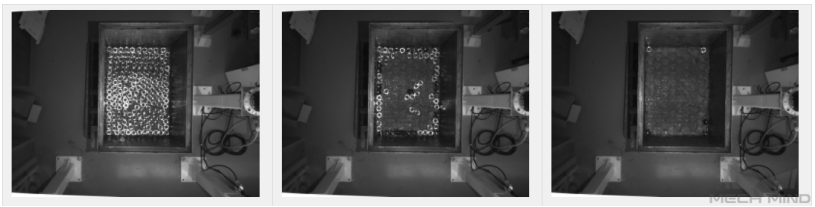
2. A grocery project involves seven classes of mixing objects, which requires classification. The objects of one class placed in different orientations and mixing objects of multiple classes need to be considered to fully capture object features. Number of images for objects of one class = 5 × number of object classes. Number of images for mixing objects of multiple classes = 20 × number of object classes. The objects may come lying flat, standing on sides, or reclining, so images containing all faces of the objects need to be considered. The objects may be in the center, on the edges, and in the corners of the bins. The objects may be placed parallelly or fitted together. Samples of the collected images are as follows:
    - Placed alone

© 2023 Mech-Mind Robotics Technologies Ltd.

- Mixedly placed

3. A track shoe project involves track shoes of many models, and thus the number of images captured was 30 multiplied by the number of models. The track shoes only face up, so only the facing-up condition needs to be considered. They may be on different heights under the camera. In addition, they are arranged regularly together, so the situation of closely fitting together needs to be considered. Samples of the collected images are as follows:

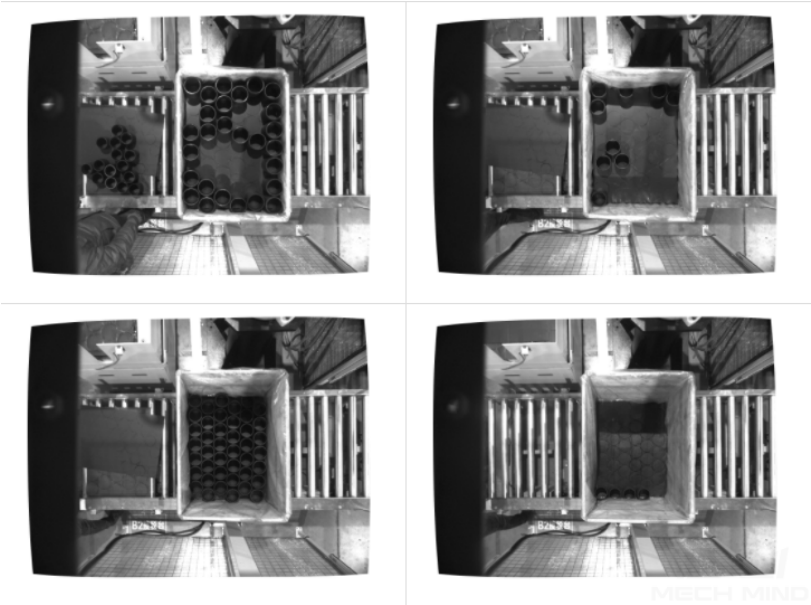
4. A metal piece project involves metal pieces presented in one layer only, and thus, only 50 images were captured. The metal pieces only face up. They are in the center, edges, and corners of the bin. In addition, they may be fitted closely together. Samples of the collected images are as follows:



5. A metal piece project involves metal pieces neatly placed in multiple layers, and thus, 30 images were collected. The metal pieces only face up. They are in the center, edges, and

© 2023 Mech-Mind Robotics Technologies Ltd.

corners of the bin and are at different heights under the camera. In addition, they may be fitted closely together. Samples of the collected images are as follows:



Select Appropriate Data

1. Control the image quantity of datasets

   For the first-time model building of the Instance Segmentation module, capturing 30–50 images is recommended. It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later, and will make the training time longer.

2. Collect representative data

   Image capturing should consider all the conditions in terms of illumination, color, size, etc. of the objects to be recognized.

   - Lighting: Project sites usually have environmental lighting changes, and the data should contain images with different lighting conditions.
   - Color: Objects may come in different colors, and the data should contain images of objects of all the colors.
   - Size: Objects may come in different sizes, and the data should contain images of objects of all existing sizes.

   If the actual on-site objects may be rotated, scaled in images, etc., and the corresponding images cannot be collected, the data can be supplemented by adjusting the data augmentation training parameters to ensure that all on-site conditions are included in the datasets.

3. Balance data proportion

   The number of images of different object classes in the datasets should be proportioned according to the actual project; otherwise, the training effect will be affected. There should be no such case where 20 images are of one object, and only 3 are of the other object.

4. Images should be consistent with the application site

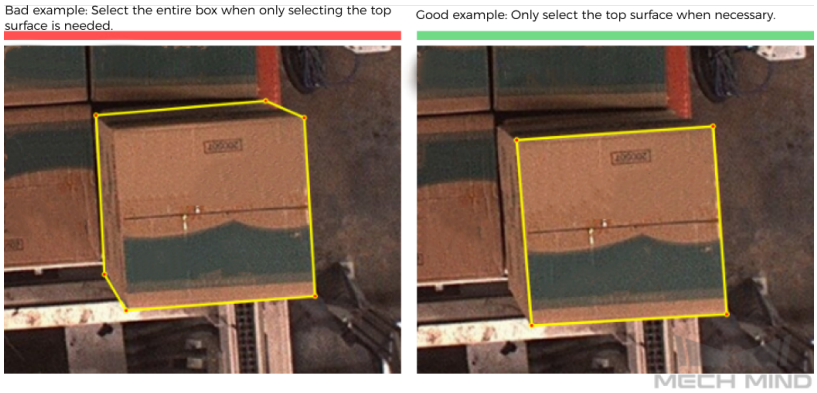
© 2023 Mech-Mind Robotics Technologies Ltd.

The factors that need to be consistent include lighting conditions, object features, background, and field of view.

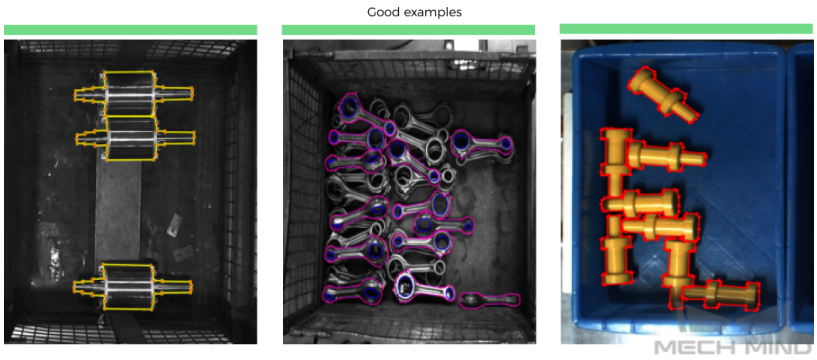
Ensure Labeling Quality

Determine the Labeling Method

1. **Label the upper surface' contour**: It is suitable for regular objects that are laid flat, such as cartons, medicine boxes, rectangular workobjects, etc. For these objects, the pick points are calculated on the upper surface contour, and the user only needs to make rectangular selections on the images.



2. **Label the entire objects' contours**: It is suitable for sacks, various types of workobjects, etc., for which only labeling the object contours is the general method.



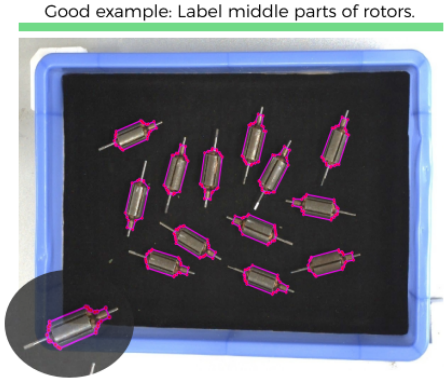
3. **Special cases**: for example, when the recognition result needs to conform to how the grippers work.
    - It is necessary to ensure that the suction cup and the tip of the bottle to pick completely fit (high precision is required), and only the bottle tip contours need to be labeled.

© 2023 Mech-Mind Robotics Technologies Ltd.

Good example: Label bottle tips.

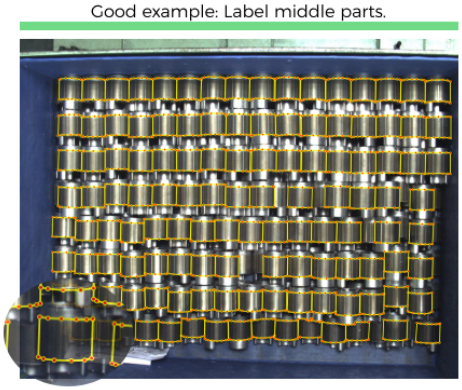
- The task of rotor picking involves recognizing rotor orientations. Only the middle parts whose orientations are clear can be labeled, and the thin rods at both ends cannot be labeled.

Good example: Label middle parts of rotors.



- It is necessary to ensure that the suction parts are in the middle parts of the metal pieces, so only the middle parts of the metal pieces are labeled, and the ends do not need to be labeled.

Good example: Label middle parts.

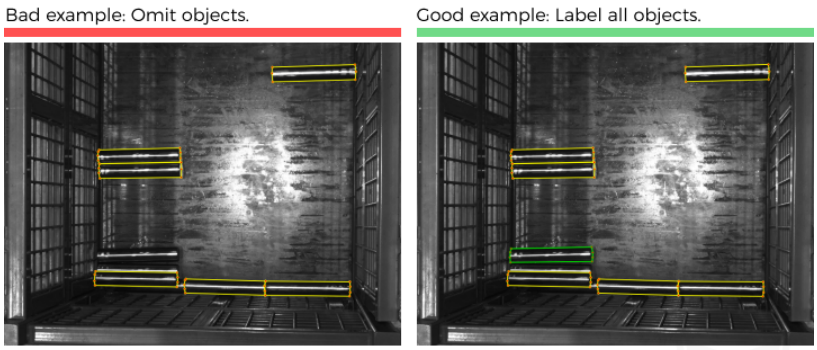


Check Labeling Quality

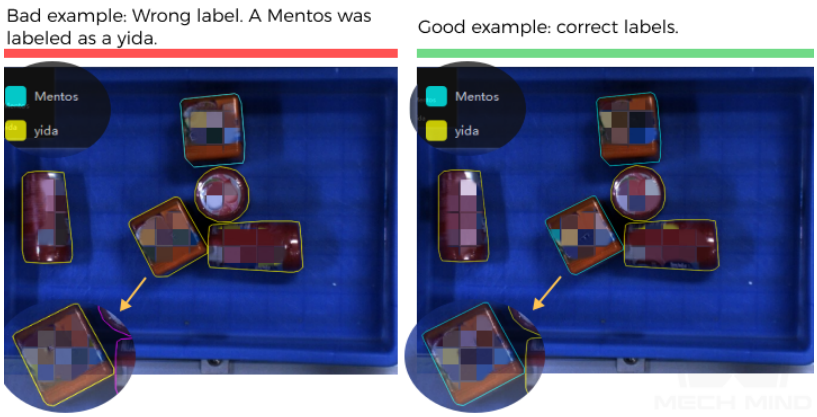
The labeling quality should be ensured in terms of completeness, correctness, consistency, and accuracy:

1. **Completeness**: Label all objects that meet the rules, and avoid missing any objects or object parts.

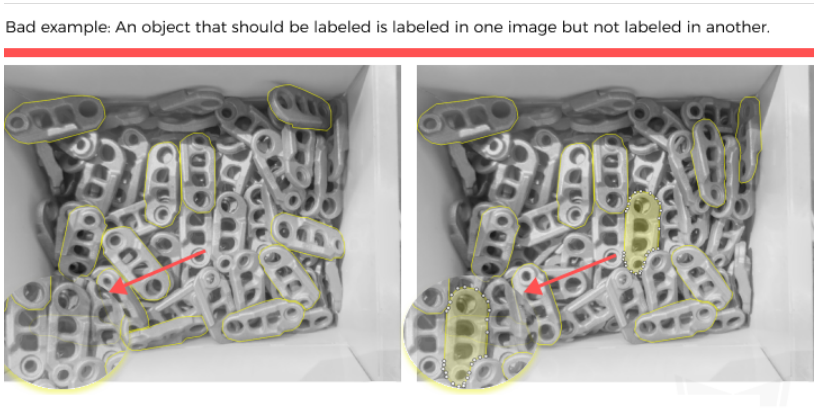
© 2023 Mech-Mind Robotics Technologies Ltd.



2. Correctness: Make sure that each object corresponds correctly to the label it belongs to, and avoid situations where the object does not match the label.

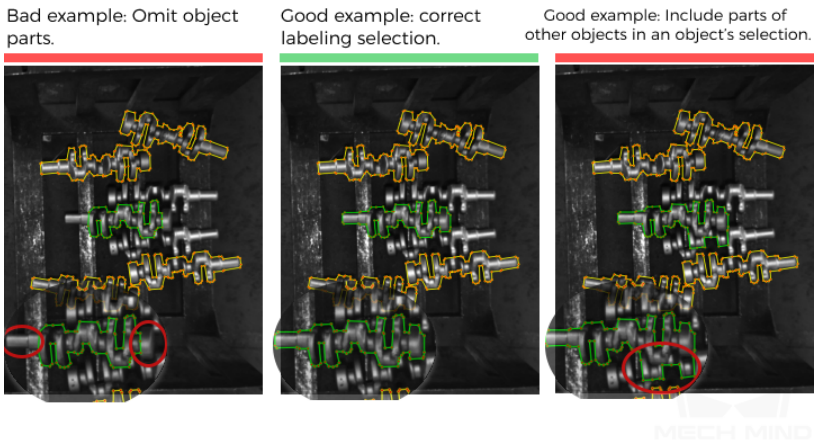


3. Consistency: All data should follow the same labeling rules. For example, if a labeling rule stipulates that only objects that are over 85% exposed in the images be labeled, then all objects that meet the rule should be labeled. Please avoid situations where one object is labeled but another similar object is not.



4. Accuracy: Make the region selection as fine as possible to ensure the selected regions' contours fit the actual object contours and avoid bluntly covering the defects with coarse large selections or omitting object parts.

© 2023 Mech-Mind Robotics Technologies Ltd.



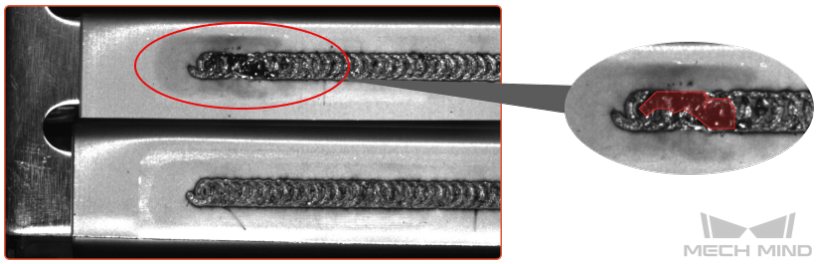
2.2. Train a Defect Segmentation Model

2.2.1. Introduction

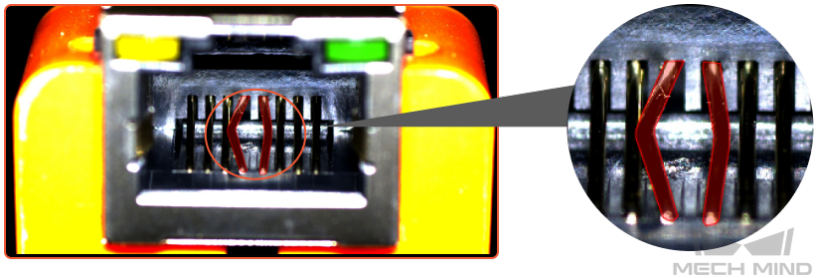
This module is used to detect and segment the defect regions in the image.

Application Scenarios

1. Renewable energy: For detecting various types of defects. This module is capable of working under complicated situations, such as when the defects are tiny, the background is complex, or the positions of the workobjects are not fixed, etc. For example, this module can be used for weld seam inspection or appearance inspection of lithium batteries.



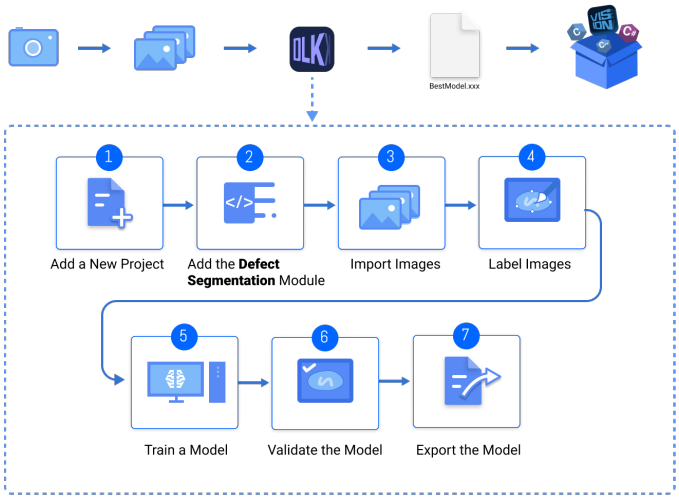
2. Electronics manufacturing: For detecting the surface defects, such as stains, bubbles, scratches, etc., of functional modules and electronic components.



3. PCB manufacturing, printing, daily necessities manufacturing, and other industries: For detecting surface defects, such as scratches or foreign object debris, of PCB, connectors, printing products, daily necessities, and other objects.

© 2023 Mech-Mind Robotics Technologies Ltd.

General Workflow

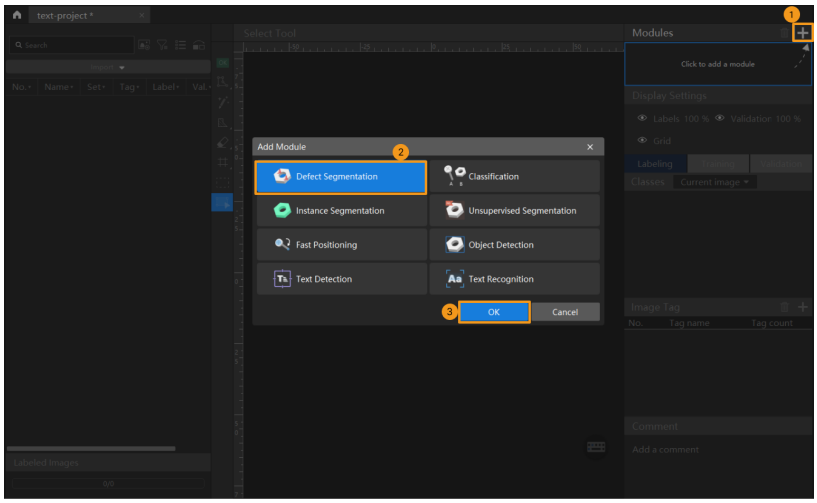


2.2.2. Use the Defect Segmentation Module

Please click here to download an image dataset of network ports, an example provided by Mech-DLK. Different from the content in Train Your First Model, in this section, we will use a Defect Segmentation module and train a model to detect defects such as deformations and fractures on the network ports.

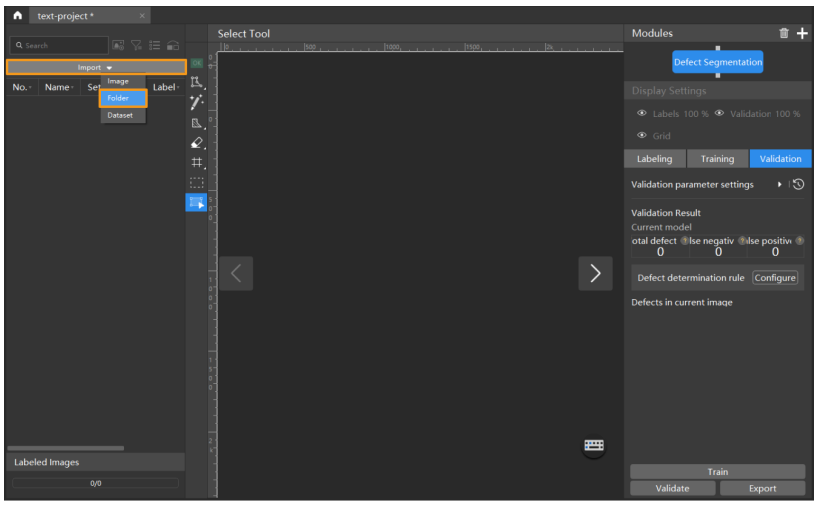
You can also use your own data. The usage process is overall the same, and the labeling part is different.

1. **Create a new project and add the Defect Segmentation module**: Click **[New Project]** in the interface, name the project, and select a directory to save the project. Click **+** in the upper right corner of the Modules panel and add the Defect Segmentation module.

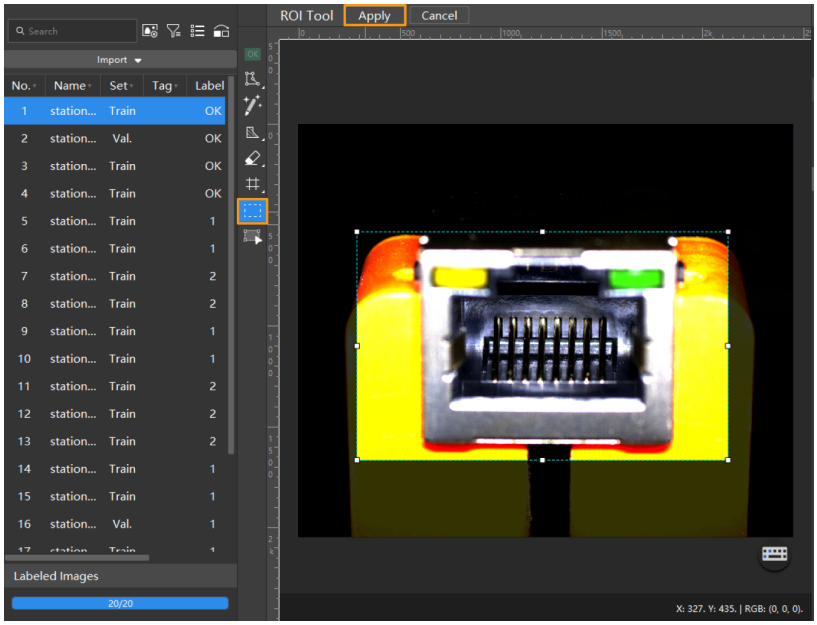


© 2023 Mech-Mind Robotics Technologies Ltd.

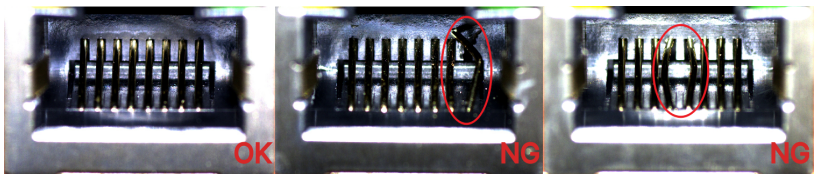
2. **Import the image dataset of network ports**: Unzip the downloaded dataset file. Click the **[Import]** button in the upper left corner, select **[Folder]**, and import the image dataset. The pins in the images can be deformed, fractured, or intact.



3. **Select an ROI**: Click the ROI Tool button and adjust the frame to select the pins in the image as an ROI, and click **[Apply]** to save the settings. Setting the ROI can avoid interferences from the background and reduce processing time. Setting the ROI can avoid interferences from the background.



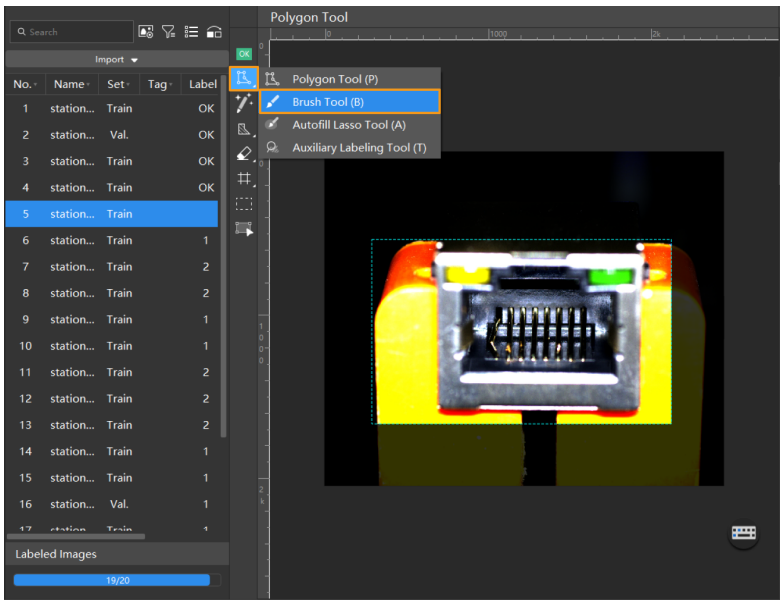
4. **Labeling**: In this example, you will need to label the OK images and NG images in each dataset. OK means that the connectors meet quality requirements and NG means that there are defects such as deformations and fractures on the Ethernet ports.



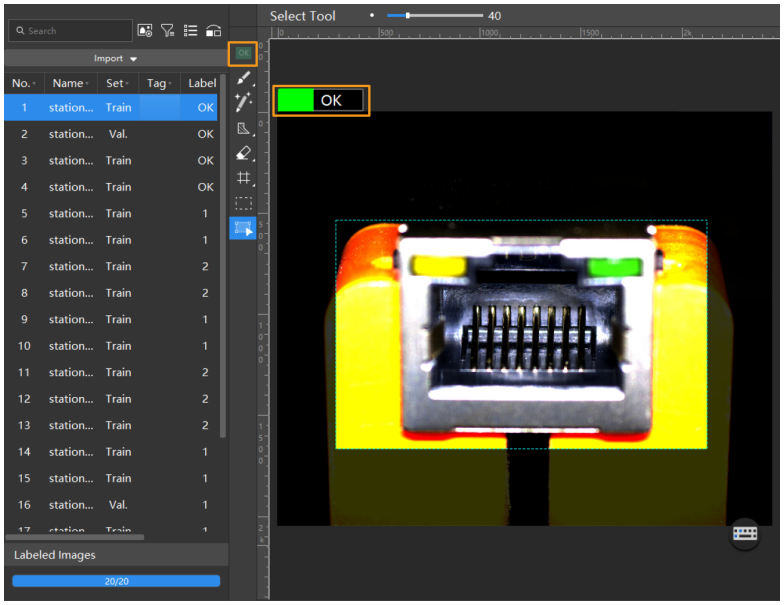
   - For NG images, right-click the Polygon Tool and select a suitable tool to select the defect

© 2023 Mech-Mind Robotics Technologies Ltd.

regions. In this example, it is recommended to use the Brush Tool. During labeling, the brush should be close to the edge of the defect to avoid the inclusion of a large number of non-defective areas. Click here to view how to use labeling tools.

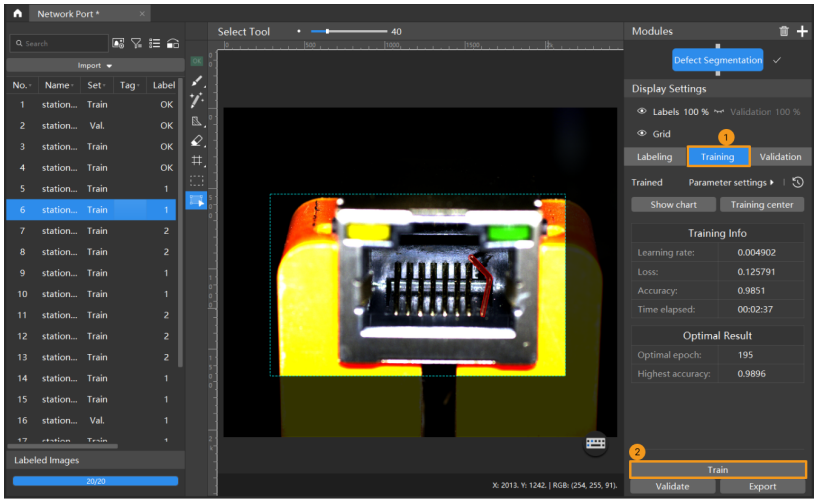


- For OK images, select an image and click the **OK Label** OK on the labeling toolbar. The image will be labeled as OK.

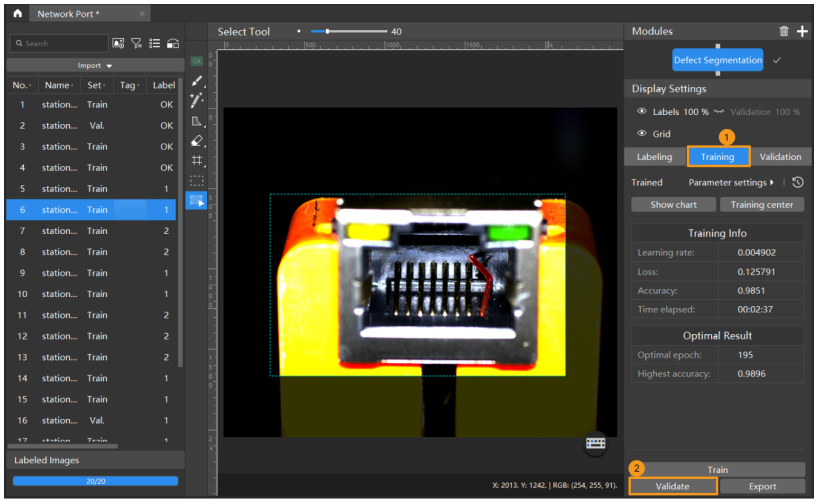


5. **Split the dataset into the training set and validation set**: Please make sure that both the training set and validation set include images with all types of defects and at least one OK image. This can guarantee that the algorithm can learn all different types of defects and validate the images with different defects properly. If the default training set and validation set cannot meet this requirement, please right-click the individual image and switch it to the training/validation set manually.
6. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.

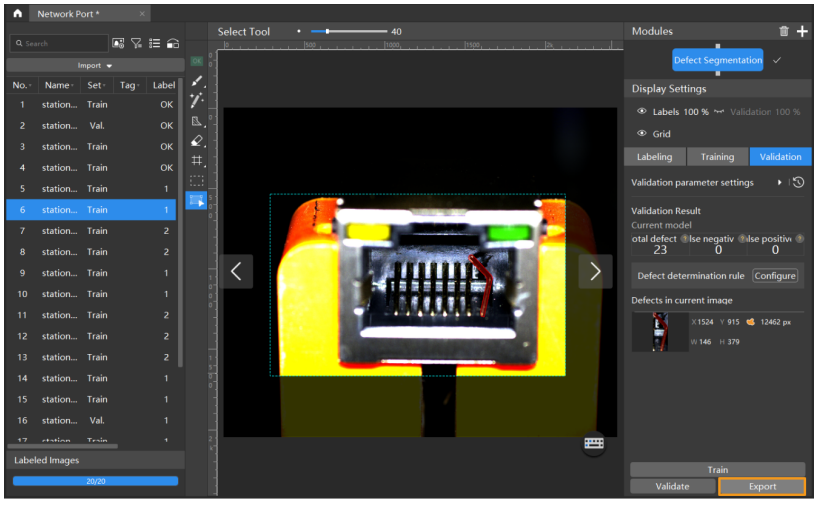
© 2023 Mech-Mind Robotics Technologies Ltd.



7. **Validate the Model**: After the training is completed, click **[Validate]** to validate the model and check the results. You can also modify the Defect determination rule to filter results.



8. **Export the model**: Click **[Export]** and select a directory to save the trained model.



The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

© 2023 Mech-Mind Robotics Technologies Ltd.

2.2.3. Introduction to Labeling Tools

OK Label

Use the **OK Label** to label an image as OK.

- Select an image from the image list and then click the **OK Label** on the toolbar. The image will be labeled as OK.

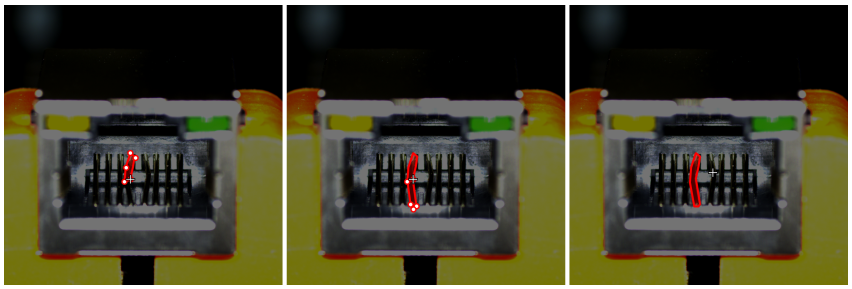
Defect Labeling Tools

Defect labeling tools can label the defects in images to provide the information required by deep learning training.

You can choose among the following labeling tools built in the software according to actual needs.

Polygon Tool

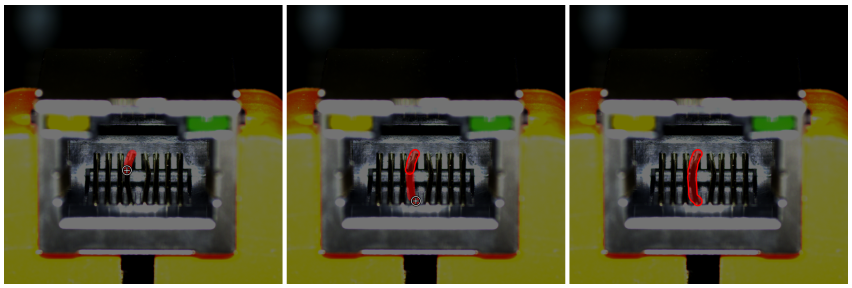
The **Polygon Tool** can draw polygon labels with more vertices, which is suitable for regular defects.



1. Click (or press P on the keyboard).
2. Click the first position (vertex) in the selection area, then click the second one, third one, etc., to draw the labels, and right-click to finish.

Brush Tool

Use the **Brush Tool** to draw labels of any shape. This tool is suitable for defects with complex shapes.

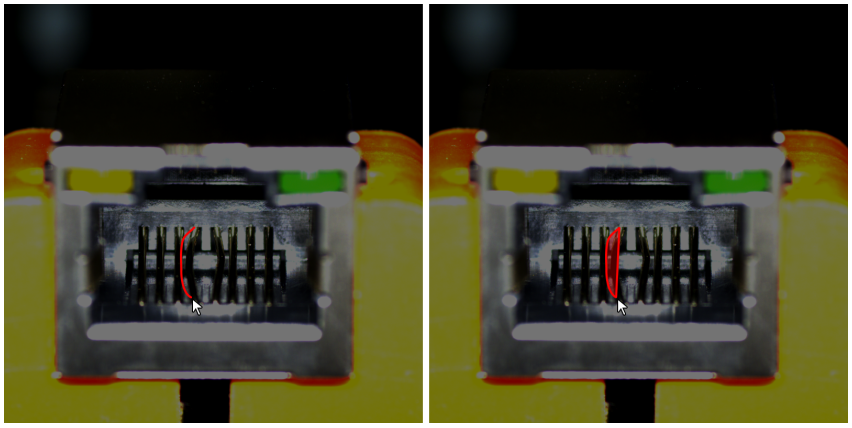


1. Right-click and then click (or press B on the keyboard).
2. Adjust the slider to set the thickness of lines according to the size of defects.
3. Long press the left mouse button in the selection area, move in any direction, and then release the left mouse button to finish the drawing.

© 2023 Mech-Mind Robotics Technologies Ltd.

Autofill Lasso Tool

Draw arbitrarily shaped labels by forming closed shapes with brush paths. This tool is used for defects with complex shapes.



1. Right-click and then click (or press A on the keyboard).
2. Adjust the slider to set the thickness of lines according to the size of defects.
3. Long press the left mouse button in the selection area and move in any direction to form closed shapes with the starting point and thus finish the drawing.

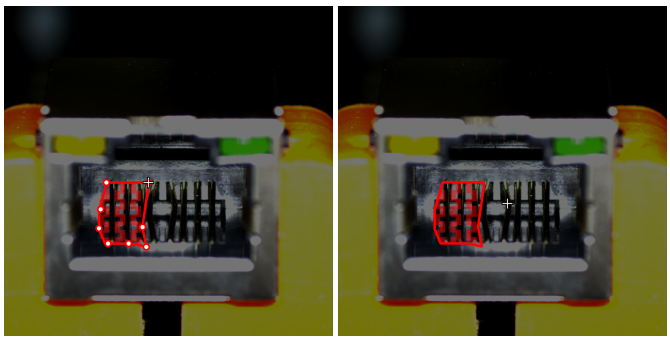
Mask Tool

If there are some irrelevant parts that may interfere with model training/inference, you can use the **Mask Tool** to cover such parts. The masked parts will not be involved in training/inference. For example, the object surface features that should not be judged as defects but are similar to defects, need to be masked out.

You can choose among the following three mask tools built in the software according to actual needs.

- The **Mask Type** includes **Mask single image** and **Mask globally**.
  - Mask single image: the mask is only valid in the current image.
  - Mask globally: the mask is valid in all images.
- It is recommended that the same type of masks should be filled with the same color.

Mask Polygon Tool

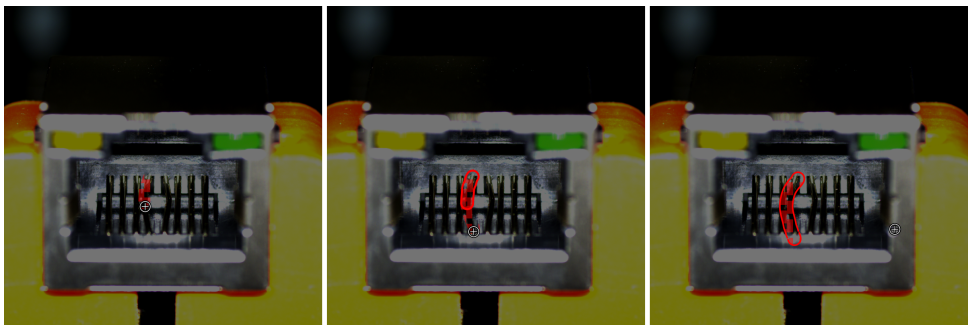


1. Click (or press Shift + P on the keyboard)

© 2023 Mech-Mind Robotics Technologies Ltd.

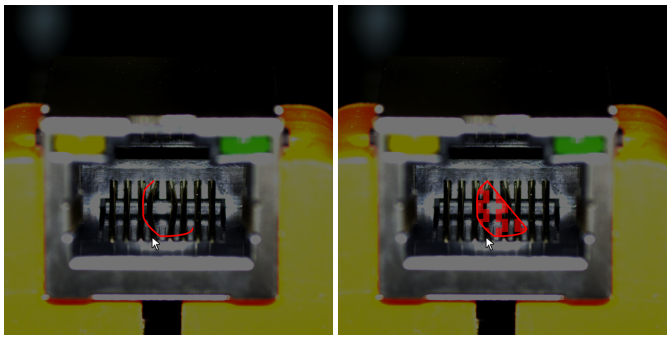
2. Set **Mask Type** and **Mask fill**.
3. Click the first position (vertex) in the selection area, then click the second one, third one, etc., to draw the labels, and right-click to finish.

Mask Brush Tool



1. Right-click and then click (or press Shift + B on the keyboard)
2. Set **Mask Type**, **Mask fill**, and **Brush size**.
3. Long press the left mouse button in the selection area, move in any direction, and then release the left mouse button to finish the drawing.

Mask Lasso Tool

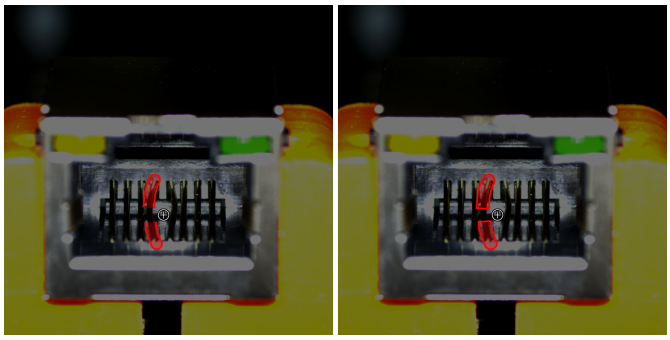


1. Right-click and then click (or press Shift + A on the keyboard)
2. Set **Mask Type** and **Mask fill**.
3. Long press the left mouse button in the selection area and move in any direction to form closed shapes with the starting point.

Labeling Eraser Tool

The **Labeling Eraser Tool** can be used to erase the labeled region.

© 2023 Mech-Mind Robotics Technologies Ltd.

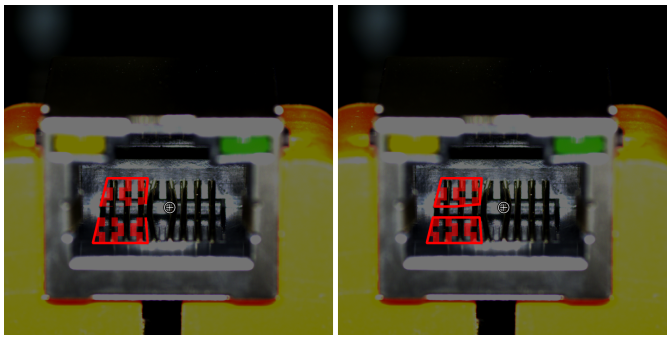


1. Click (or press E on the keyboard)
2. Long press the left mouse button in the selection area and move in any direction.

> Adjust the slider to change the eraser size.

Mask Eraser Tool

The **Mask Eraser Tool** can be used to erase the masks.



1. Right-click and then click (or press Shift + E on the keyboard)
2. Long press the left mouse button in the selection area and move in any direction.

> Adjust the slider to change the eraser size.

Grid Cutting Tool

In industrial inspection scenarios, if the size of images captured by the camera is large, smaller defects may be inconspicuous. If training in such a case is performed, defects are difficult to detect. You can use the **Grid Cutting Tool** to cut the large images into cell images of the same size according to the set dimension. Defect labeling should be completed for all images before the application of this tool.

1. Click (or press U on the keyboard).
2. Set **Rows** and **Columns** and then click **[Apply]**.
   - Place the cursor in the parameter boxes and then scroll the mouse wheel.
   - Enter values in the parameter boxes.

> Note that the number of rows and columns should not be too large, or else the number of cell images after cutting is great, which slows down subsequent inference.

© 2023 Mech-Mind Robotics Technologies Ltd.

Grid Selection Tool

Right-click and then click (or press I on the keyboard) to open the Grid Selection Tool. By default, the cell images with defect labels are checked, which will be added into the training/validation set. You can select cell images with and without defects on demand. Click the **Preview** button in the upper right corner of the selected image to preview the cell images.

- Select defects: Select all the cell images containing defects to put into the training/validation set.
- Select all: Select all the produced cell images into the training/validation set, and set those containing no defects to OK.
- Clear selection: Clear the selections on cell images.

2.2.4. Train a High-Quality Model

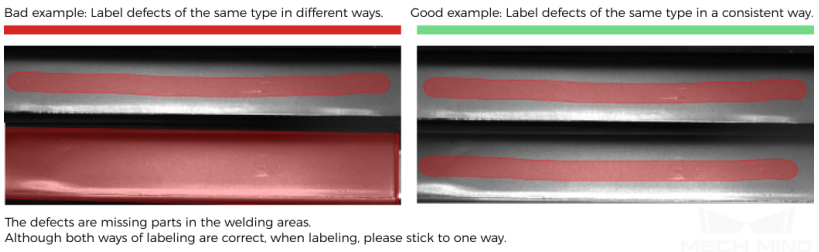
Industrial quality inspections usually have strict limits on false negative and false positive rates. Therefore, the quality of a defect segmentation model is very important. This section introduces the factors that most affect the model quality and how to train a high-quality defect segmentation model.

Ensure Labeling Quality

Labeling quality is one of the most significant factors affecting model performance. In actual projects, low labeling quality accounts for the reasons for more than 90% of poor model performance cases. Therefore, if the model is not performing well, solving labeling quality issues should be prioritized.

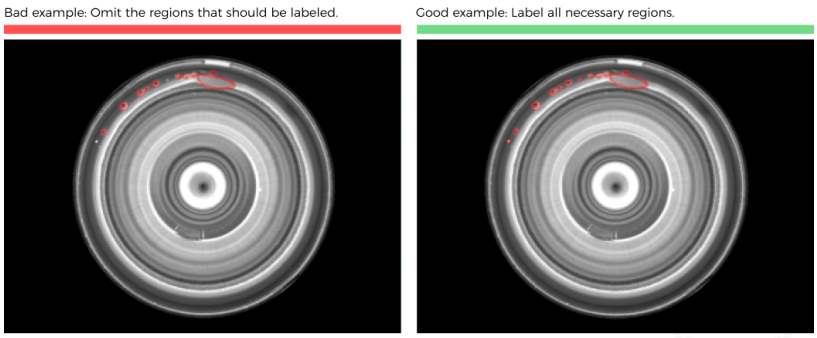
Labeling quality involves consistency, completeness, accuracy, and certainty:

1. **Consistency**: Ensure the consistency of defect labeling methods, and avoid using different labeling methods for the same type of defects.

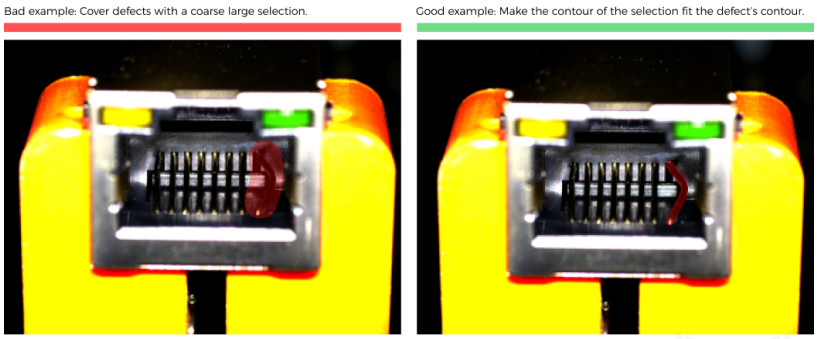


2. **Completeness**: Ensure that all regions that should be considered as defect regions according to the user-defined standard are selected, and avoid any missed selections.

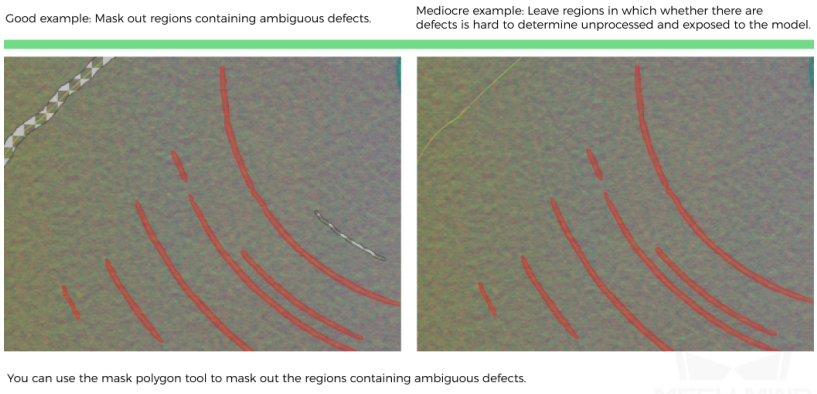
© 2023 Mech-Mind Robotics Technologies Ltd.



3. **Accuracy**: Make the region selection as fine as possible to ensure the selected regions' contours fit the actual defects' contours and avoid bluntly covering the defects with coarse large selections.



4. **Certainty**: For ambiguous defects, when it is impossible to judge whether the defect judgment criteria are met, the mask polygon tool can be used to cover the defect regions.

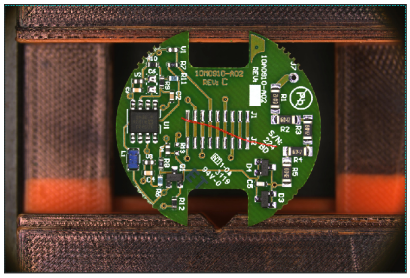


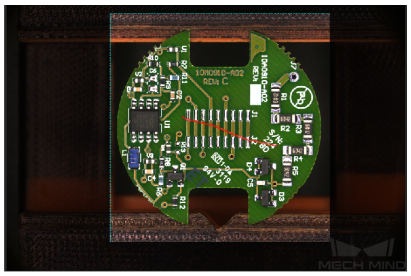
When there are multiple defects in the image, if it is impossible to judge whether each defect meets the defect judgment criteria, you can delete the current image to avoid affecting the model training effect.

Set the Proper Region of Interest (ROI)

Setting the ROI can effectively eliminate the interference of the background, and the ROI boundary should be as close to the outer contours of the objects as possible.

© 2023 Mech-Mind Robotics Technologies Ltd.





The same ROI setting will be applied to all images, so it is necessary to ensure that objects in all images are located within the ROI, especially in scenarios where the object positions/sizes are not fixed.

Select the Right Dataset

- **Control dataset image quantities** For the first-time model building of the Defect Segmentation module, capturing 20 to 30 images is recommended. It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later, and will make the training time longer.
- **Collect representative data** The datasets should contain NG images covering all the defect types with all defect features, in terms of shape, background, color, size, etc. When the features in OK images do not differ across images, the number of OK images can be relatively small.
- **Balance data proportion** The number of images of different conditions/object classes in the datasets should be proportioned according to the actual project; otherwise, the training effect will be affected.
- **Images should be consistent with the application site** The factors that need to be consistent include lighting conditions, object features, background, field of view, etc.

2.2.5. Configure Defect Determination Rules

Configure Logical Rules

A logical rule can contain one or more defect filter items, and the logical relationship between these items is “or”. When a candidate defect matches any of the defect filter items, it matches the logical rule. The software allows the configuration of multiple logical rules, and the logical relationship between them is “and”. If multiple filtering rules are configured, the candidate defect will be determined as a defect only when it matches all the logical rules.

If any general rules are configured, the candidate defect also needs to match the general rules to be finally determined as a defect.

Defect Filter Items

- Defect area filter

  A single candidate defect will be determined as a defect when its area falls within the set area range(s).

- Defect aspect ratio filter

  Defect aspect ratio refers to the ratio of the short side to the long side of the minimum circumscribed rectangle of the defect. A single candidate defect will be determined as a defect when its aspect ratio falls within the set aspect ratio range(s).

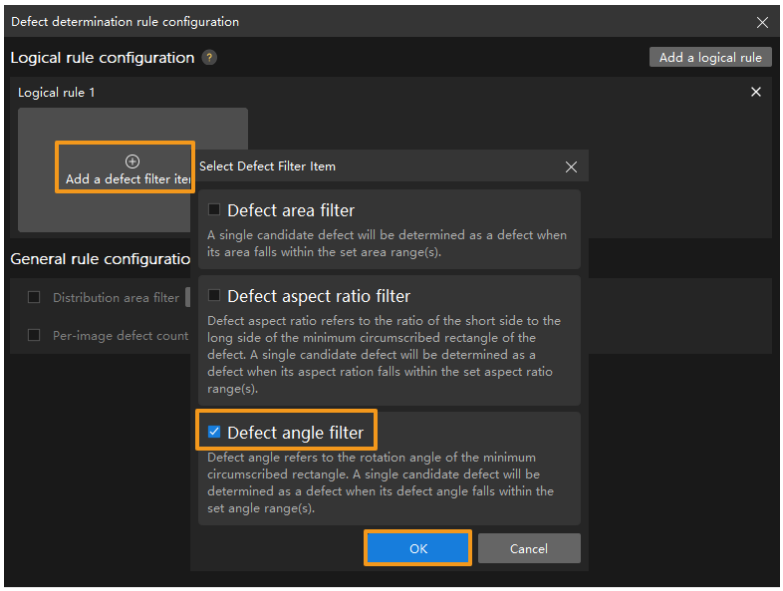
© 2023 Mech-Mind Robotics Technologies Ltd.

- Defect angle filter

  Defect angle refers to the rotation angle of the minimum circumscribed rectangle. A single candidate defect will be determined as a defect when its defect angle falls within the set defect angle range(s).

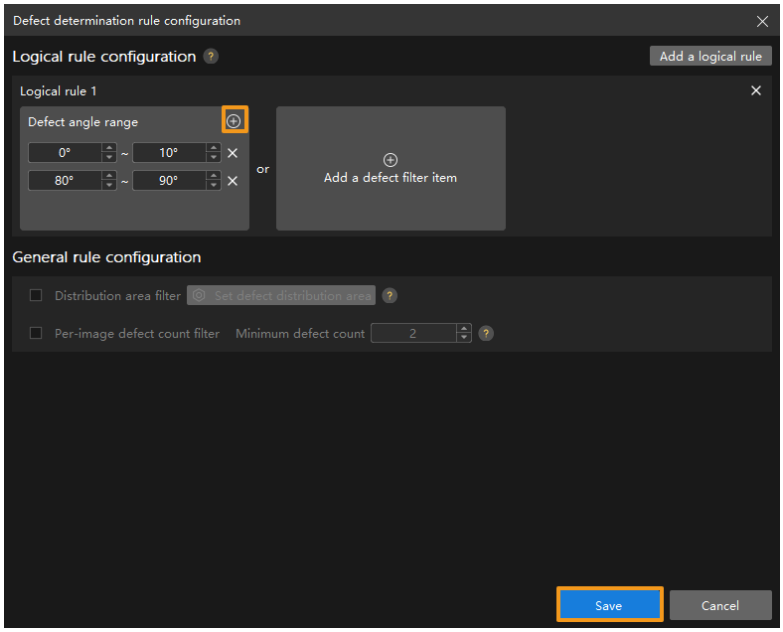
Application Examples

- Filter the defects with angles ranging from 0° to 10° and 80° to 90°.
  1. Click [Add a defect filter item].
  2. Check the Defect angle filter.
  3. Click [OK].

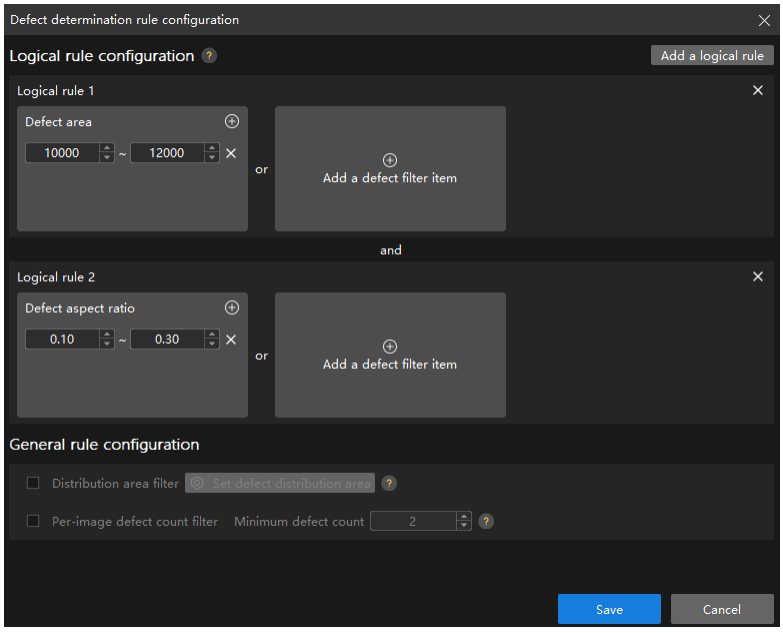


  4. Click [+] and set the angle ranges separately.
  5. Click [Save].

© 2023 Mech-Mind Robotics Technologies Ltd.



- Filter defects with an area ranging from 10000 to 12000 and an aspect ratio ranging from 0.1 to 0.3.
  1. Click **[Add a logical rule]**.
  2. Click **[Add a defect filter item]** under Logical rule 1.
  3. Check the Defect area filter and set the area range.
  4. Click **[Add a defect filter item]** under Logical rule 2.
  5. Check Defect aspect ratio filter and set the aspect ratio range.



General Rule Configuration

Filter Distribution Area

The part(s) of a single candidate defect which intersects with the set distribution areas will be

© 2023 Mech-Mind Robotics Technologies Ltd.

determined as defect(s).

1. Check this parameter and then click **[Set defect distribution area]**.
2. Draw the defect distribution area(s) in the pop-up window.
3. Click **[OK]**.

Filter Per-Image Defect Count

Candidate defects on a single image will be determined as defects when the total number of defects on the image is equal to or larger than the set Minimum defect count parameter.

Check this parameter and then set the Minimum defect count.

After the configuration, click **[Save]** and the configuration takes effect. Validation results change with the configuration, and relevant configurations will take effect in the exported model.

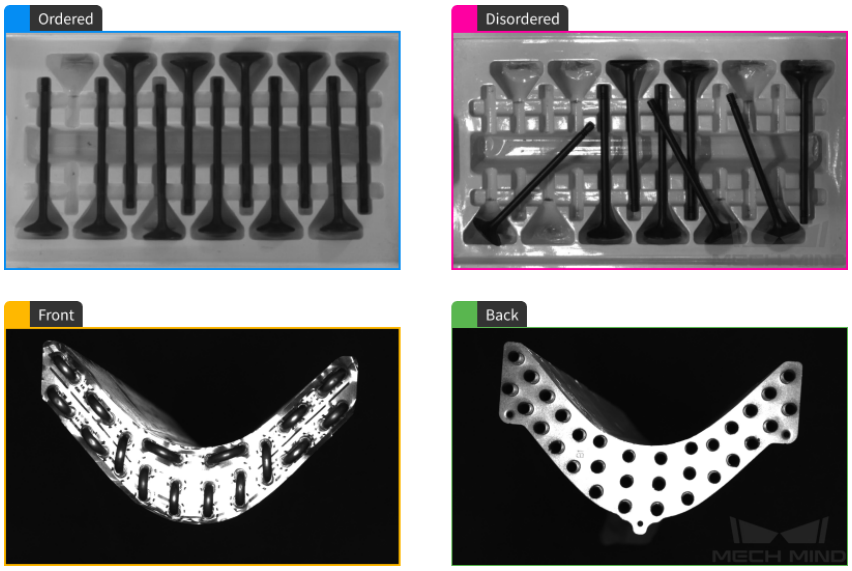
2.3. Train a Classification Model

2.3.1. Introduction

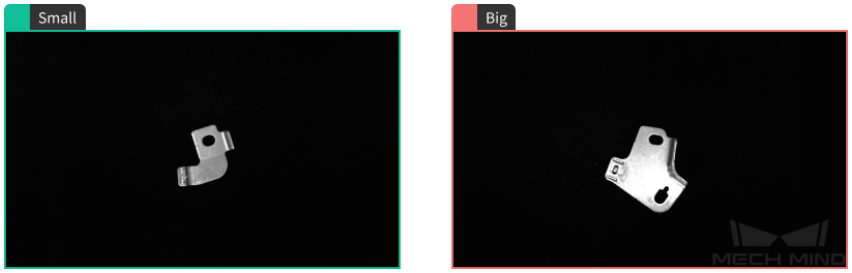
This module is used to classify different images.

Application Scenarios

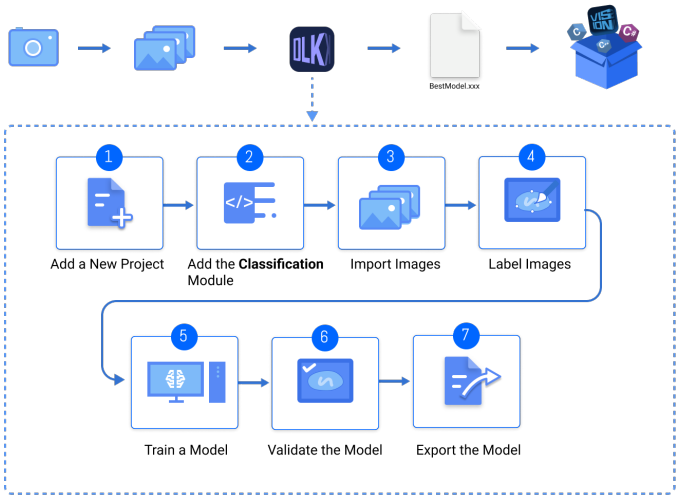
Machine tending: Suitable for classifying the front and back sides, positions, types, or other properties of the workobjects in industries such as steel and machinery.



© 2023 Mech-Mind Robotics Technologies Ltd.



General Workflow

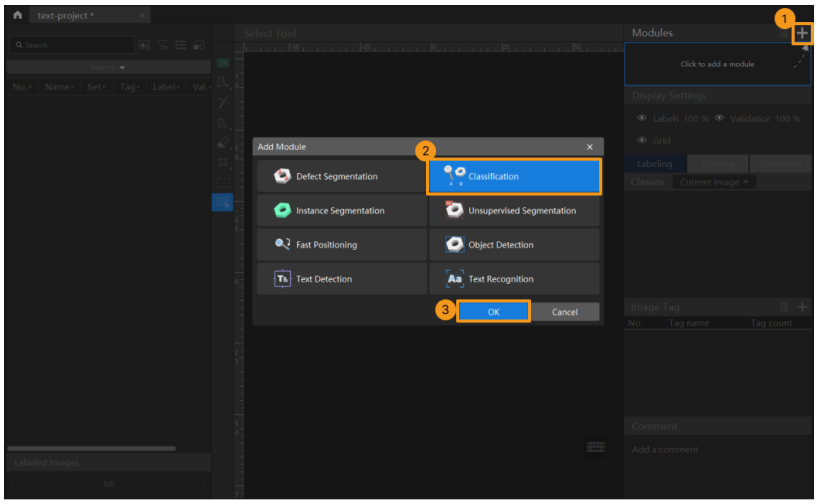


2.3.2. Use the Classification Module

Please click here to download an image dataset of condensers, an example project in Mech-DLK. In this section, we will use the Classification module to train a model that can distinguish between the front and back sides of the condensers.

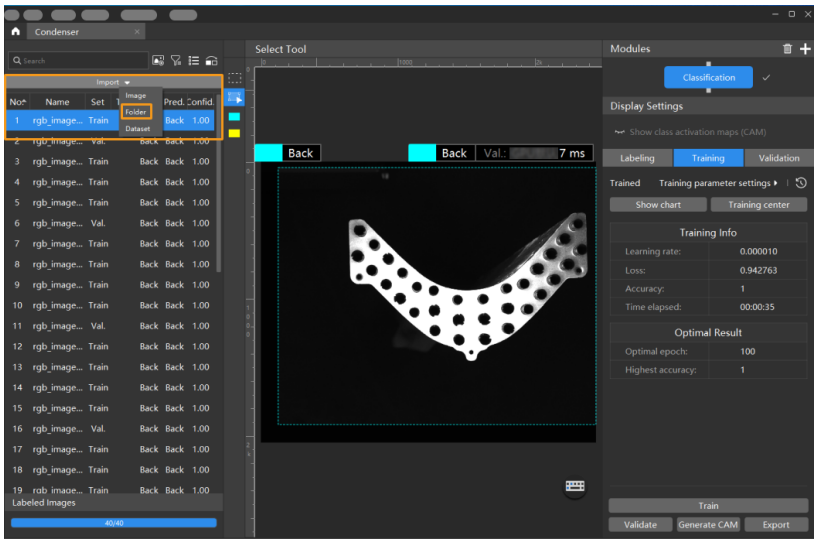
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a New Project and add the Classification module**: Click **[New Project]** in the interface, name the project, and select a directory to save the project. Click **+** in the upper right corner of the Modules panel and add the Classification module.

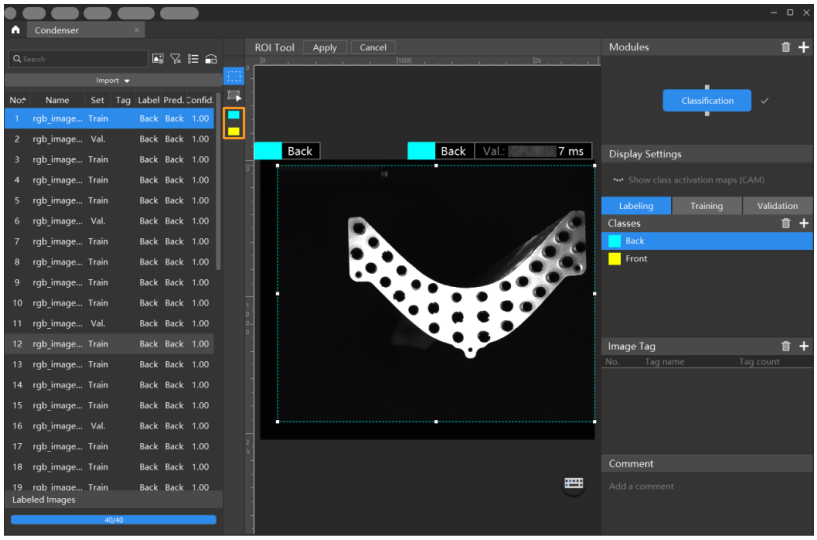


© 2023 Mech-Mind Robotics Technologies Ltd.

2. **Import the image dataset of condensers**: Unzip the downloaded dataset file. Click the **[Import]** button in the upper left corner, select **[Folder]**, and import the image dataset.



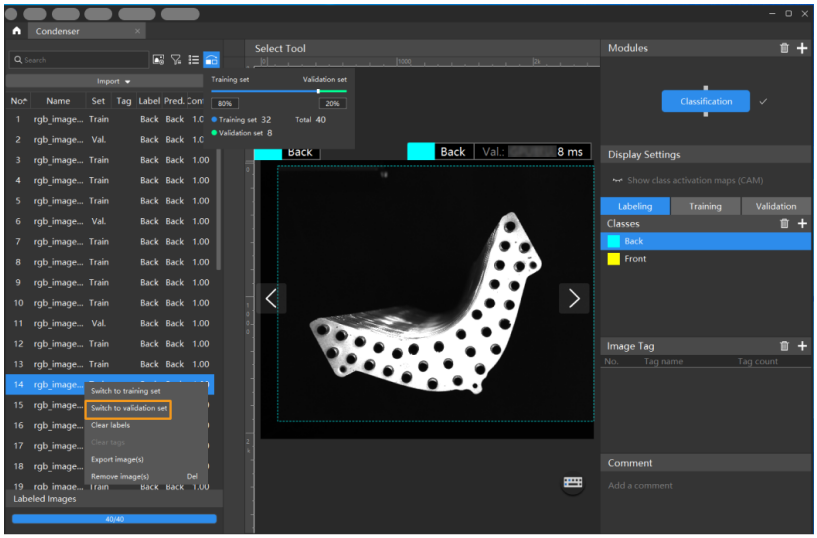
3. **Create Labels**: Select Labeling and click the **[+]** button to create labels based on the type or feature of different objects. In this example, the labels are named front and back to distinguish between the front and back sides of the condenser.
4. **Label images**: Classify the images with corresponding labels. You can select multiple images and label them together. Please make sure that you have labeled the images accurately.



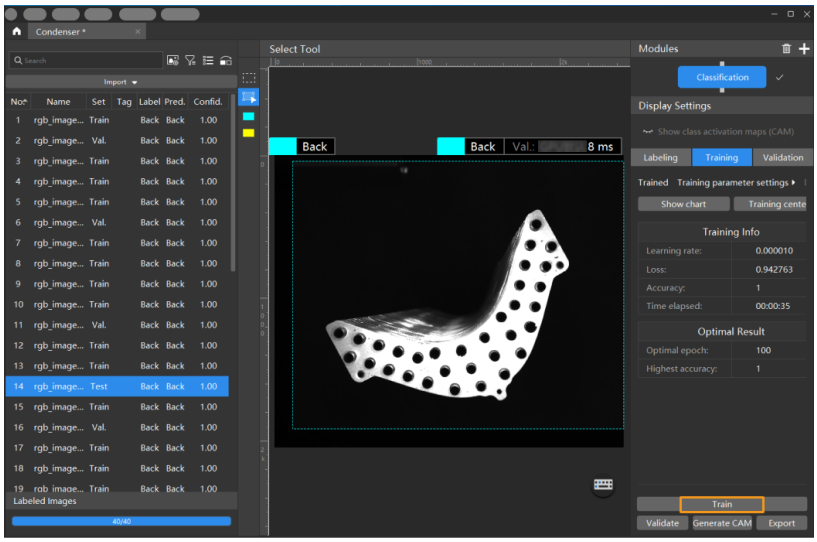
The Classification module supports selecting multiple images for labeling in batches.

5. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. Please make sure that both the training set and validation set include **images in all different classes**, which will guarantee that the model can learn all the different features and validate the images of different classes properly. If the default training set and validation set cannot meet this requirement, please right-click the name of the image and then click **Switch to training set** or **Switch to validation set** to adjust the set to which the image belongs.

© 2023 Mech-Mind Robotics Technologies Ltd.

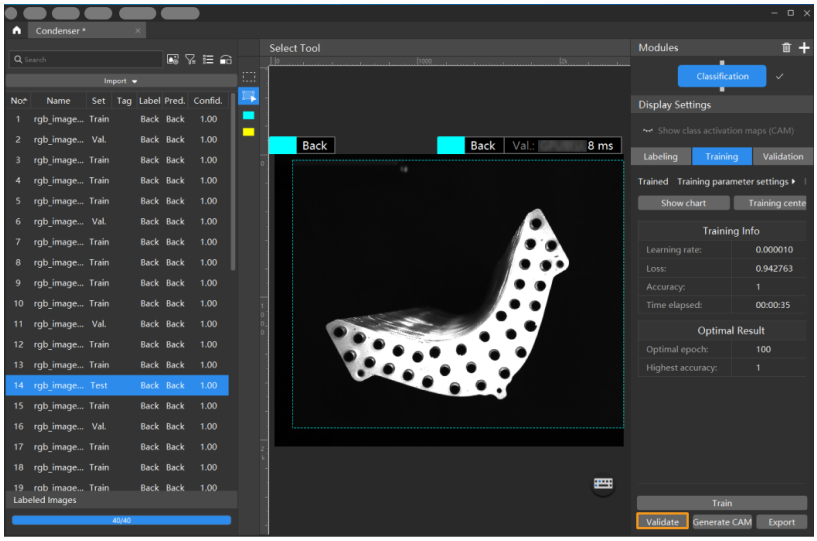


6. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.



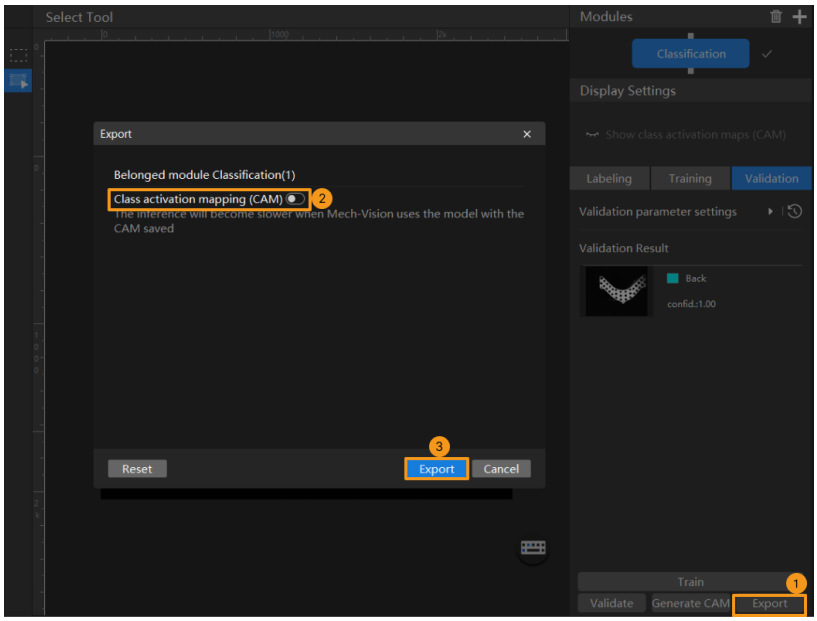
7. **Validate the model**: After the training is completed, click **[Validate]** to validate the model and check the results. You can also click > **Wrong results** > **OK** to filter the results and check the wrong ones.

© 2023 Mech-Mind Robotics Technologies Ltd.



8. **Export the model**: Click **[Export]**, set the **Class activation mapping (CAM)** option of the model to be exported in the pop-up window, and then click 1Export and select a directory to save the exported model.

By default, the **Class activation mapping (CAM)** option is disabled. The inference will slow down when the model saved with CAMs is used in Mech-Vision.



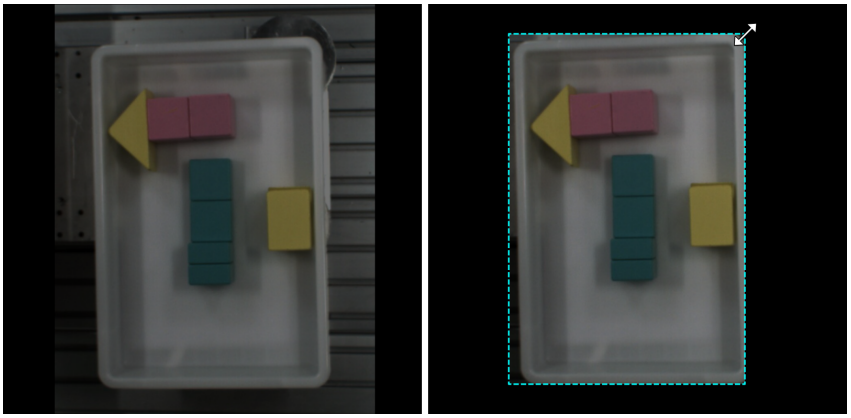
The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.3.3. ROI Tool

You can use the ROI Tool to set the region of interest.

Setting the ROI can avoid interferences from the background.

© 2023 Mech-Mind Robotics Technologies Ltd.



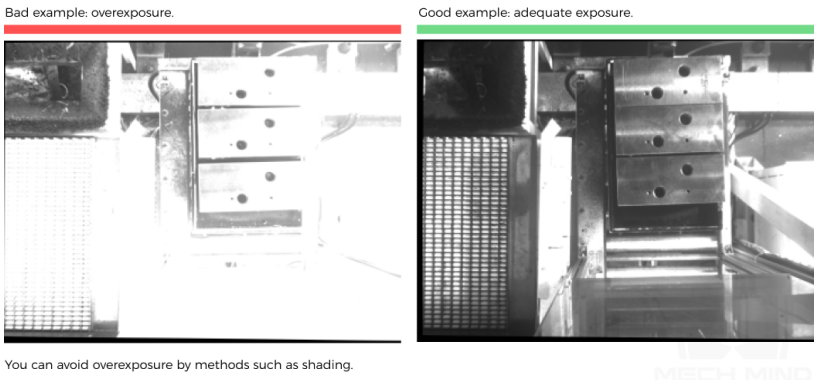
1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.
3. Click **[Apply]** in the upper left corner of the labeling interface.

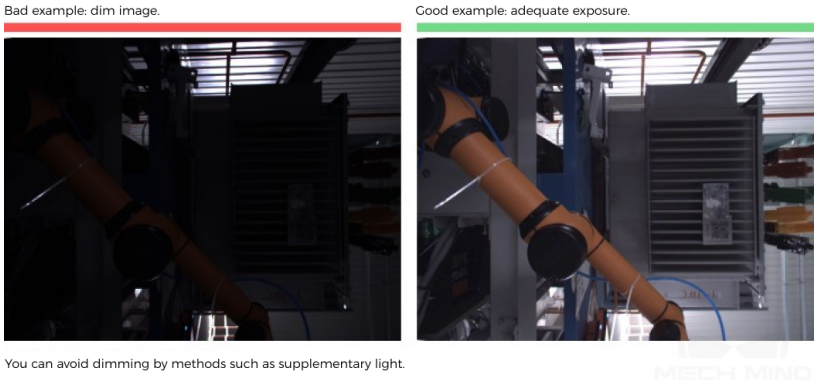
2.3.4. Train a High-Quality Model

This section introduces the factors that most affect the model quality and how to train a high-quality classification model.

Ensure Image Quality

1. Avoid **overexposure, dimming, color distortion, blur, occlusion**, etc. These conditions can lead to the loss of features that the deep learning model relies on, which will affect the model training effect.

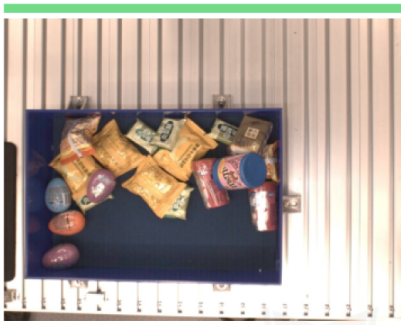




© 2023 Mech-Mind Robotics Technologies Ltd.

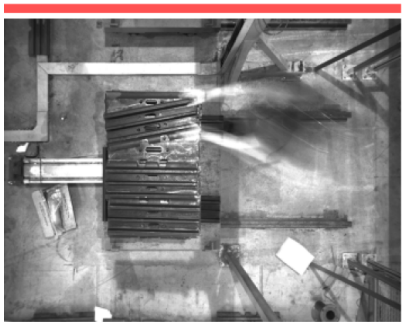
Bad example: color distortion.

Good example: normal color.



Color distortion can be avoided by adjusting the white balance.

Bad example: blur.

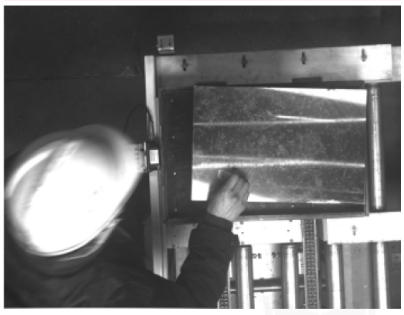


Good example: clear.

Please avoid capturing images when the camera or the objects are still moving.

Bad example: occluded by the robot arm.

Bad example: occluded by a human.



Please make sure there is no robot or human in the way from the camera to the objects.

2. Ensure that the **background, perspective, and height** of the image-capturing process are consistent with the actual application. Any inconsistency can reduce the effect of deep learning in practical applications. In severe cases, data must be re-collected. Please confirm the conditions of the actual application in advance.

Bad example: The background in the training data (left) is different from the background in the actual application (right).

Please make sure the background stays the same when capturing the training data and when deploying the project.

© 2023 Mech-Mind Robotics Technologies Ltd.

Bad example: The field of view and perspective in the training data (left) are different from that in the actual application (right).

Please make sure the field of view and perspective stay the same when capturing the training data and when deploying the project.

Bad example: The camera height in the training data (left) is different from the background in the actual application (right).

Please make sure the camera height stays the same when capturing the training data and when deploying the project.

The quality of image classification is sensitive to lighting, and the lighting conditions need to be consistent during collection. If the light is inconsistent in the morning and evening, data needs to be collected separately according to the situation.

Ensure Data Quality

The Classification module obtains a model by learning the features of existing images and applies what is learned to the actual application. Therefore, to train a high-quality model, the conditions of the collected and selected data must be consistent with those of the actual applications.

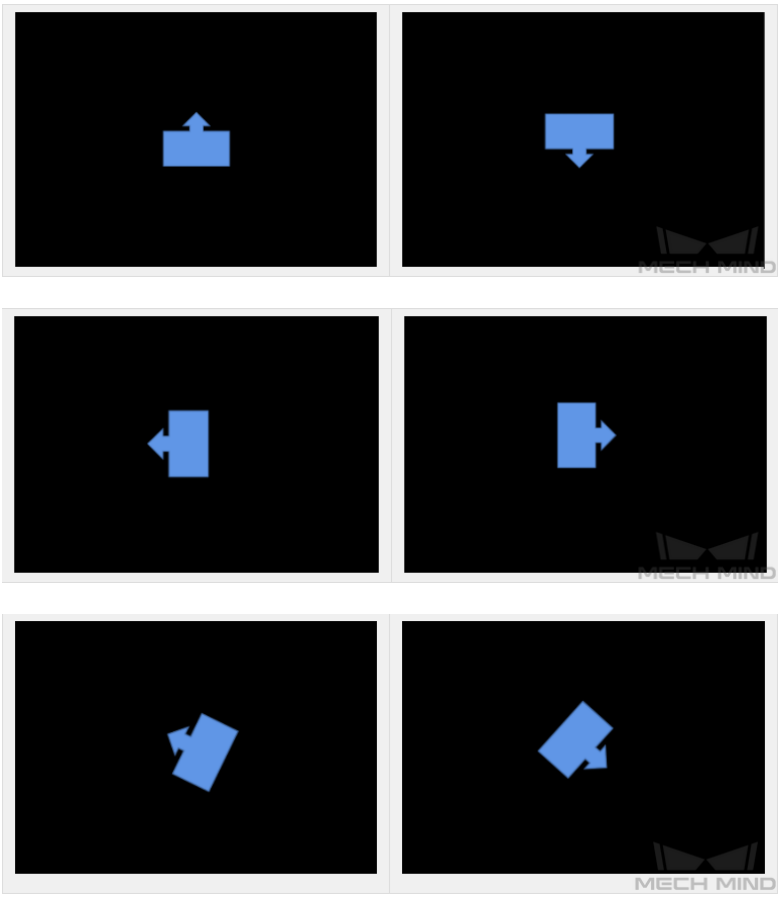
Collect Data

Various placement conditions need to be properly allocated. For example, if there are horizontal and vertical incoming materials in actual production, but only the data of horizontal incoming materials are collected for training, the classification effect of vertical incoming materials cannot be guaranteed. Therefore, during data collection, it is necessary to **consider various conditions** of the actual application, including the following:

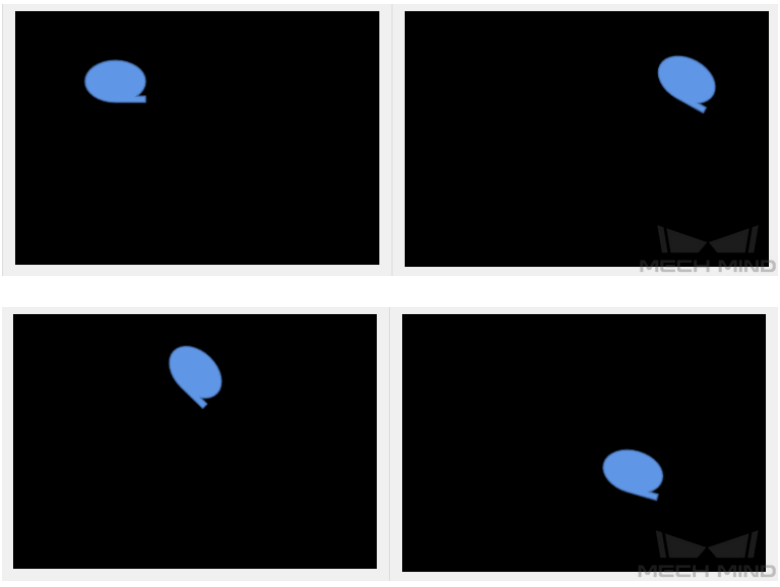
- The features presented given different object placement **orientations**.
- The features presented given different object placement **positions**.

1. Different orientations

© 2023 Mech-Mind Robotics Technologies Ltd.



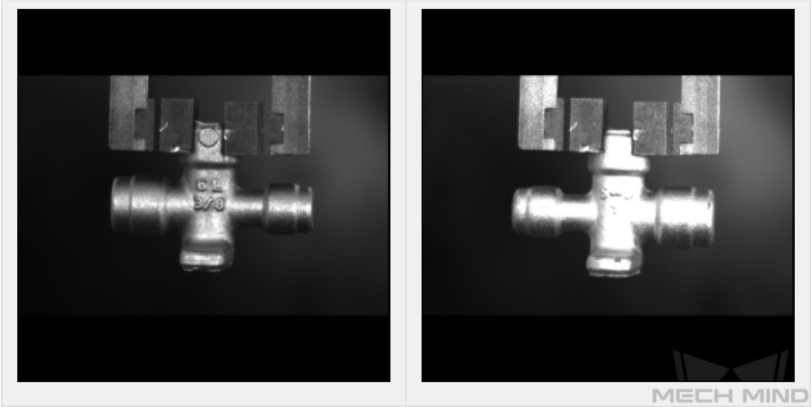
2. Different positions



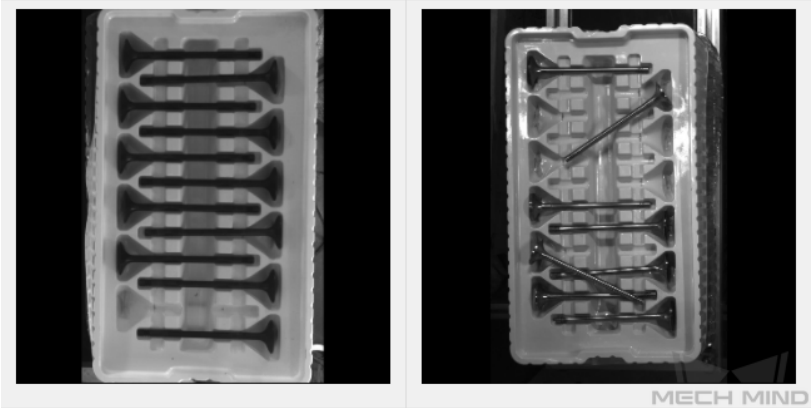
Data Collection Examples

1. A valve tube project: Single object class. Distinguishing between the front and back sides of the valve tubes is needed. Positions are generally fixed with small deviations. Fifteen images for the front and back sides each were collected.

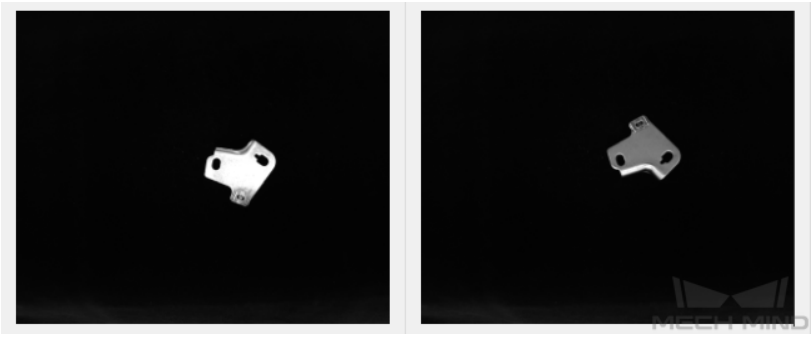
© 2023 Mech-Mind Robotics Technologies Ltd.



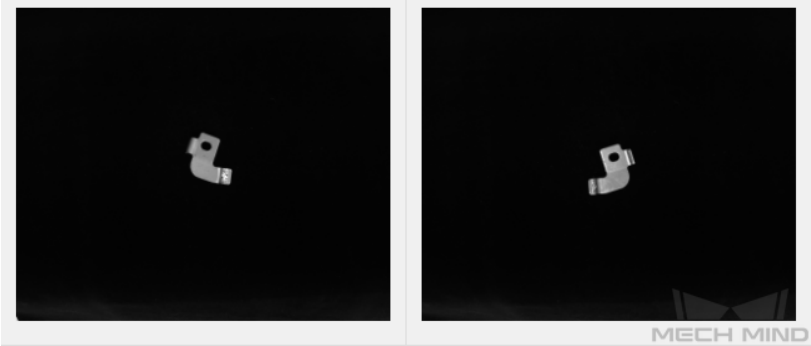
2. An engine valve assembly project: Single object class. Determining whether the object is correctly placed in the slot is needed. Since outside the slot, the object may appear in various positions and orientations, it is necessary to consider different positions and orientations, and 20 images were collected for objects outside the slot. In the slot, only the factor of different positions needs to be considered, so 10 images were collected for objects inside the slot.



3. A sheet metal project: Two object classes. Different object sizes need to be recognized. Objects may come in different positions and orientations. Twenty images were collected for the front and back sides each.



© 2023 Mech-Mind Robotics Technologies Ltd.



Select the Right Dataset

1. Control dataset image quantities

   For the first-time model building of the Classification module, capturing 30 images is recommended. It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later, and will make the training time longer.

2. Collect representative data

   Image capturing should consider all the conditions in terms of illumination, color, size, etc. of the objects to be recognized.

   - Lighting: Project sites usually have environmental lighting changes, and the data should contain images with different lighting conditions.
   - Color: Objects may come in different colors, and the data should contain images of objects of all the colors.
   - Size: Objects may come in different sizes, and the data should contain images of objects of all existing sizes.

   If the actual on-site objects may be rotated, scaled in images, etc., and the corresponding images cannot be collected, the data can be supplemented by adjusting the data augmentation training parameters to ensure that all on-site conditions are included in the datasets.

3. Balance data proportion

   The number of images of different object classes in the datasets should be proportioned according to the actual project; otherwise, the training effect will be affected. There should be no such case where 20 images are of one object, and only 3 are of the other object.

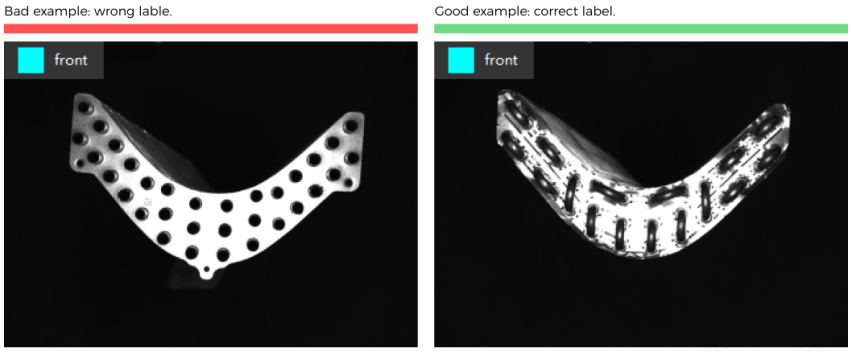
4. Images should be consistent with the application site

   The factors that need to be consistent include lighting conditions, object features, background, and field of view.

Ensure labeling quality

Please ensure **consistency**, namely that there are no missed or incorrect labels.

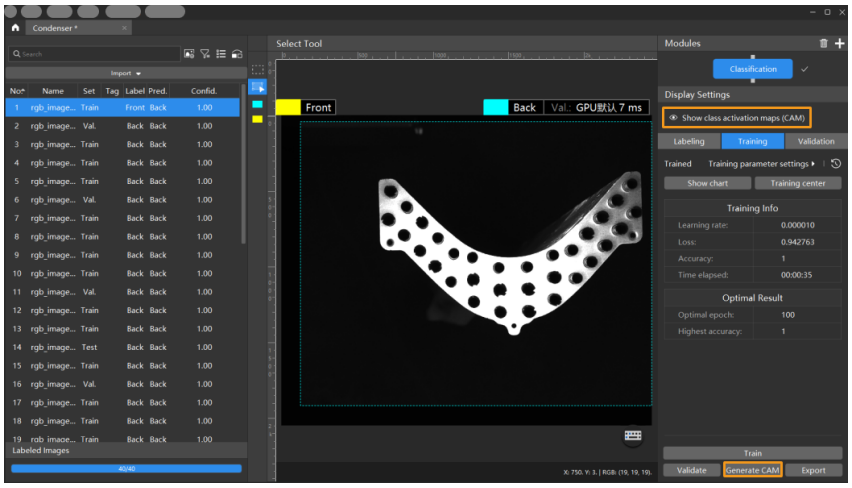
© 2023 Mech-Mind Robotics Technologies Ltd.



The left image is the workpiece front, and the right image is the workpiece back.

Class Activation Maps

After the training of the image classification model is completed, click **[Generate CAM]** to generate the class activation maps, and click **[Show class activation maps (CAM)]**. The class activation maps show the feature regions in the images that are paid attention to when training the model, and they help check the classification performance, thus providing references for optimizing the mode.



2.4. Train an Object Detection Model

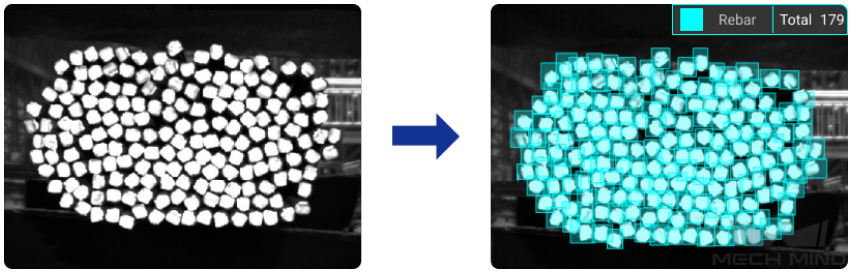
2.4.1. Introduction

This module is used to detect the location of all target objects and estimate their classes.

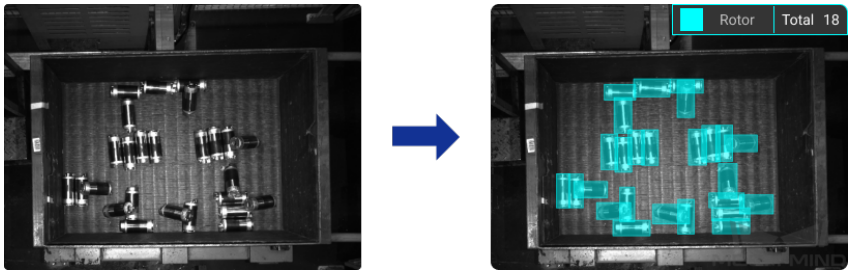
Application Scenarios

Counting workobjects: Suitable for counting bundles of steel bars, loose parts, and tiny parts in factories.

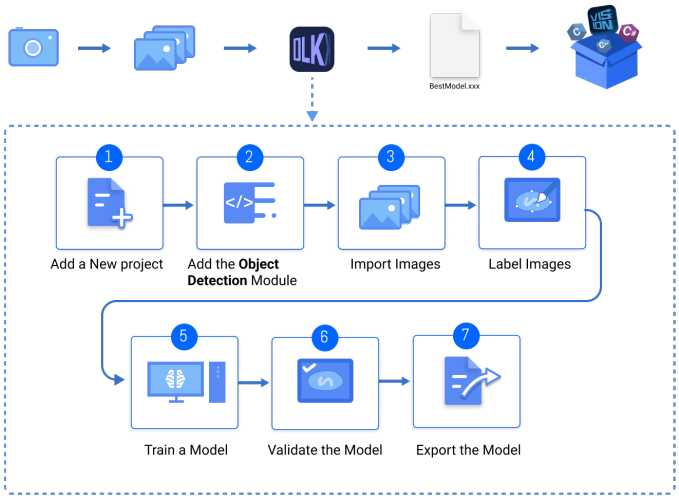
© 2023 Mech-Mind Robotics Technologies Ltd.



Detecting the location of workobjects: Suitable for detecting and locating metal parts in factories or on assembly lines.



General Workflow



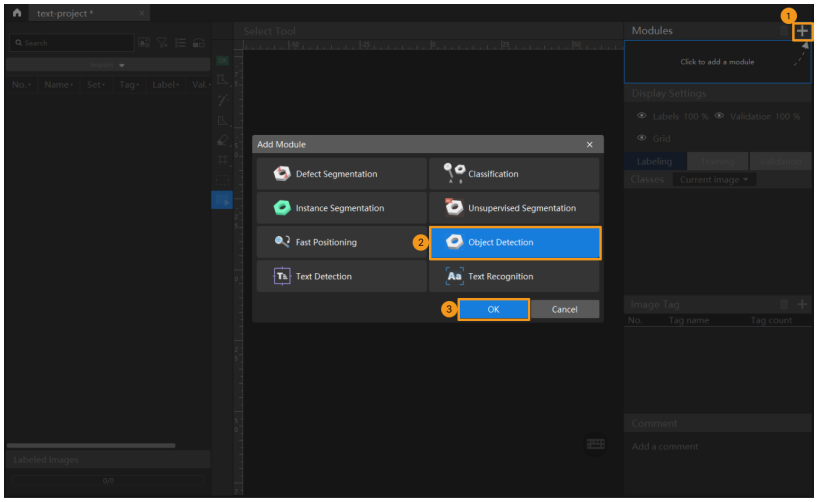
2.4.2. Use the Object Detection Module

Please click here to download an image dataset of rotors. In this section, we will use an Object Detection module and train a model to detect the positions of rotors in the image and output the quantity.

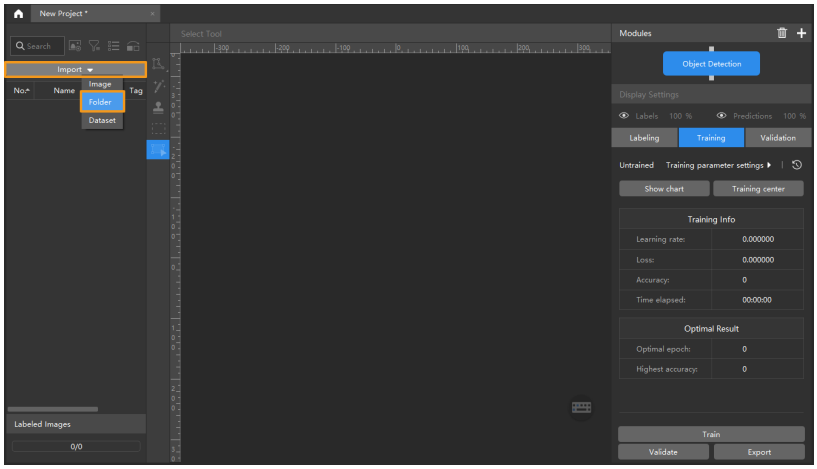
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Object Detection module**: Click **[New Project]** in the interface, name the project, and select a directory to save the project. Click **+** in the upper right corner of the Modules panel and add the Object Detection module.

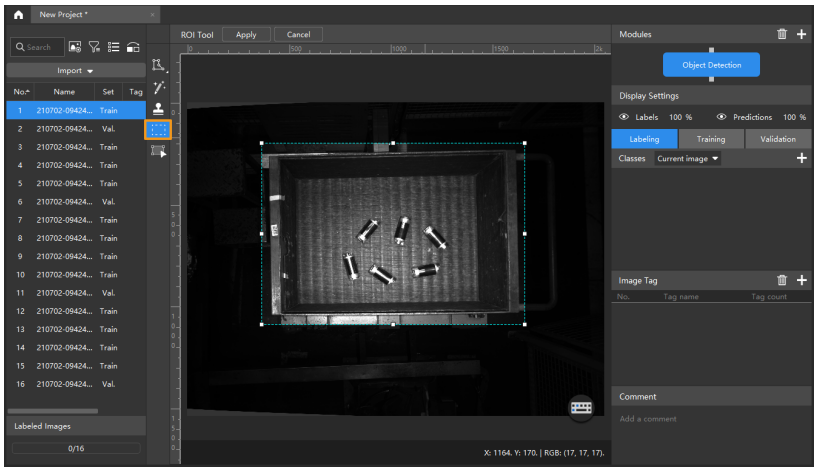
© 2023 Mech-Mind Robotics Technologies Ltd.



2. **Import the image dataset of rotors**: Unzip the downloaded dataset file. Click the **[Import]** button in the upper left corner, select **[Folder]**, and import the image data.



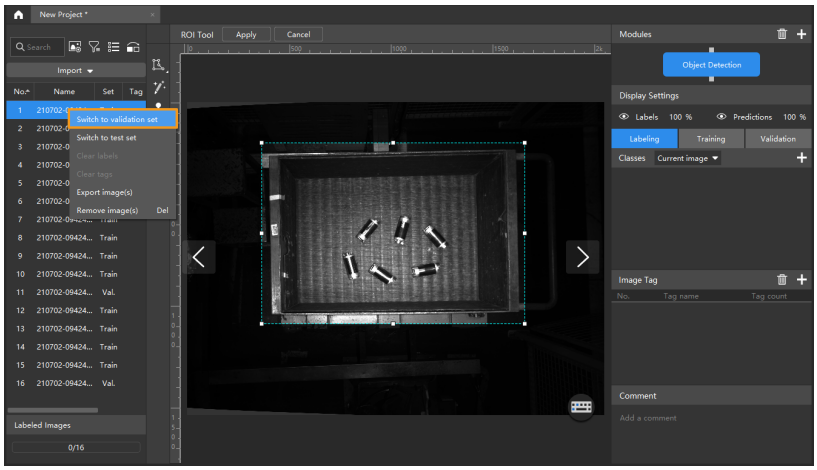
3. **Select an ROI**: Click the ROI Tool button and adjust the frame to select the bin containing rotors in the image as an ROI, and click **[Apply]** to save the settings. Setting the ROI can avoid interferences from the background and reduce processing time.



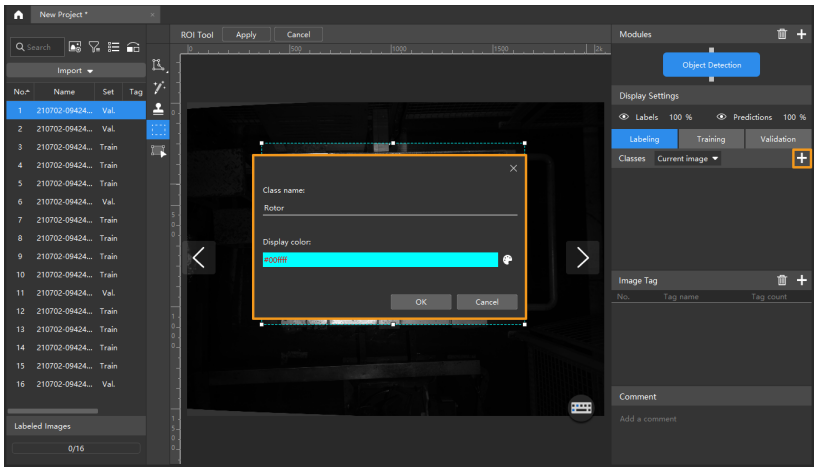
4. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. You can click and drag the slider to adjust the proportion. Please make sure that both the

© 2023 Mech-Mind Robotics Technologies Ltd.

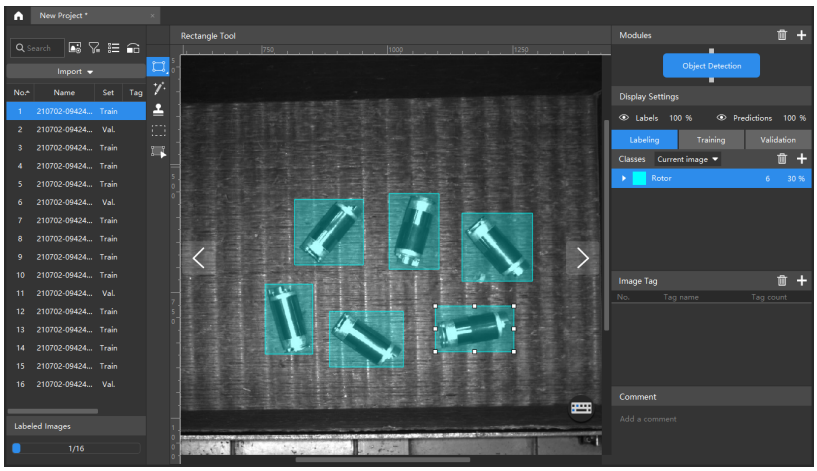
training set and validation set include objects of all classes to be detected. If the default training set and validation set cannot meet this requirement, please right-click the name of the image and then select **Switch to training set** or **Switch to validation set** to adjust the set to which the image belongs.



5. **Create Labels**: Create labels based on the type or feature of different objects. In this example, the labels are named after the rotors.

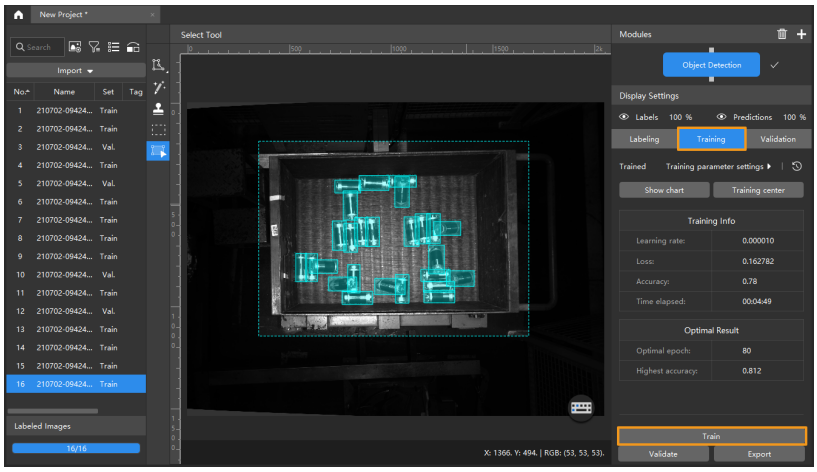


6. **Label images**: Make rectangular selections on the images to label all the rotors. Please select the rotors as precisely as possible and avoid including irrelevant regions. Inaccurate labeling will affect the training result of the model. Click here to view how to use labeling tools.

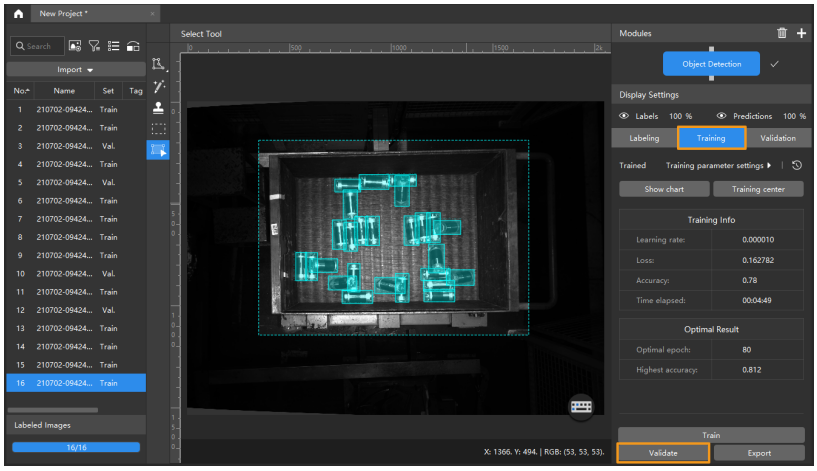


© 2023 Mech-Mind Robotics Technologies Ltd.

7. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.



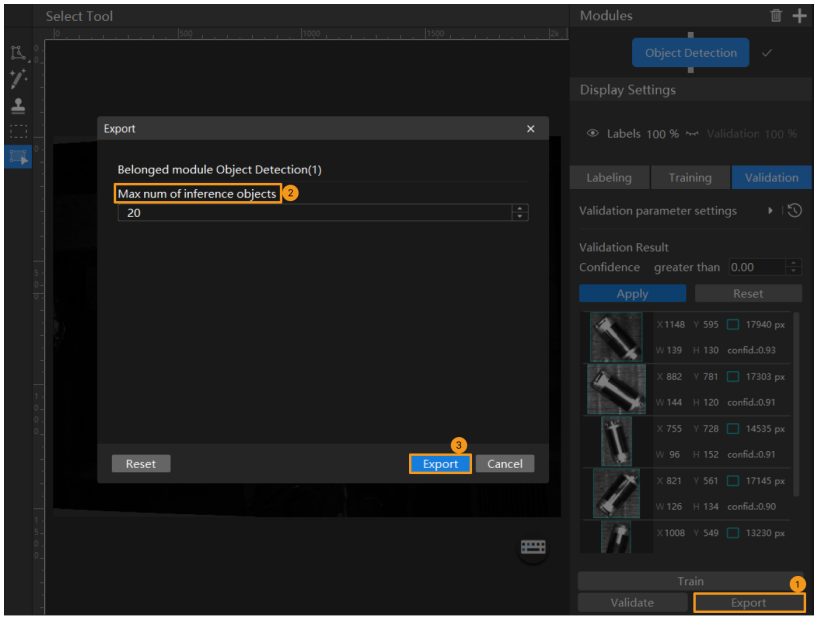
8. **Validate the model**: After the training is completed, click **[Validate]** to validate the model and check the results.



9. **Export the model**: Click **[Export]**. Then, set **Max num of inference objects** in the pop-up dialog box, click **[Export]**, and select a directory to save the exported model.

The maximum number of objects during a round of inference, which is 100 by default.

© 2023 Mech-Mind Robotics Technologies Ltd.



The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.4.3. Introduction to Labeling Tools

You can use labeling tools to label the images and hence provide data for deep learning training.

You can choose among the following labeling tools built in the software according to actual needs.

Please create labels of corresponding classes according to the needs of the current project before using labeling tools.

Polygon Tool

The **Polygon Tool** can draw polygon labels with more vertices, which is suitable for objects of complex shapes.

1. Click (or press P on the keyboard).
2. Click the first position (vertex) in the selection region, then click the second one, third one, etc., to draw the labels, and right-click to finish. (At least three vertices are required.)
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

After labeling, use the **Select Tool** to select a label and adjust the label by the following methods.

- Click the label edges to increase the number of vertices.
- Right-click a vertex to delete it.
- Long press the left mouse button and drag the vertex in any direction to modify the label shape.

Ellipse Tool

Use more vertices to make elliptical selections. This tool is suitable for elliptical objects.

© 2023 Mech-Mind Robotics Technologies Ltd.

1. Right-click and then click (or press L on the keyboard).
2. Click the first position (vertex) in the selection region, and then continue clicking. An elliptical label should have at least five vertices.
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

After labeling, use the **Select Tool** to select the label and then long press the left mouse button to drag the vertex in any direction and thus modify the label shape.

Rectangle Tool

The **Rectangle Tool** can be used to draw rectangular labels, which is suitable for rectangular objects.

1. Right-click and then click (or press R on the keyboard).
2. Long press the left mouse button in the selection region, move it in any direction, and then release the left mouse button to finish the rectangular selection.
3. If multiple label classes are created, colors corresponding to different label classes should be selected.

Smart Labeling Tool

The **Smart Labeling Tool** can be used to automatically select the objects in the image.

When multiple objects in an image have large color differences and are scattered, you can use the **Smart Labeling Tool** to conveniently label the objects in the image.

1. Click (or press M on the keyboard).
2. Move the cursor in the selection region and then click the object to be labeled.
   - If the selection cannot completely cover the object, click the uncovered part to expand the selection area.
   - If the selection covers the areas outside the object, right-click these areas to reduce the selection area.
3. Click **[Apply]** in the upper-left corner of the selection region.

You can use the **Select Tool** to fine-tune the labeled contour by the following steps:

1. Use the **Select Tool** to select the label to be adjusted.
2. Adjust the contour in one of the following three ways according to actual situation. Please ensure that the selected area closely aligns with the object contour.
   a. Place the mouse cursor on a vertex of the contour. When the cursor turns into , long-press the left mouse button and drag the vertex to adjust the contour.
   b. Place the mouse cursor on a vertex of the contour. When the cursor turns into , click the right mouse button to delete the vertex.
   c. Place the mouse cursor on the contour. When the cursor turns into , click the left mouse button to add a vertex.

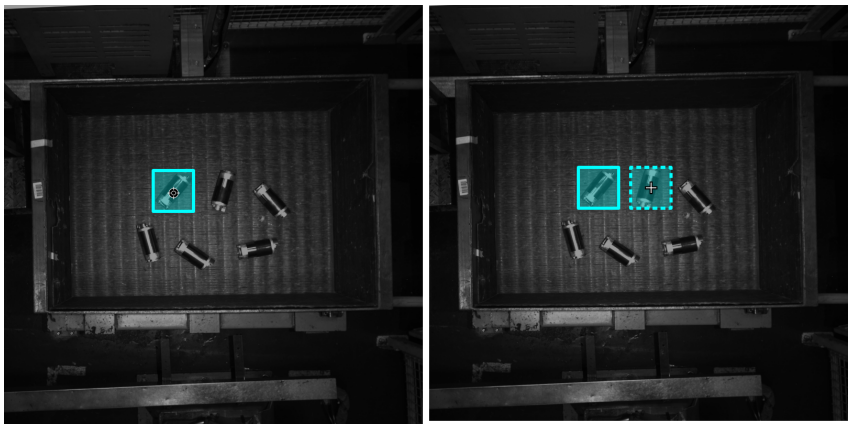
When the objects differ greatly in colors and have clear contours, it is recommended that you label multiple objects at a time and click **[Apply]**. If the objects are not obviously distinct, it is recommended to label one at a time.

© 2023 Mech-Mind Robotics Technologies Ltd.

Template Tool

You can use the **Template Tool** to set an existing selection as a template. After setting, you can use this template to rapidly label contours and objects with the same pose.

It is suitable for scenarios where there are multiple neatly arranged objects of the same type in an image, which can improve labeling efficiency.



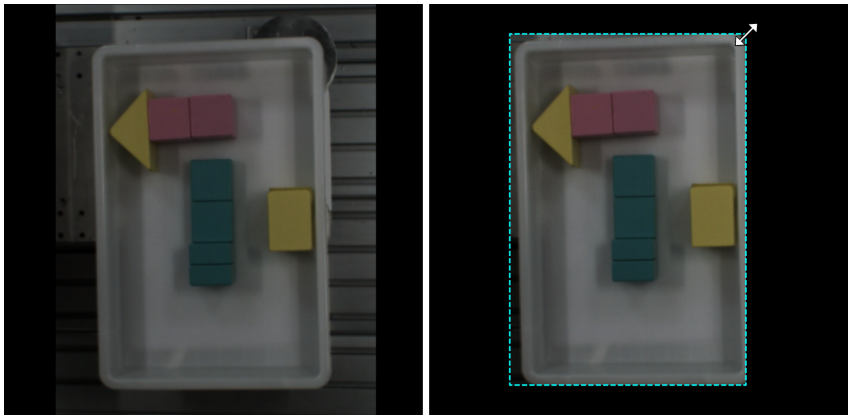
1. Click (or press C on the keyboard).
2. Click the region that needs to be set as the template.
3. Move the template to the to-be-selected object, adjust the angle of the template to make it fit the object, and then click it.
   - Coarse adjustment: Press and hold the `Shift` key and then scroll the mouse wheel.
   - Fine adjustment: Adjust the **Rotation angle** parameter.

During labeling, press and hold the `Ctrl` key and click the selection to switch the template. You can also click **[Replace template]** and then click the selection to achieve the same purpose.

ROI Tool

You can use the **ROI Tool** to set the region of interest.

Setting the ROI can avoid interferences from the background.



1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.

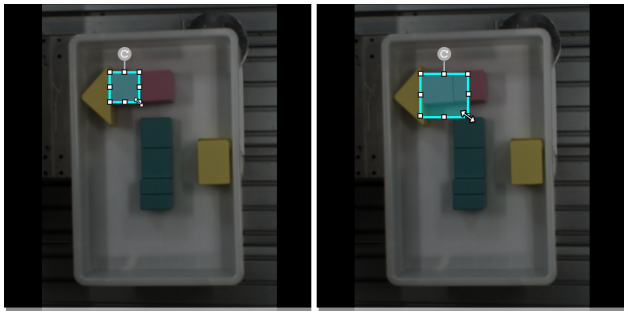
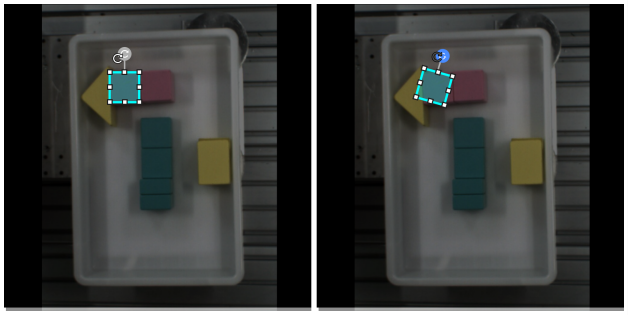
© 2023 Mech-Mind Robotics Technologies Ltd.

3. Click **[Apply]** in the upper left corner of the selection region.

Select Tool

You can use the **Select Tool** to select, move, and adjust the selections.

1. Click (or press S on the keyboard).
2. Move the cursor in the selection region and then click the selection to be processed. Select multiple selections by pressing and holding the `Ctrl` key.

Adjust size	
Rotation	
Move	

2.4.4. Train a High-Quality Model

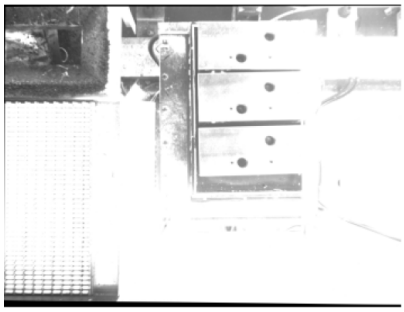
This section introduces the factors that most affect the model quality and how to train a high-quality object detection model.

Ensure Image Quality

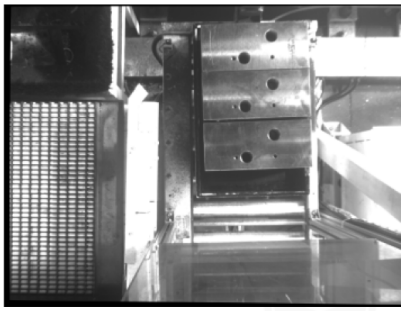
1. Avoid **overexposure, dimming, color distortion, blur, occlusion**, etc. These conditions can lead to the loss of features that the deep learning model relies on, which will affect the model training effect.

© 2023 Mech-Mind Robotics Technologies Ltd.

Bad example: overexposure.



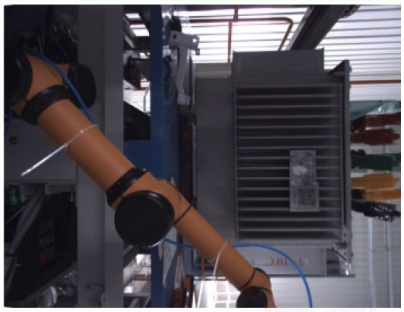
Good example: adequate exposure.



You can avoid overexposure by methods such as shading.

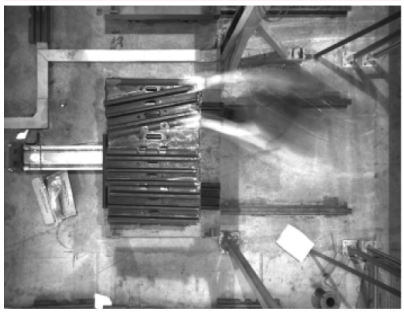
Bad example: dim image.

Good example: adequate exposure.



You can avoid dimming by methods such as supplementary light.

Bad example: blur.



Good example: clear.

Please avoid capturing images when the camera or the objects are still moving.

Bad example: occluded by the robot arm.

Bad example: occluded by a human.

Please make sure there is no robot or human in the way from the camera to the objects.

2. Ensure that the **background, perspective, and height** of the image-capturing process are consistent with the actual application. Any inconsistency can reduce the effect of deep learning in practical applications. In severe cases, data must be re-collected. Please confirm the conditions of the actual application in advance.

© 2023 Mech-Mind Robotics Technologies Ltd.

Bad example: The background in the training data (left) is different from the background in the actual application (right).

Please make sure the background stays the same when capturing the training data and when deploying the project.

Bad example: The field of view and perspective in the training data (left) are different from that in the actual application (right).

Please make sure the field of view and perspective stay the same when capturing the training data and when deploying the project.

Bad example: The camera height in the training data (left) is different from the background in the actual application (right).

Please make sure the camera height stays the same when capturing the training data and when deploying the project.

Ensure Data Quality

The Object Detection module obtains a model by learning the features of existing images and applies what is learned to the actual application. Therefore, to train a high-quality model, the conditions of the collected and selected data must be consistent with those of the actual applications.

Collect Data

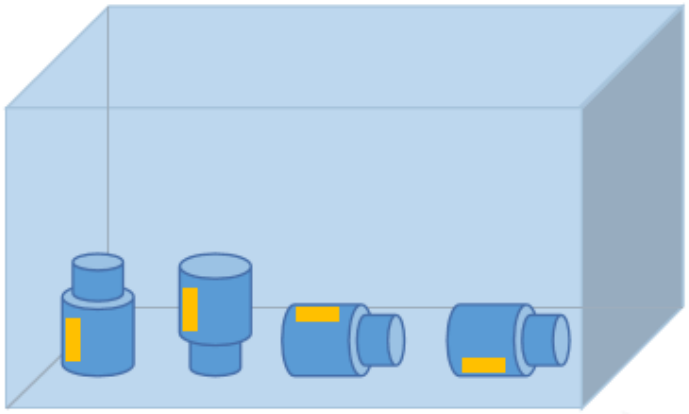
Various placement conditions need to be properly allocated. For example, if there are horizontal and vertical incoming materials in actual production, but only the data of horizontal incoming materials are collected for training, the recognition effect of vertical incoming materials cannot be guaranteed. Therefore, when collecting data, it is necessary to **consider various conditions** of the actual application, including the following:

© 2023 Mech-Mind Robotics Technologies Ltd.

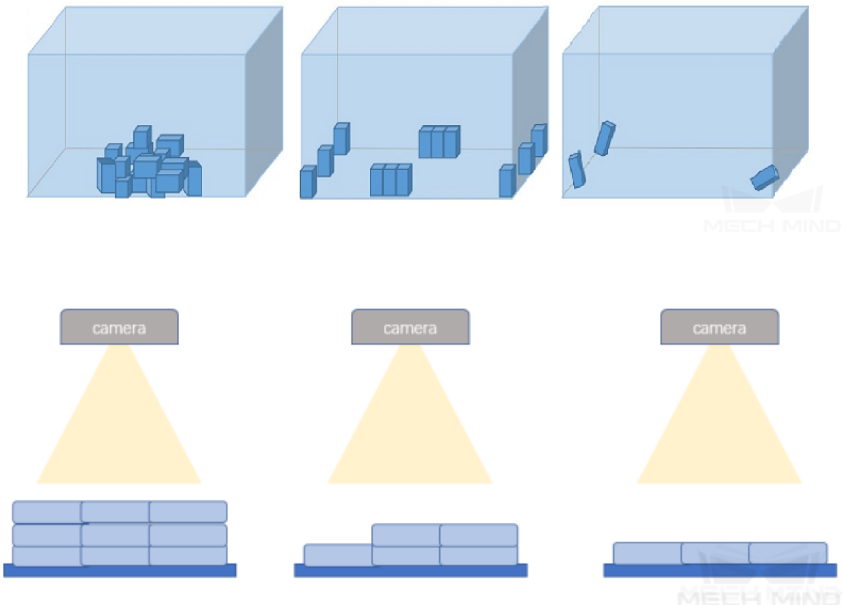
- The features presented given different object placement **orientations**.
- The features presented given different object placement **positions**.
- The features presented given different **positional relationships between objects**.

If some situations are not in the datasets, the deep learning model will not go through inadequate learning on the corresponding features, which will cause the model to be unable to effectively make recognitions given such conditions. In this case, data on such conditions must be collected and added to reduce the errors.

Orientations

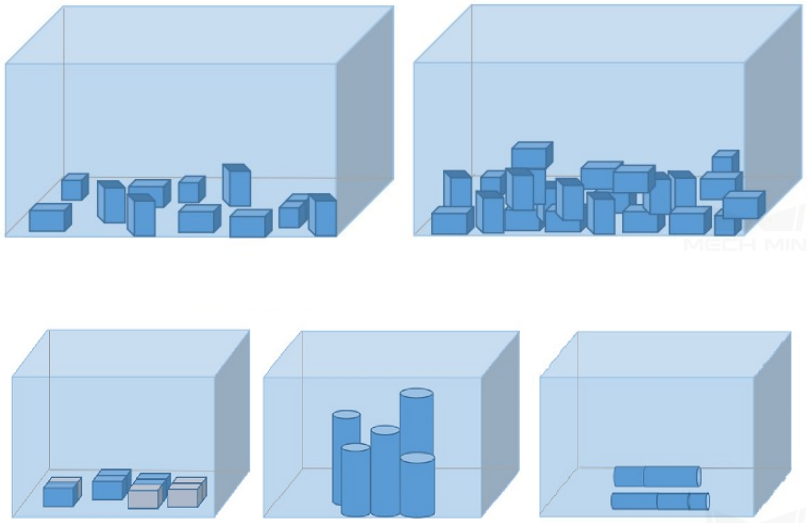


Object positions



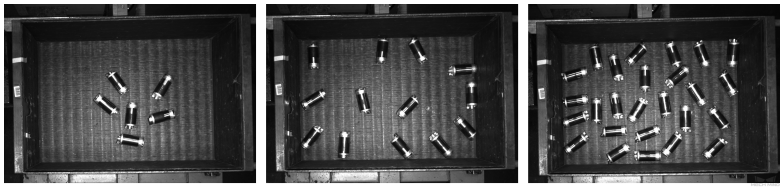
Positional relationships between objects

© 2023 Mech-Mind Robotics Technologies Ltd.

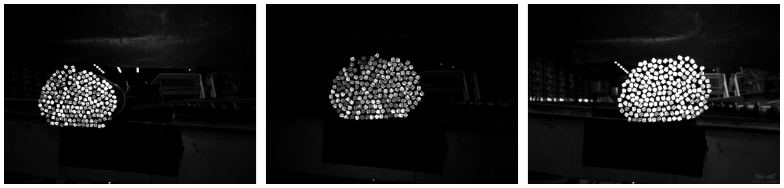


Data Collection Examples

1. A workobject inspection project: The incoming objects are rotors scattered randomly. The project requires accurate detection of all rotor positions. Thirty images were collected.
   - **Positions**: In the actual application, the rotors may be in any position in the bin, and the quantity will decrease after picking each time.
   - **Positional relationships**: The rotor may come scattered, neatly placed, or overlapped.



2. A steel bar counting project: The incoming objects are steel bars in bundles. The project requires accurate counting of steel bars. Twenty images were collected.
   - Steel bars have relatively simple features, so only the variations of **object positions** need to be considered. Images in which steel bars are in any position in the camera's field of view were captured.



Select the Appropriate Data

1. Control the image quantity of datasets

   For the first-time model building of the Object Detection module, capturing 20 images is recommended. It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later, and will make the training time longer.

© 2023 Mech-Mind Robotics Technologies Ltd.

2. Collect representative data

   Image capturing should consider all the conditions in terms of illumination, color, size, etc. of the objects to be recognized.

   - **Lighting**: Project sites usually have environmental lighting changes, and the data should contain images with different lighting conditions.
   - **Color**: Objects may come in different colors, and the data should contain images of objects of all the colors.
   - **Size**: Objects may come in different sizes, and the data should contain images of objects of all existing sizes.

   > If the actual on-site objects may be rotated, scaled in images, etc., and the corresponding images cannot be collected, the data can be supplemented by adjusting the data augmentation training parameters to ensure that all on-site conditions are included in the datasets.

3. Balance data proportion

   The number of images of **different conditions/object classes** in the datasets should be proportioned according to the actual project; otherwise, the training effect will be affected. There should be no such case where 20 images are of one object, and only 3 are of the other object, or the case where 40 images are of the objects neatly arranged, and only 5 are of scattered ones.

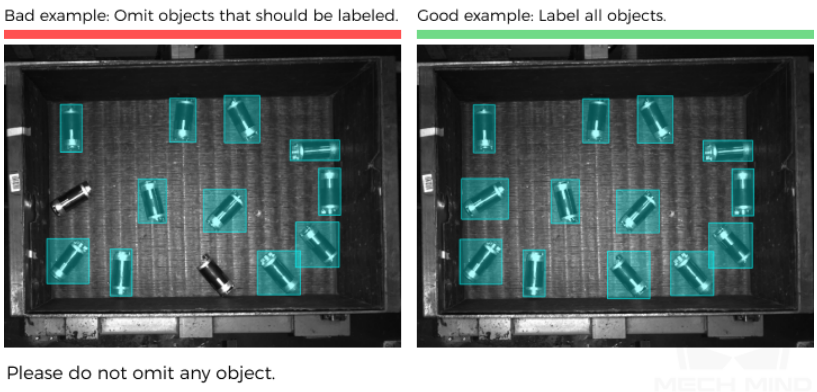
4. Images should be consistent with the application site

   The factors that need to be consistent include lighting conditions, object features, background, and field of view.

Ensure Labeling Quality

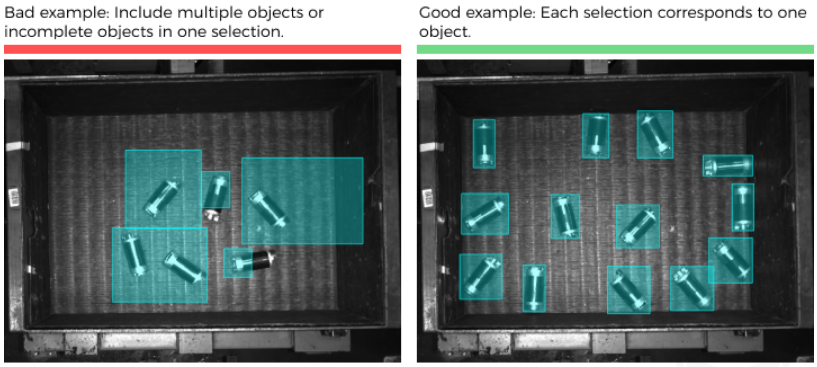
Labeling quality should be ensured in terms of completeness and accuracy.

1. **Completeness**: Label all objects that meet the rules, and avoid missing any objects or object parts.



2. **Accuracy**: Each rectangular selection should contain the entire object. Please avoid missing any object parts, or including excess regions outside the object contours.

© 2023 Mech-Mind Robotics Technologies Ltd.



Please do not include unnecessary regions or omit necessary object parts.

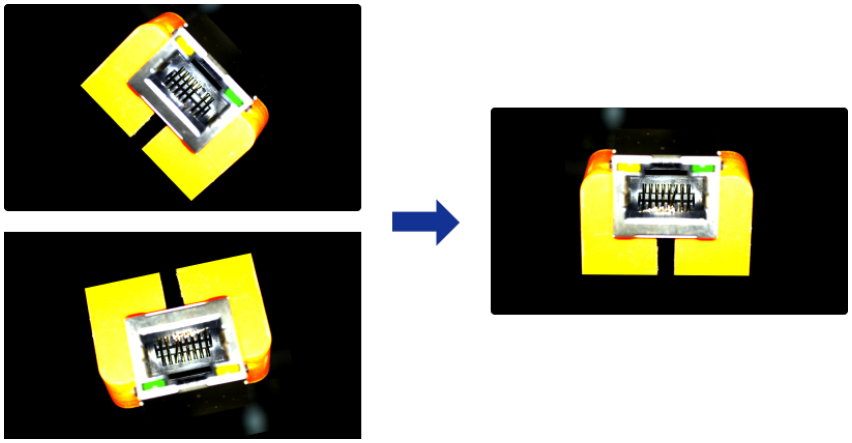
2.5. Train a Fast Positioning Model

2.5.1. Introduction

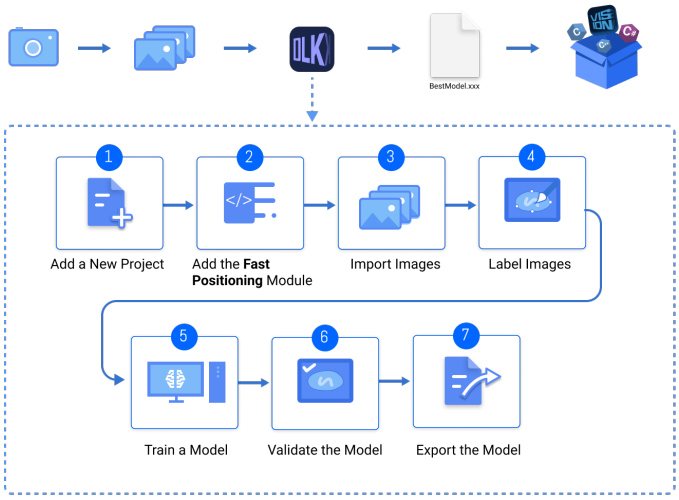
This module is used to detect objects in images and rotate the images to a specified orientation and a specified position.

Application Scenarios

Electronics manufacturing: Detect the electronic components with different placement orientations in the images and adjust the orientations to a specified one.



General Workflow

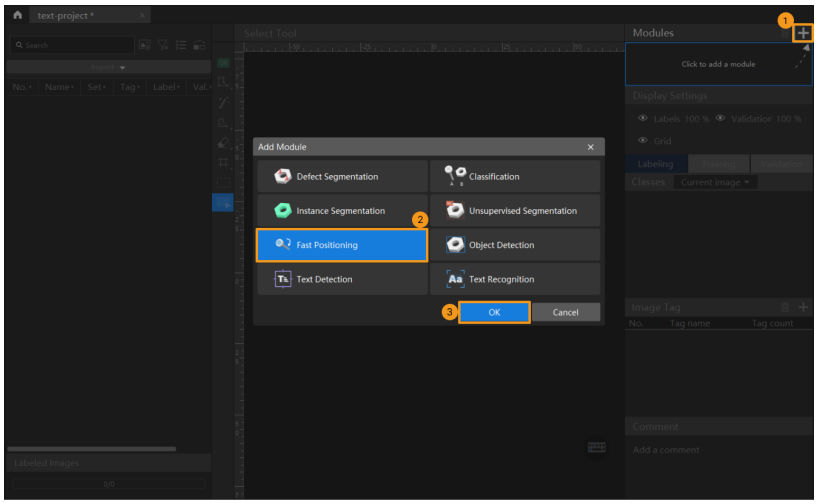


2.5.2. Use the Fast Positioning Module

Please click here to download an image dataset of connectors. In this section, we will use a Fast Positioning module and train a model to rotate the connectors in the images to a uniform orientation.

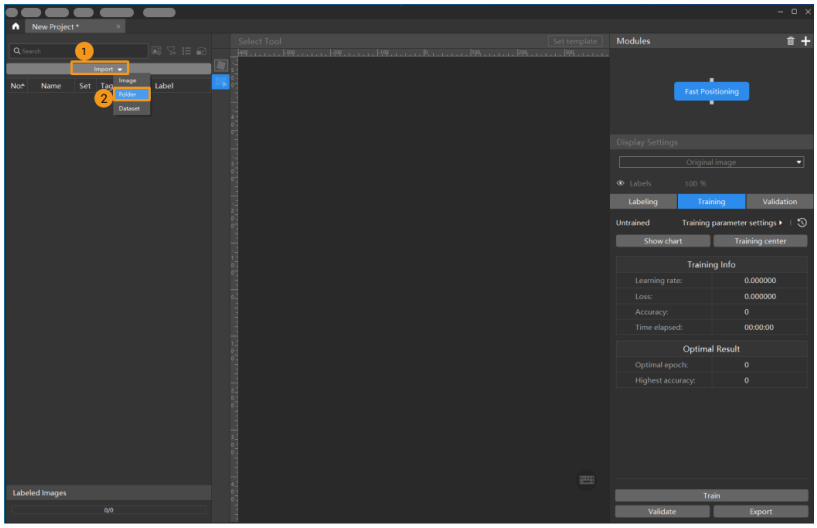
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Fast Positioning module**: Click **[New Project]** in the interface, name the project, and select a directory to save the project. Click + in the upper right corner of the Modules panel, select Fast Positioning, and click **[OK]**.

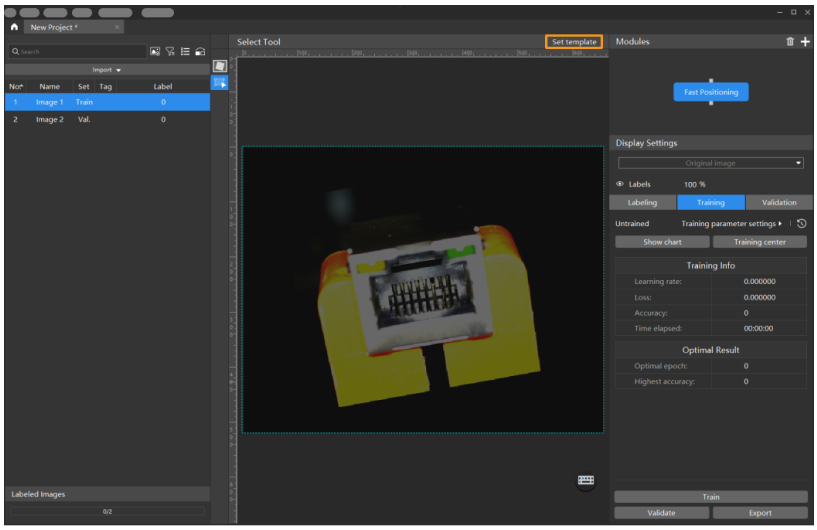


2. **Import images**: Click Import › Folder and select the downloaded image dataset of connectors.

© 2023 Mech-Mind Robotics Technologies Ltd.

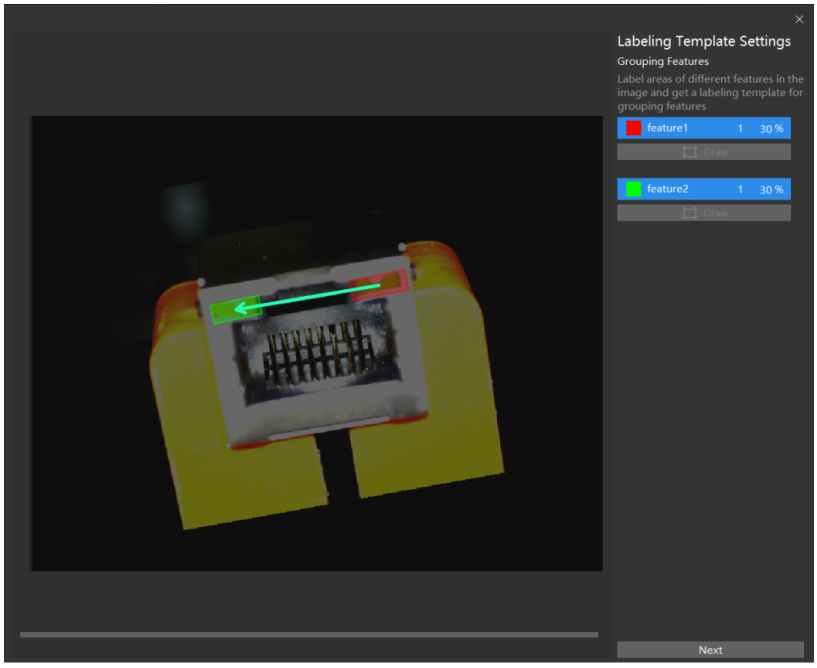


3. **Set a template**: Click **[Set template]**.

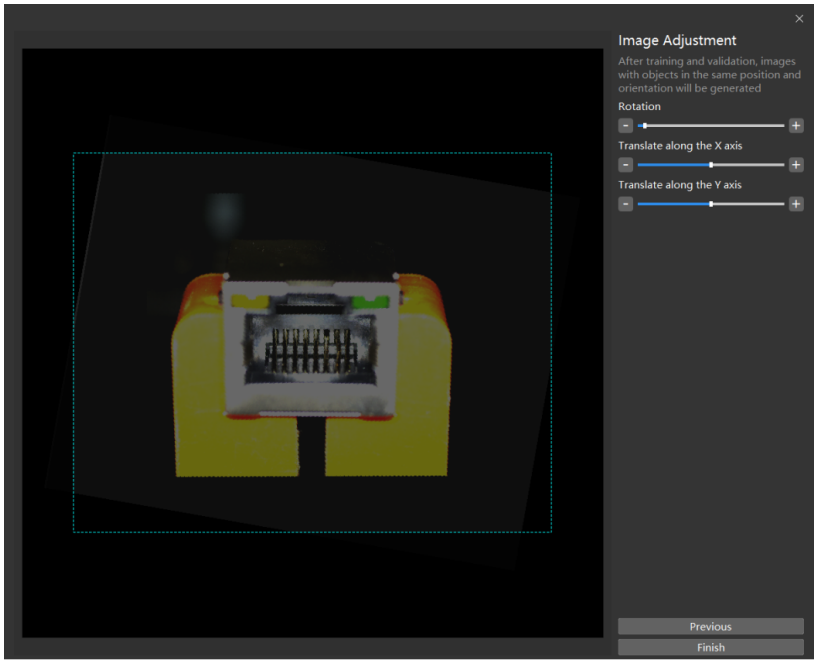


Click **[Draw]** under **feature1** to select the first feature and then click **[Draw]** under **feature2** to select the second feature. Select ⟳ of the feature frame to adjust the expected image orientation. Click **[Next]** upon the setting.

© 2023 Mech-Mind Robotics Technologies Ltd.

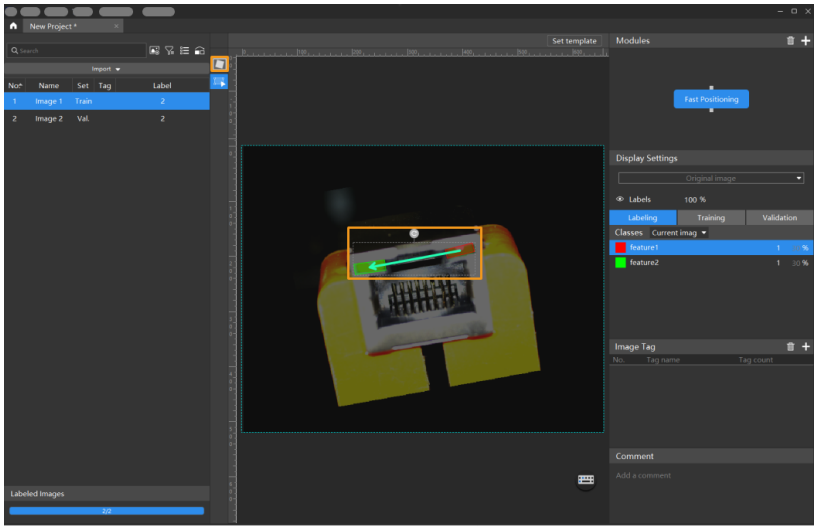


Drag the slider or click **[−]** and **[+]** to adjust the image to an expected orientation and position, and click **[Finish]** to confirm settings.

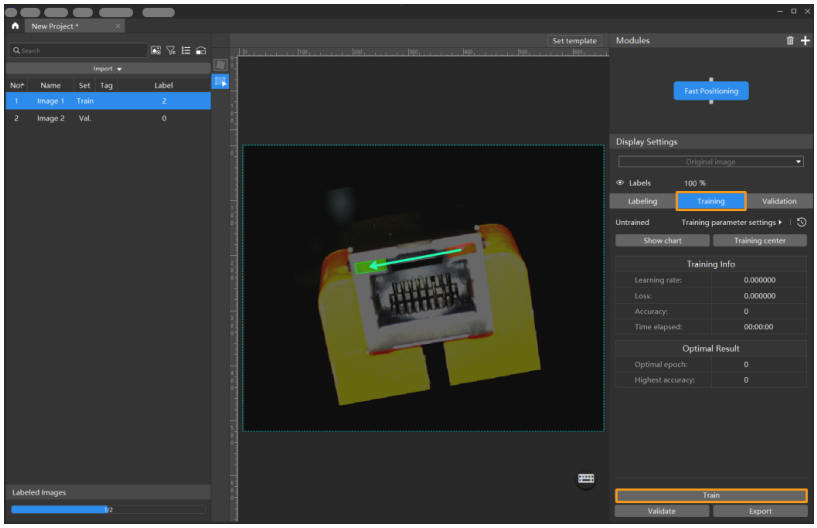


4. **Modify the labeling**: Click and then click Set template. You can translate and rotate the feature frames to cover both features. Repeat the operations to label all images. Click here to view how to use labeling tools.

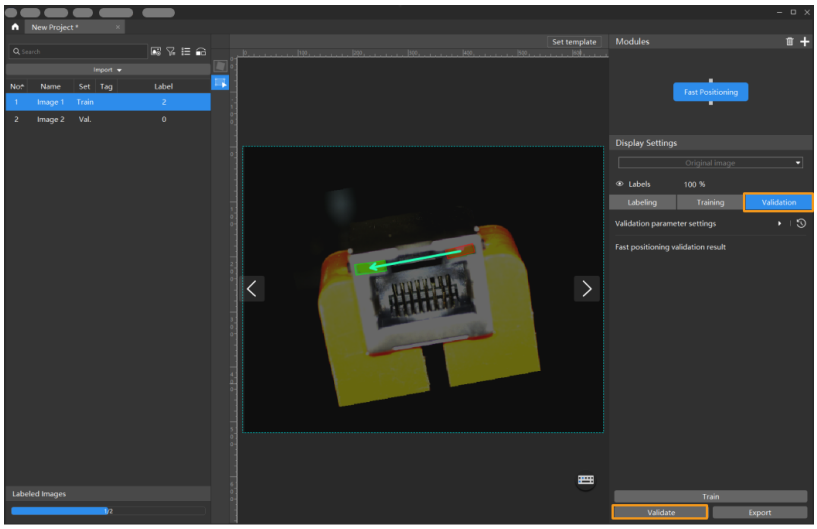
© 2023 Mech-Mind Robotics Technologies Ltd.



5. **Train the model**: Switch to the parameter bar of **Training** and click **[Train]**.



6. **Validate the model**: After the training is completed, click **[Validate]** to validate the model and check the results.

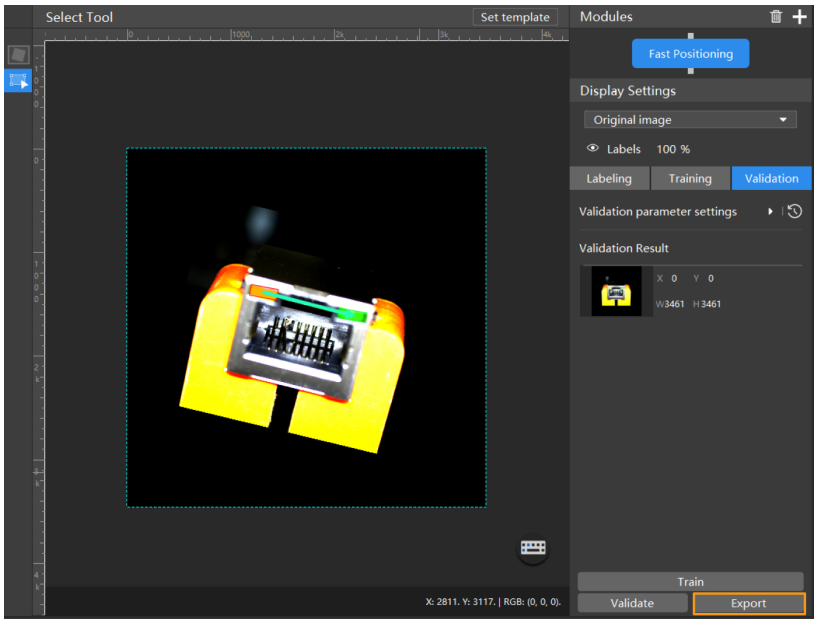


After validation, you can view the fast positioning results of the selected image in the

© 2023 Mech-Mind Robotics Technologies Ltd.

parameter bar of **Validation**. You can also click **[▼]** on the right side of **Original image** to switch to the **Inference Result**. In this way, the original image is switched to the image that has undergone pose correction.

7. **Export the model**: Click **[Export]** and select a directory to save the trained model.

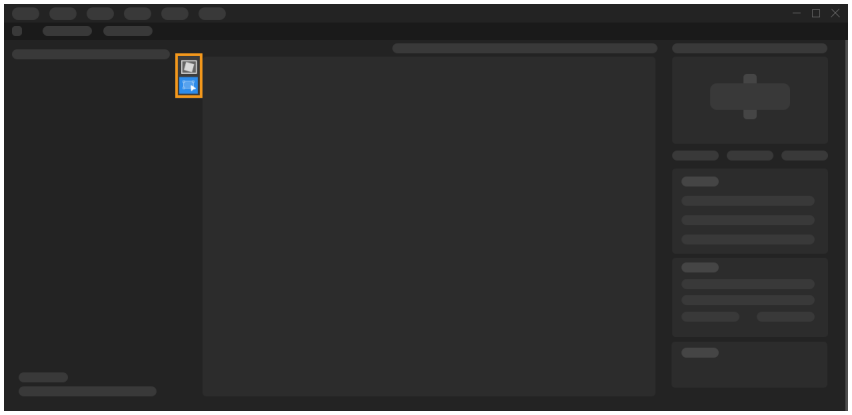


The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.5.3. Introduction to Labeling Tools

You can use labeling tools to label the images and hence provide data for deep learning training.

The yellow frame in the image below shows the labeling tools.



There are two labeling tools in the Fast Positioning module, i.e., **Quick Template Tool** and **Select Tool**.

Quick Template Tool

By the **Quick Template Tool**, you can apply the labeling template to the current image. Click (or press F on the keyboard), click the image at any point, and then use the **Select Tool** to adjust the template to cover the features of the current image.

© 2023 Mech-Mind Robotics Technologies Ltd.

Select Tool

You can use the **Select Tool** to select and adjust the labels.

1. Click (or press S on the keyboard).
2. Move the cursor on the labeling interface and then click the labels.
   - Click the labels and then hold the left mouth button to drag them for a position change.
   - Click and then hold the left mouth button to drag it for a change in label orientations.

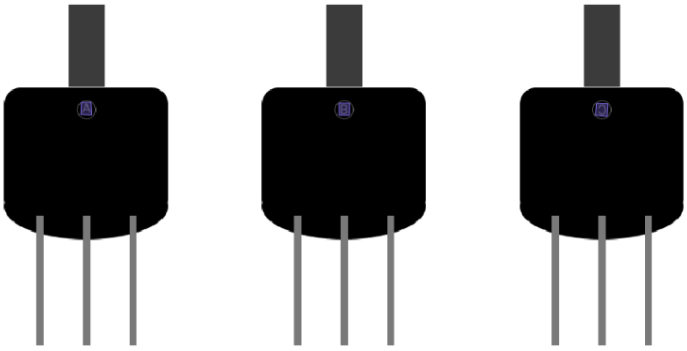
2.6. Train a Text Detection Model

2.6.1. Introduction

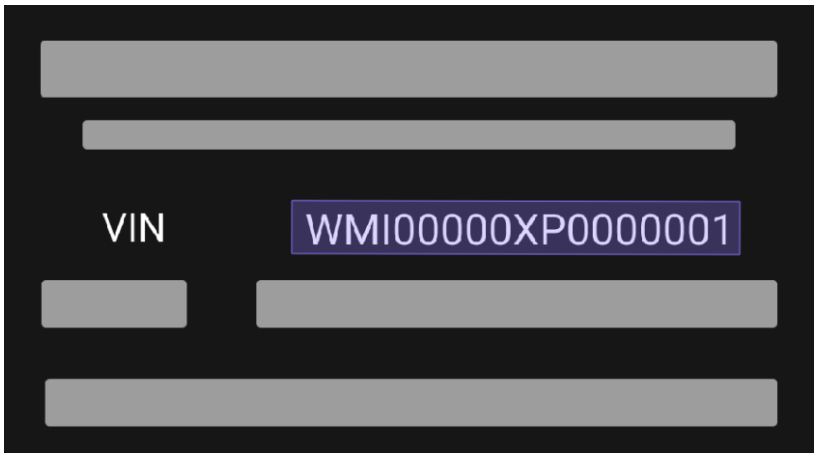
The text detection algorithm can be used to detect the text area in an image. It is usually used with the text recognition algorithm.

Application Scenarios

- **Consumer electronics industry**: This algorithm can detect the small characters on electronic parts. It works as well when the characters are embossed or debossed or are similar to the background in color.



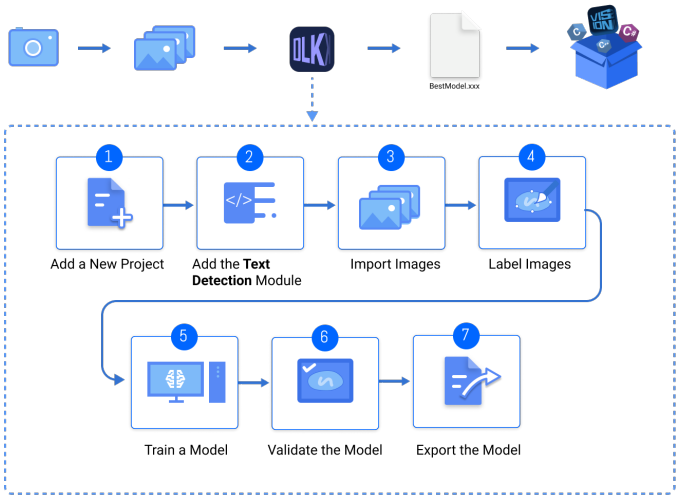
- **Automotive industry**: This algorithm can detect vehicle-related data, such as vehicle identification number (VIN) and the DOT code on tires, against a complex background. It can work when the characters are embossed or debossed or are similar to the background in color. For characters covered with some oil or water, it works as well.



© 2023 Mech-Mind Robotics Technologies Ltd.

- **Packaging industry**: This algorithm can detect characters displayed on the packaging, such as the serial number and the lot number. For ring labels or wrinkled or worn labels, it works as well.

General Workflow



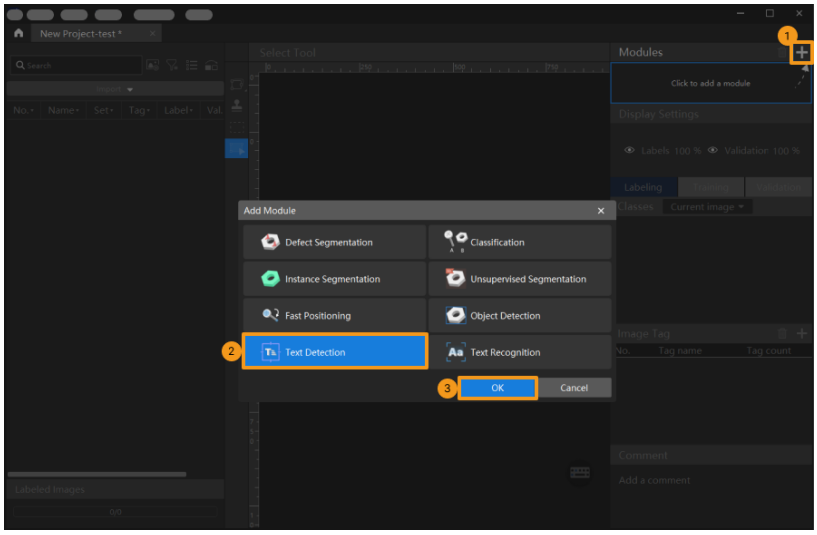
2.6.2. Use the Text Detection Module

Taking an image dataset of identification numbers (download) as an example, this section will show you how to use the **Text Detection** module to detect the text area in an image. The **Text Detection** module can be followed by a **Text Recognition** module to output the recognized numbers, letters, or some special symbols.

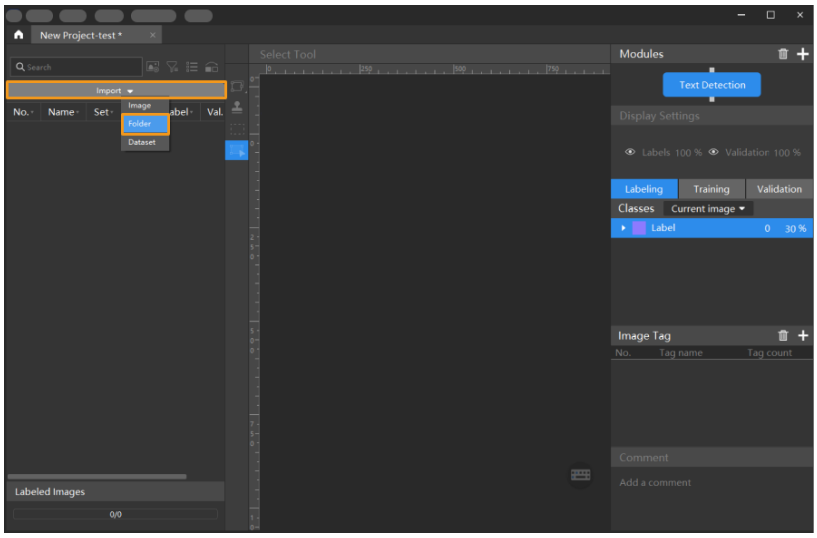
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Text Detection module**: Click **[New Project]** after you opened the software, name the project, and select a directory to save the project. Then, click + in the upper-right corner and add the **Text Detection** module.

© 2023 Mech-Mind Robotics Technologies Ltd.

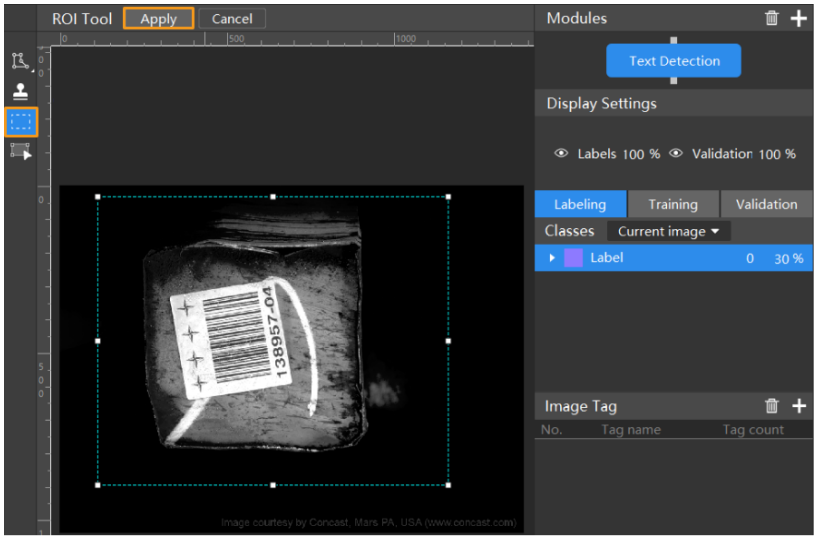


2. **Import the image dataset**: Extract the downloaded zip file. Select Import › Folder on the left of the interface to import the image dataset.

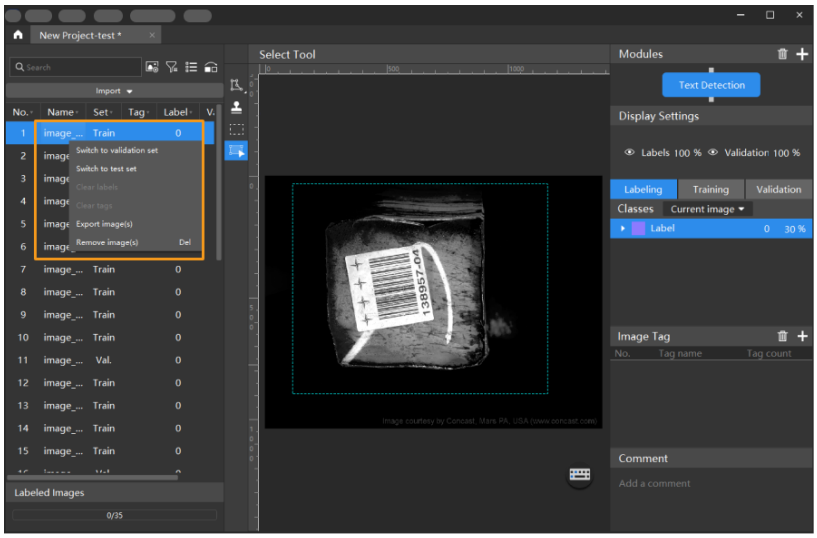


3. **Select an ROI**: Click the ROI Tool button and adjust the frame to set an ROI that covers the text areas of all images. Then, click **[Apply]** on the top of the selection region to save the setting. Setting the ROI can avoid interferences from the background.

© 2023 Mech-Mind Robotics Technologies Ltd.

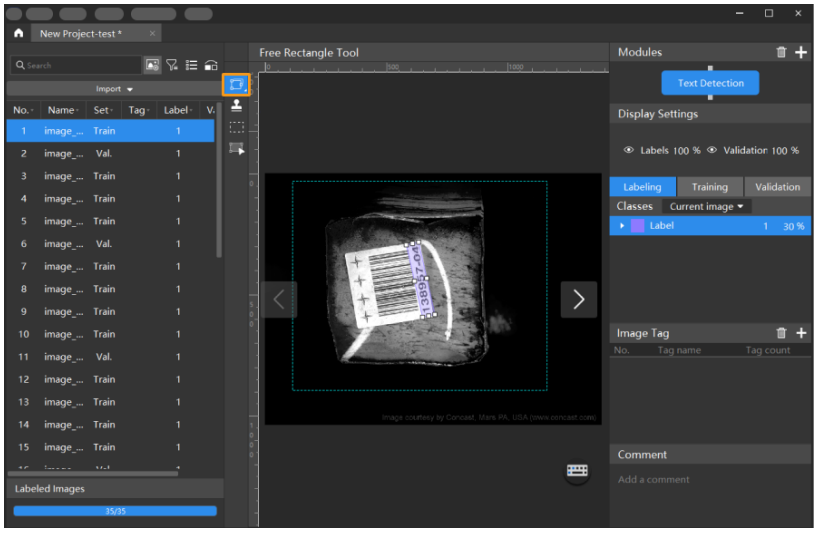


4. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. You can click and drag the slider to adjust the proportion. Please make sure that both the training set and validation set include all kinds of text areas to be detected. If the default training set and validation set cannot meet this requirement, please right-click the name of the image and then click **Switch to training set** or **Switch to validation set** to adjust the set to which the image belongs.

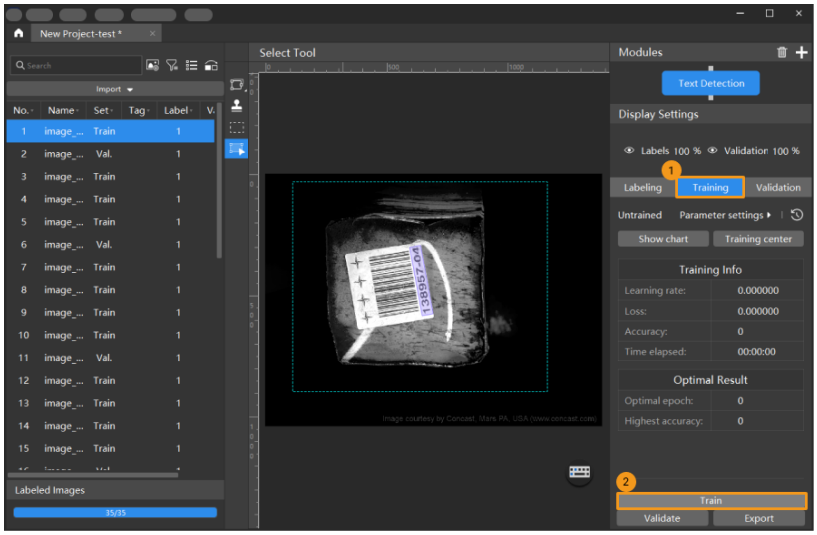


5. **Label the images**: Select an appropriate tool from the toolbar to label the images.

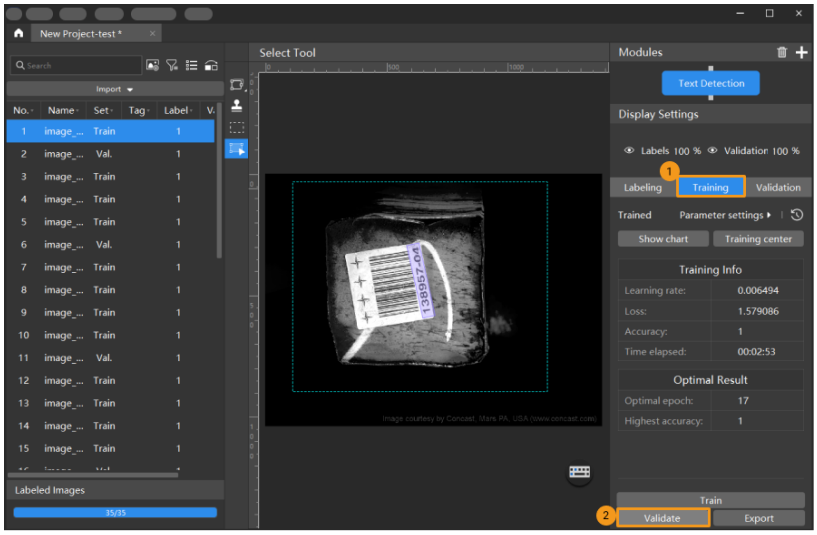
© 2023 Mech-Mind Robotics Technologies Ltd.



6. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.



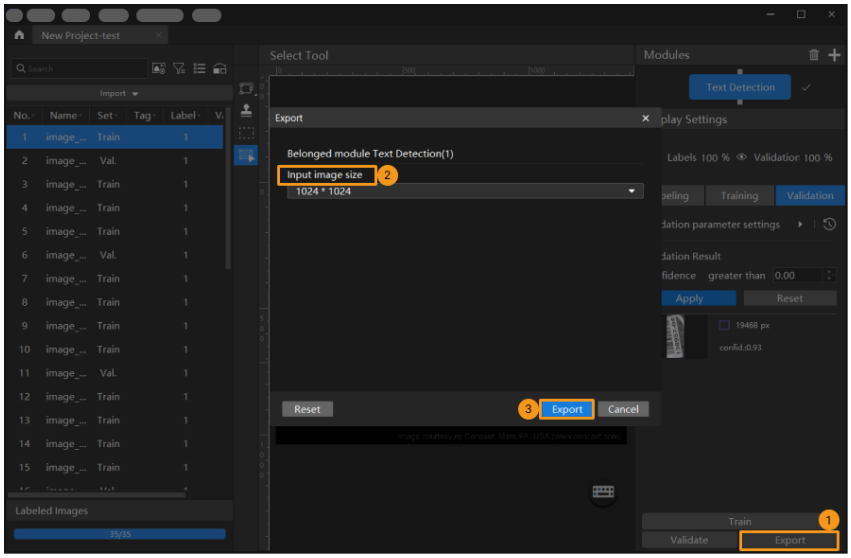
7. **Validate the Model**: After the training is completed, click **[Validate]** to validate the model and check the results.



© 2023 Mech-Mind Robotics Technologies Ltd.

8. **Export the model**: Click **[Export]**. Then, set **Input image size** in the pop-up dialog box, click **[Export]**, and select a directory to save the exported model.

   By default, the **Input image size** is consistent with that set in the **Validation Parameters** dialog box. In validation, if the inference speed and accuracy are up to standard, you can keep the default setting.



The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.6.3. Introduction to Labeling Tools

Free Rectangle Tool

The **Free Rectangle Tool** can be used to draw a rectangle around the text area. Therefore, it is recommended to select this tool for rectangular text areas.

[free rectangle tool] | *dlk-software-manual/2.5.0/algorithm-modules/_images/text-detection/free-rectangle-tool.gif*

1. Click (or press D on the keyboard).
2. Move the crosshair cursor in the selection region and left-click to select the starting point of the area you want to outline. Then, move the cursor and left-click again to select the second point to define the range you want to outline. At this point, a straight line will be created, forming one side of the rectangular selection frame.
3. Moving the cursor to either side of the straight line will allow you to create a rectangle. Click the left mouse button to complete the labeling.

   During the process, you can click the right mouse button to cancel the labeling.

Polygon Tool

The **Polygon Tool** can draw polygon labels with more vertices, which is suitable for text areas of complex shapes or those near the image edges.

1. Click (or press P on the keyboard).
2. Click the first position (vertex) in the selection region, then click the second one, third one, etc., to draw the labels, and right-click to finish. (At least three vertices are required.)

© 2023 Mech-Mind Robotics Technologies Ltd.

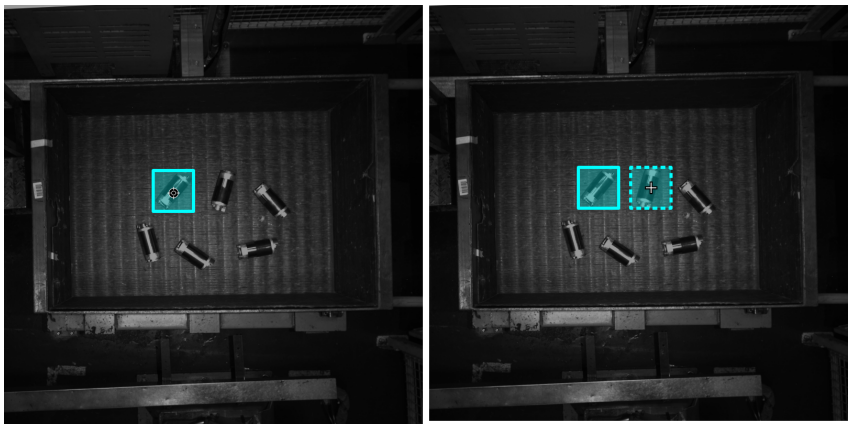
After labeling, use the **Select Tool** to select a label and adjust the label by the following methods.

- Click the label edges to increase the number of vertices.
- Right-click a vertex to delete it.
- Long press the left mouse button and drag the vertex in any direction to modify the label shape.

Template Tool

You can use the **Template Tool** to set an existing selection as a template. After setting, you can use this template to rapidly label contours and objects with the same pose.

It is suitable for scenarios where there are multiple neatly arranged objects of the same type in an image, which can improve labeling efficiency.



1. Click (or press C on the keyboard).
2. Click the region that needs to be set as the template.
3. Move the template to the to-be-selected object, adjust the angle of the template to make it fit the object, and then click it.
   - Coarse adjustment: Press and hold the `Shift` key and then scroll the mouse wheel.
   - Fine adjustment: Adjust the **Rotation angle** parameter.

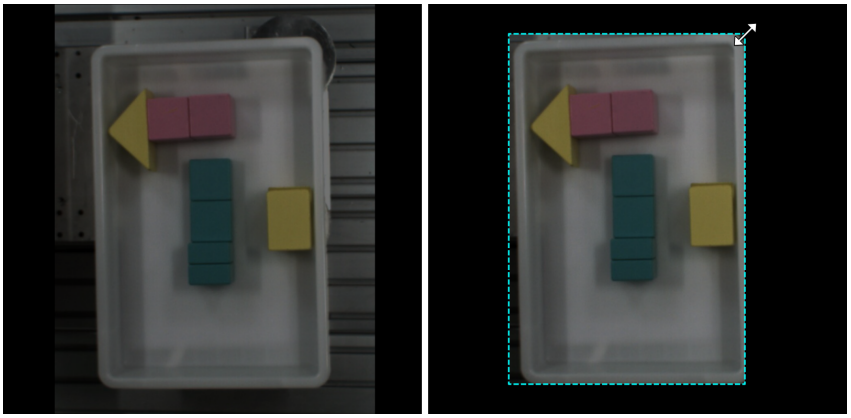
During labeling, press and hold the `Ctrl` key and click the selection to switch the template. You can also click **[Replace template]** and then click the selection to achieve the same purpose.

ROI Tool

You can use the **ROI Tool** to set the region of interest.

Setting the ROI can avoid interferences from the background.

© 2023 Mech-Mind Robotics Technologies Ltd.

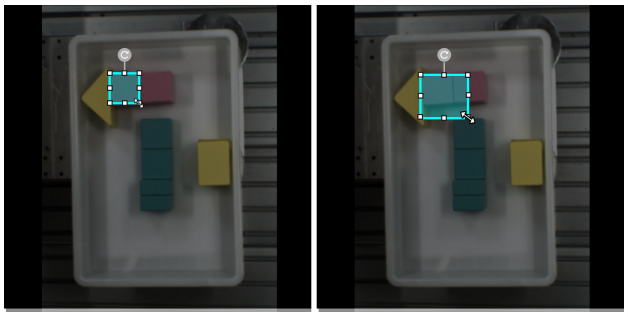
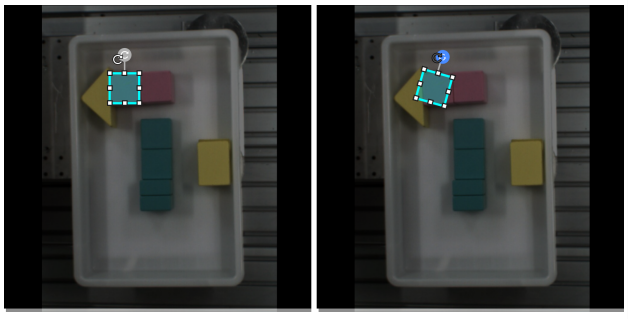


1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.
3. Click **[Apply]** in the upper left corner of the selection region.

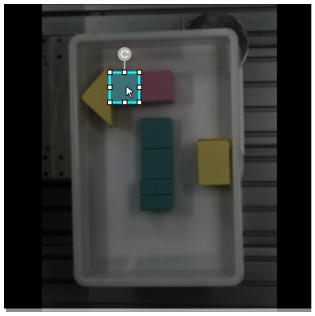
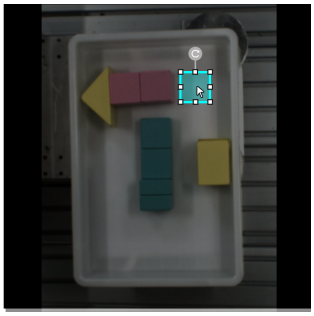
Select Tool

You can use the **Select Tool** to select, move, and adjust the selections.

1. Click (or press S on the keyboard).
2. Move the cursor in the selection region and then click the selection to be processed. Select multiple selections by pressing and holding the `Ctrl` key.

Adjust size	
Rotation	

© 2023 Mech-Mind Robotics Technologies Ltd.

Move	 

2.6.4. Train a High-Quality Model

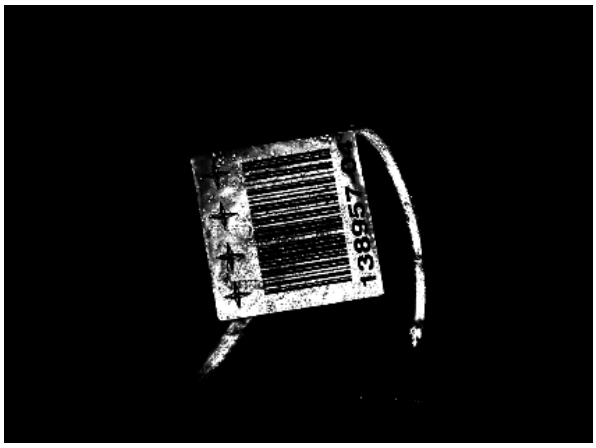
This section introduces the factors that most affect the model quality and how to train a high-quality text detection model.

Ensure Image Quality

Avoid **overexposure, dimming, blur, occlusion**, etc. These conditions can lead to the loss of features that the deep learning model relies on, which will affect the model training effect.

Poor images: The text areas are partially occluded, which may affect the detection result and the subsequent recognition result.	
Suggestion: Make sure that the text areas are clear and complete.	

© 2023 Mech-Mind Robotics Technologies Ltd.

Poor image: overexposure	Poor image: underexposure
	

Suggestion: You can avoid overexposure by methods such as shading and avoid dimming by methods such as supplemental light.

Ensure Data Quality

The **Text Detection** module can be used to detect the text area in an image. In practical applications, the text to be detected only takes a small part of the image. Thus, the camera and lighting conditions should stay stable once adjusted during data collection to avoid data deviations and ensure model performance.

Data Selection

Select Images with Different Text Orientations

Select diverse images. In practical applications, the texts in images may be horizontal (0° and 180°), vertical (90°), or tilted (other angles). If only images with horizontal or vertical texts are applied for training, the trained model can hardly detect tilted texts in images.

If some situations are not included in the dataset, the model will not go through adequate learning. As a result, the trained model cannot effectively detect the text areas given such conditions. In this case, data on such conditions must be collected and added to reduce the errors.

© 2023 Mech-Mind Robotics Technologies Ltd.

Select Appropriate Data

- Control the image quantity of training set

  When you first use the **Text Detection** module, it is recommended to use 30 to 50 images as a training set for model training. If the validation results are not up to standard, you can then add more data.

- Balance data proportion

  In the training set, the images with **different text orientations** should be balanced. Using diverse data can help improve model performance.

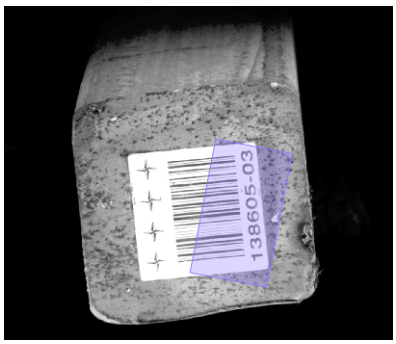
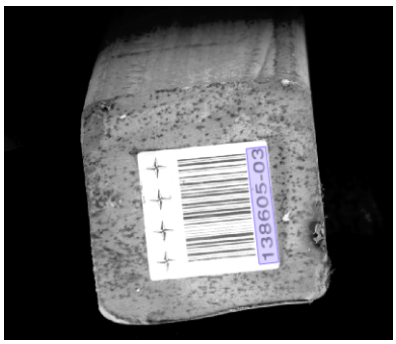
- Reduce inadequate data

  High-quality images can ensure better model training. Poor images or irrelevant images may do damage to model training.

It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later and will slow down the training process.

Ensure Labeling Quality

Ensure **accuracy**. During labeling, the selection frame should be placed as close to the edges of the text area as possible to minimize interference. Incomplete selections or overly large selection frames should be avoided.

Bad labeling		Good labeling
		

2.7. Train a Text Recognition Model

2.7.1. Introduction

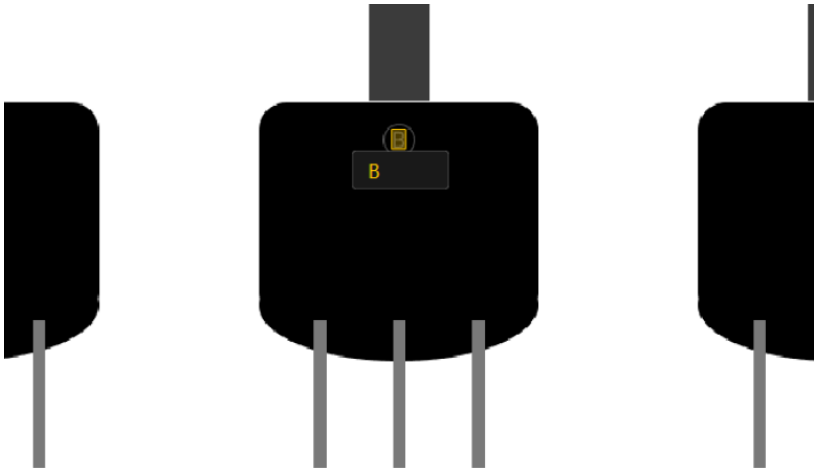
The text recognition algorithm can be used to recognize characters in images. The recognition of the following characters is supported:

- **Letters**: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
- **Numbers**: 0123456789
- **Special symbols**: !"#$%&'()*+,-./:;>=<?@[\]_`~

© 2023 Mech-Mind Robotics Technologies Ltd.

Application Scenarios

- **Customer electronics industry**: This algorithm can recognize the small characters on electronic parts. It works as well when the characters are embossed or debossed or are similar to the background in color.

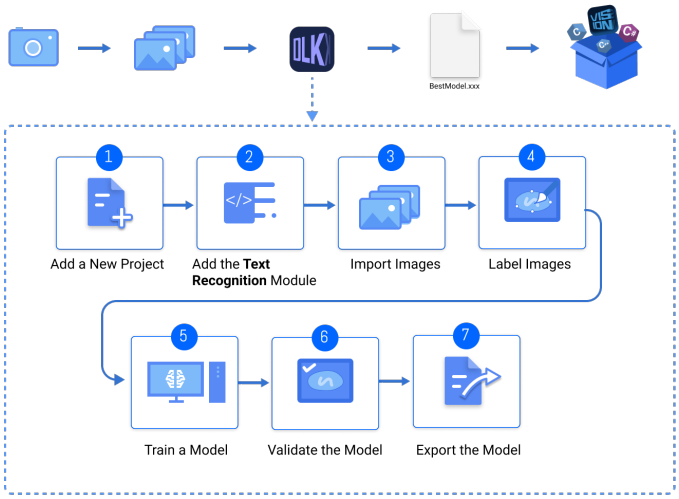


- **Automotive industry**: This algorithm can recognize vehicle-related data, such as vehicle identification number (VIN) and the DOT code on tires, against a complex background. It can work when the characters are embossed or debossed or are similar to the background in color. For characters covered with some oil or water, it works as well.

- **Packaging industry**: This algorithm can recognize characters displayed on the packaging, such as the serial number and the lot number. For ring labels or wrinkled or worn labels, it works as well.

© 2023 Mech-Mind Robotics Technologies Ltd.

General Workflow

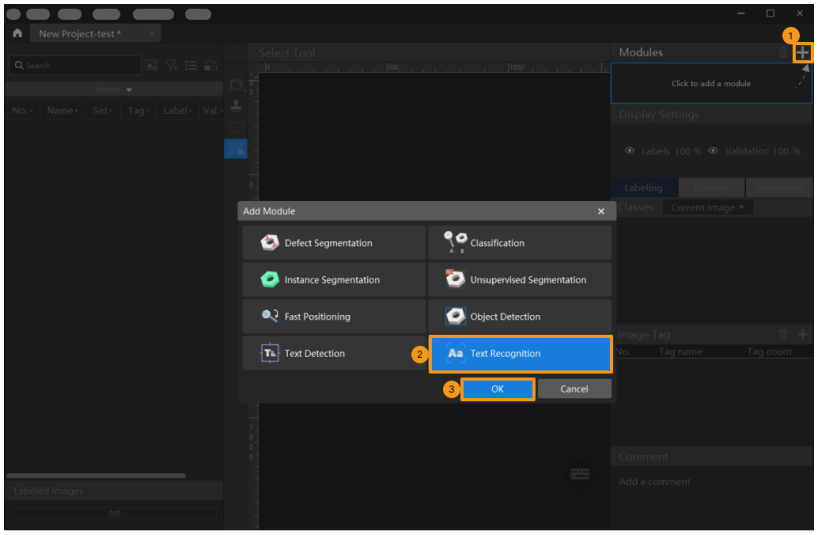


2.7.2. Use the Text Recognition Module

Taking an image dataset of numbers (download) as an example, this section will show you how to use the **Text Recognition** module to recognize and output the characters in an image. The characters that can be recognized include numbers, letters, and some special symbols.

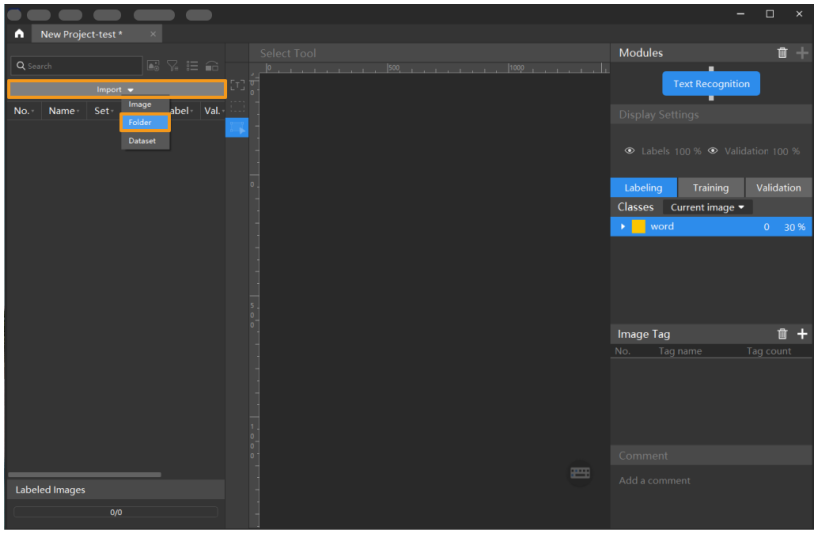
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Text Detection module**: Click **[New Project]** after you opened the software, name the project, and select a directory to save the project. Then, click + in the upper-right corner and add the **Text Recognition** module.



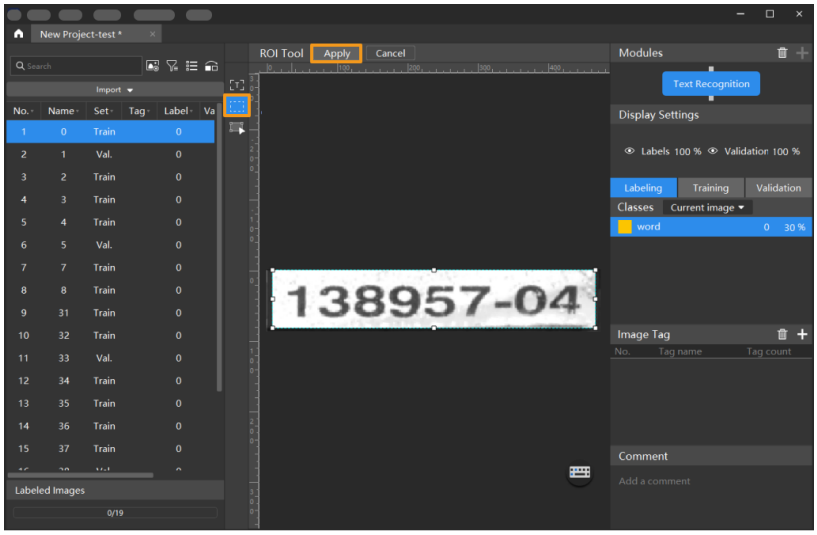
2. **Import the image dataset**: Extract the downloaded zip file. Select Import › Folder on the left of the interface to import the image dataset.

© 2023 Mech-Mind Robotics Technologies Ltd.



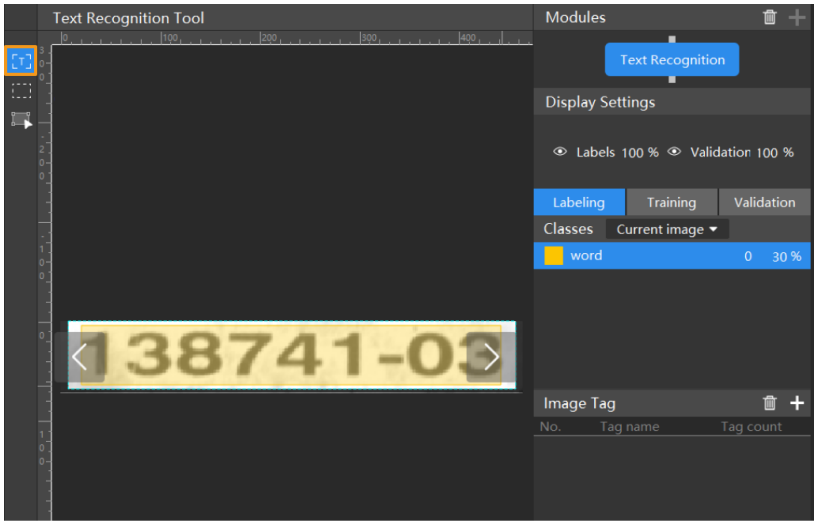
For better model performance, the Text Recognition module is often preceded by another module. In this case, you can select Import › Import from previous module to import data.

3. **Select an ROI**: Click the ROI Tool button and adjust the frame to set an ROI that covers the text areas of all images. Then, click **[Apply]** on the top of the selection region to save the setting. Setting the ROI can avoid interferences from the background.

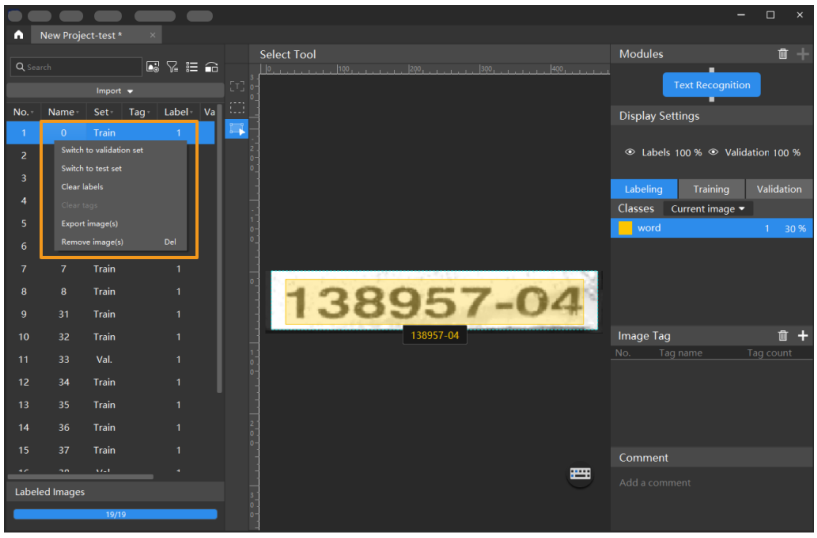


4. **Label the images**: Select the **Text Recognition Tool** from the toolbar to label the images.

© 2023 Mech-Mind Robotics Technologies Ltd.

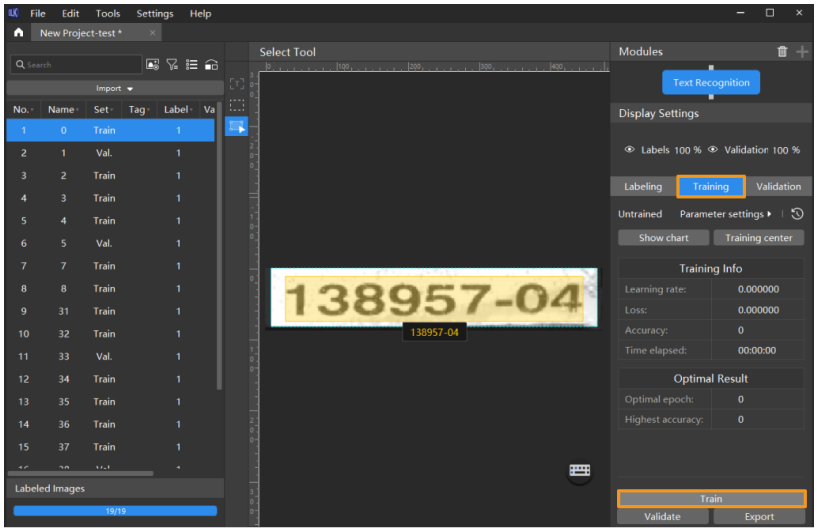


5. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. You can click and drag the slider to adjust the proportion. Please make sure that both the training set and validation set include all kinds of texts to be detected. If the default training set and validation set cannot meet this requirement, please right-click the name of the image and then click **Switch to training set** or **Switch to validation set** to adjust the set to which the image belongs.

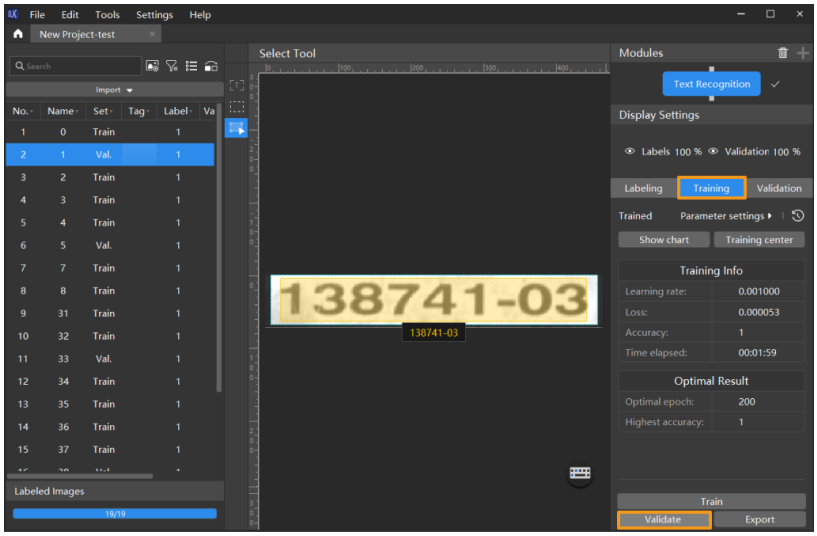


6. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.

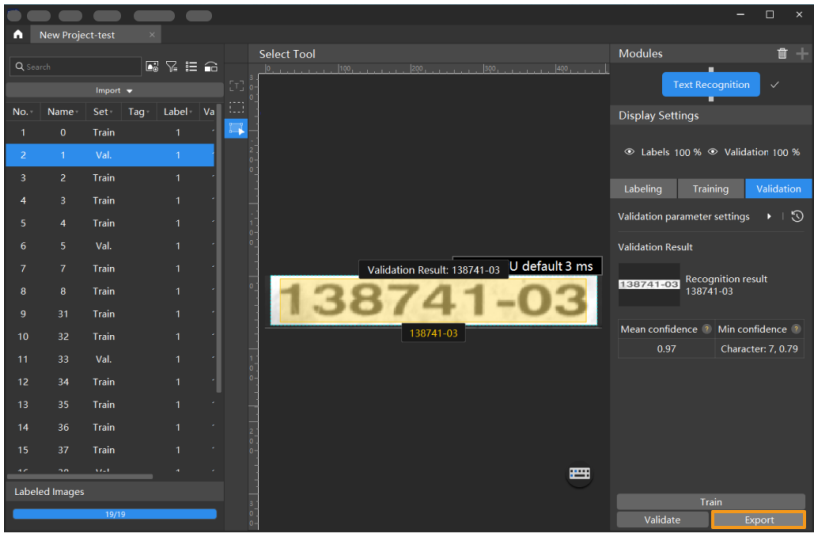
© 2023 Mech-Mind Robotics Technologies Ltd.



7. **Validate the Model**: After the training is completed, click **[Validate]** to validate the model and check the results.



8. **Export the model**: Click **[Export]** and select a directory to save the trained model.



© 2023 Mech-Mind Robotics Technologies Ltd.

The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.7.3. Introduction to Labeling Tools

Text Recognition Tool

The **Text Recognition Tool** can be used to define the text recognition range and automatically generate the recognition result.

[text recognition tool] | *dlk-software-manual/2.5.0/algorithm-modules/_images/text-recognition/text-recognition-tool.gif*

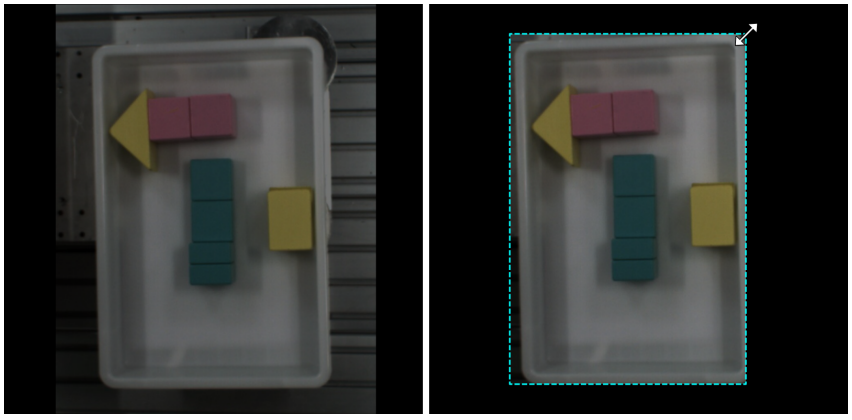
1. Click (or press G on the keyboard).
2. Long press the left mouse button in the selection region and move the cursor to draw a rectangle. Release the mouse button to confirm the selection.
3. The automatically recognized characters will be displayed right under the selection frame. If the recognition result is correct, click **[OK]** to confirm it; otherwise, please correct the result first and then click the **OK** button.

> If **Unconfirmed** is displayed right under the selection frame, click the frame and then confirm the recognition result.

ROI Tool

You can use the **ROI Tool** to set the region of interest.

Setting the ROI can avoid interferences from the background.



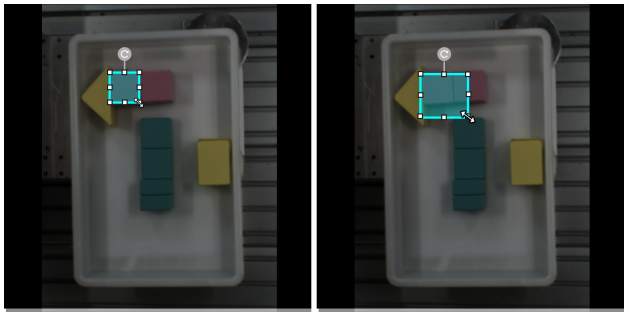
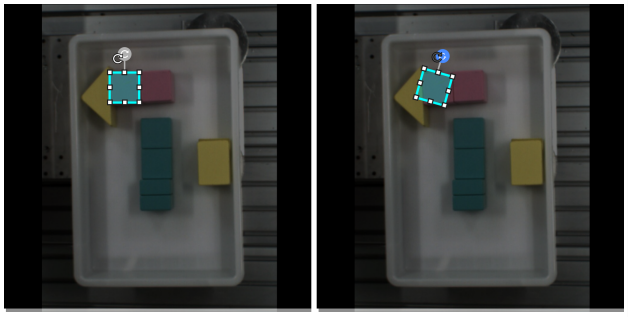
1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.
3. Click **[Apply]** in the upper left corner of the selection region.

Select Tool

You can use the **Select Tool** to select, move, and adjust the selections.

1. Click (or press S on the keyboard).
2. Move the cursor in the selection region and then click the selection to be processed. Select multiple selections by pressing and holding the `Ctrl` key.

© 2023 Mech-Mind Robotics Technologies Ltd.

Adjust size	
Rotation	
Move	

2.7.4. Train a High-Quality Model

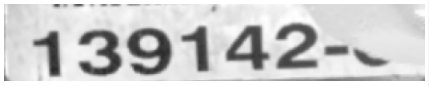
This section introduces the factors that most affect the model quality and how to train a high-quality instance segmentation model.

Ensure Image Quality

Single Module

Please pay attention to the following suggestions so as to train a high-quality text recognition model:

- Avoid **overexposure, dimming, blur, occlusion**, etc. These conditions can lead to the loss of features that the deep learning model relies on, which will affect the model training effect.

Poor images: The texts in the images are partially occluded, which affects recognition performance.	
	
Suggestion: Make sure that the text areas are clear and complete.	

© 2023 Mech-Mind Robotics Technologies Ltd.

Poor image: overexposure	Poor image: underexposure
Suggestion: You can avoid overexposure by methods such as shading and avoid dimming by methods such as supplemental light.	

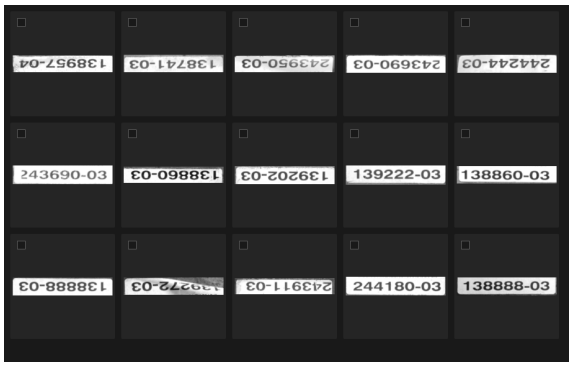
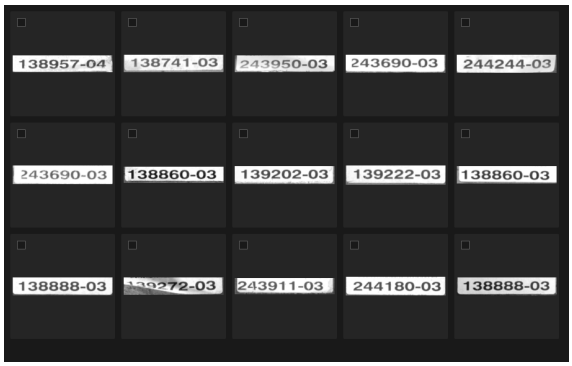
- The texts in the imported images should be oriented toward the positive direction (0°). If the text orientations vary greatly, it is recommended to add a **Text Detection** module before this module. When importing data, select Import › Import from previous module and make sure the **Rectify image(s)** function is enabled to rectify images to 0°.

Cascaded Modules

The **Text Detection** or **Object Detection** module can precede the **Text Recognition** module for better recognition results. Note that the **Text Recognition** module cannot be followed by any other modules.

- If a **Text Detection** module precedes it, select Import › Import from previous module when importing data and make sure the **Rectify image(s)** function is enabled to rectify images to 0°.

Typically, the **Rectify image(s)** function reliably carries out its designated task. However, occasionally, few images oriented at 0° may inadvertently undergo rectification to 180°. In such situations, it is advisable to exercise discretion in practical applications.

Before rectification	After rectification
	

Ensure Data Quality

The **Text Recognition** module can recognize the characters in the selected text area of an image and automatically generate the recognition result. Manual checking is required for the result. The used data should be as consistent with the actual scenarios as possible for the training of a high-quality model.

Select Appropriate Data

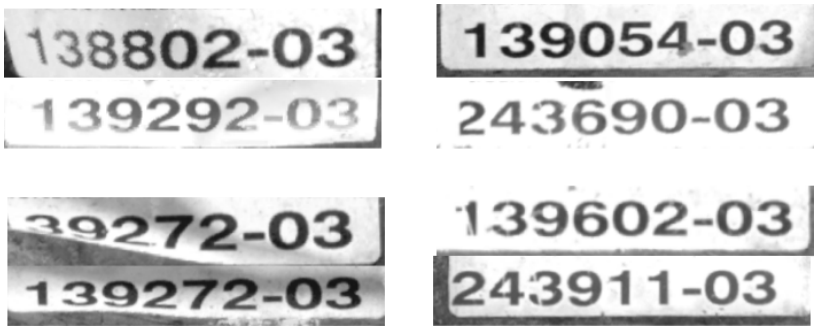
- Control the image quantity of training set

  When you first use the **Text Recognition** module, it is recommended to use 20 to 30 images as a training set for model training. If the validation results are not up to standard, you can then add more data.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Balance data proportion

  In the training set, the images with different text types should be balanced. Using diverse data can help improve model performance.



- Reduce inadequate data

  High-quality images can ensure better model training. Poor images or irrelevant images may do damage to model training.

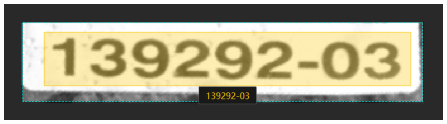
  It is not true that the larger the number of images the better. Adding a large number of inadequate images in the early stage is not conducive to model improvement later and will slow down the training process.

Ensure Labeling Quality

When the **Text Recognition Tool** is used to make a selection, the recognition result will automatically appear right under the selection frame. Manual verification and confirmation are required. Therefore, making a valid selection and confirming a correct recognition result in time are conducive to improving model quality.

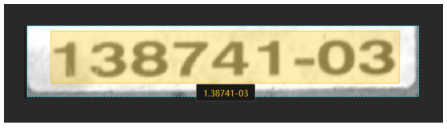
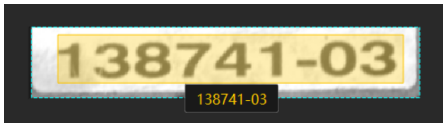
Make Valid Selection

If the target text is not covered completely by the selection frame, the labeling will interfere with model training and thus affect model performance. Therefore, the frame should cover the whole target text.

Incorrect example	Correct example
	

Confirm Correct Recognition Result

It is possible that the automatic recognition result is incorrect. If so, please correct it first and then click the **OK** button. Confirming an incorrect result may detrimentally impact the recognition accuracy of the trained model.

Incorrect example	Correct example
	

© 2023 Mech-Mind Robotics Technologies Ltd.

2.8. Train an Unsupervised Segmentation Model

2.8.1. Introduction

The unsupervised segmentation algorithm can be used to judge whether the image of an object is OK, NG, or Unknown on the basis of set thresholds. Moreover, heat maps are available to show the possible areas with defects.

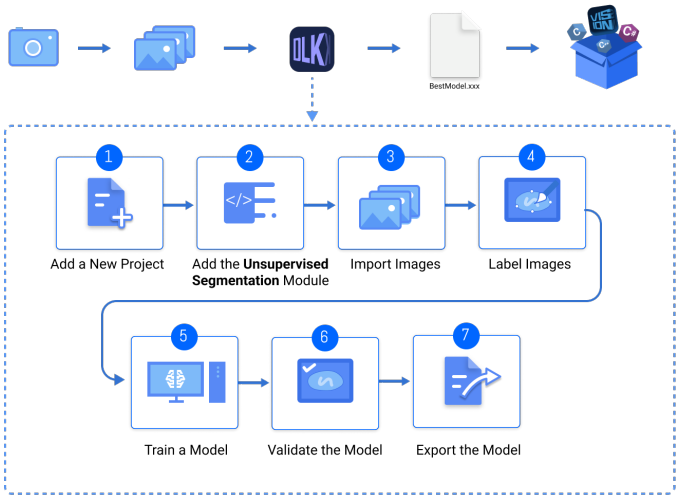
- If the **Defect confidence** of an image is less than the threshold set for OK results, the image will be labeled as OK.
- If the **Defect confidence** of an image is greater than the threshold set for NG results, the image will be labeled as NG.
- If the **Defect confidence** of an image is greater than the threshold set for OK results and less than the threshold set for NG results, the image will be labeled as Unknown.

Application Scenario

Quality inspection: The algorithm is applicable to scenarios where objects have defects of different shapes and size and in different positions but their OK images have small but important differences

General Workflow

© 2023 Mech-Mind Robotics Technologies Ltd.

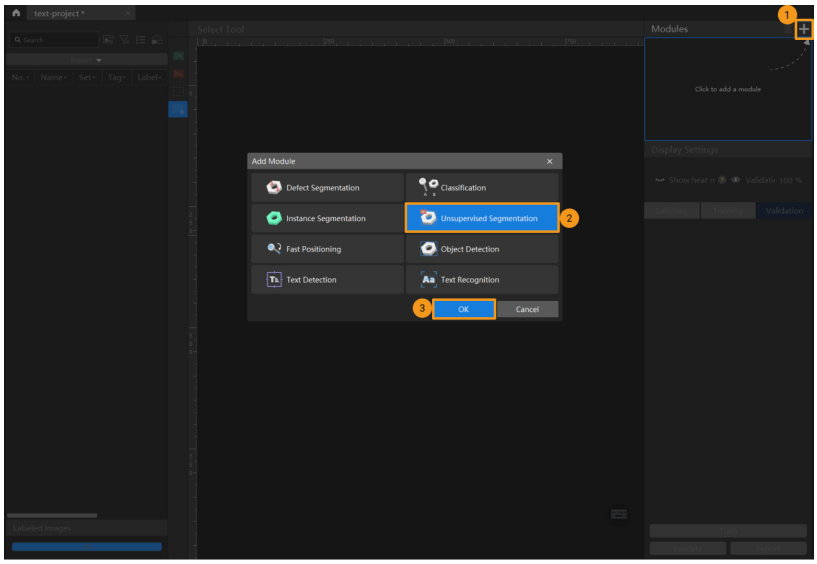


2.8.2. Use the Unsupervised Segmentation Module

Taking the bottom images of bottles (download) as an example, this section introduces you to how to use the **Unsupervised Segmentation** module to judge whether there are defects in an image.

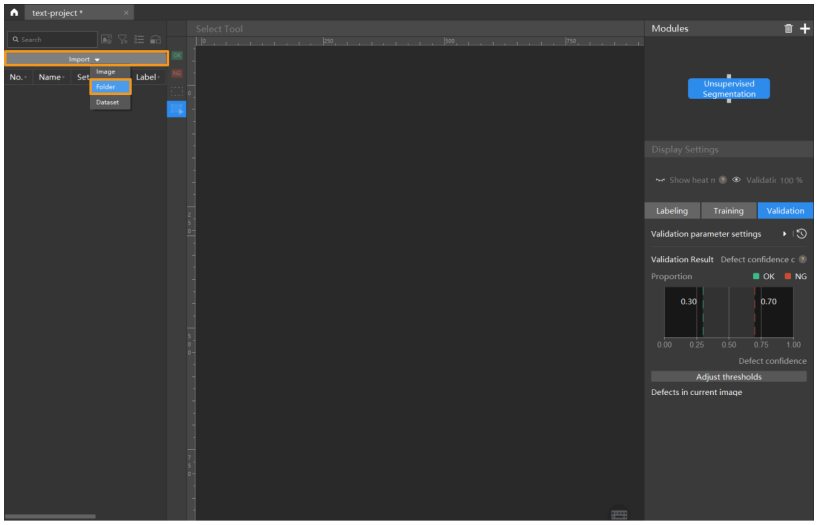
You can also use your own data. The usage process is overall the same, but the labeling part is different.

1. **Create a new project and add the Unsupervised Segmentation module**: Click **[New Project]** after you opened the software, name the project, and select a directory to save the project. Then, click + in the upper-right corner and add the **Unsupervised Segmentation** module.



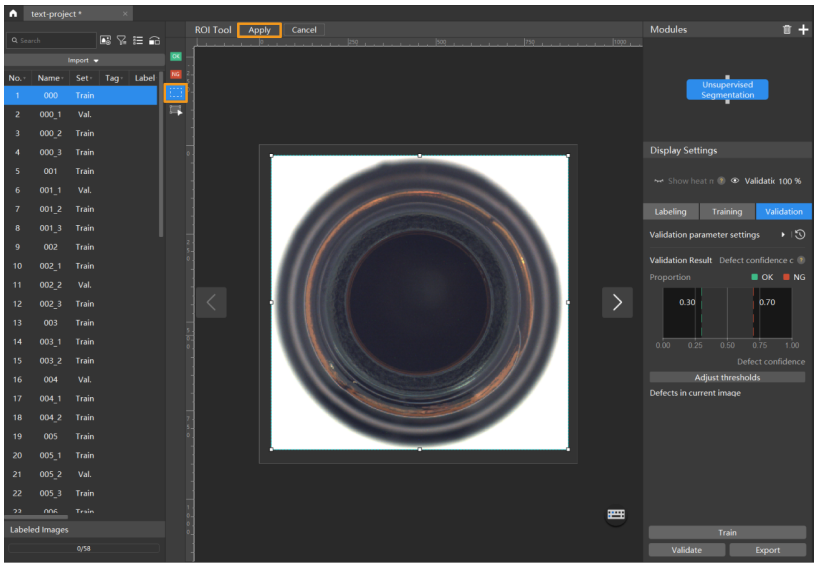
2. **Import the image dataset**: Extract the downloaded zip file. Select Import › Folder on the left of the interface to import the image dataset.

© 2023 Mech-Mind Robotics Technologies Ltd.



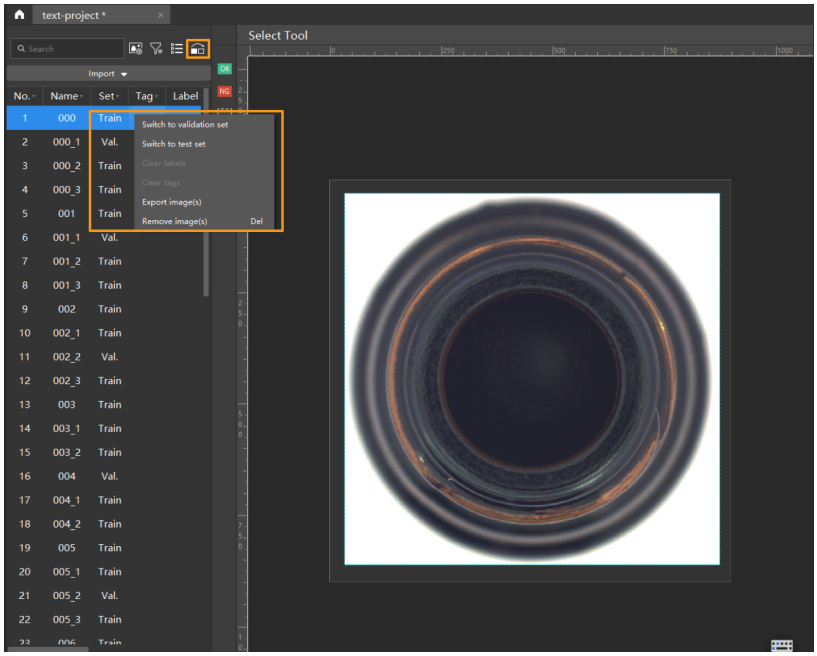
3. **Select an ROI**: Click the ROI Tool button and adjust the frame to set an ROI that covers the target objects in the images. Then, click **[Apply]** on the top of the selection region to save the setting. Setting the ROI can avoid interferences from the background.

The same ROI setting will be applied to all images, so it is necessary to ensure that objects in all images are within the set ROI.

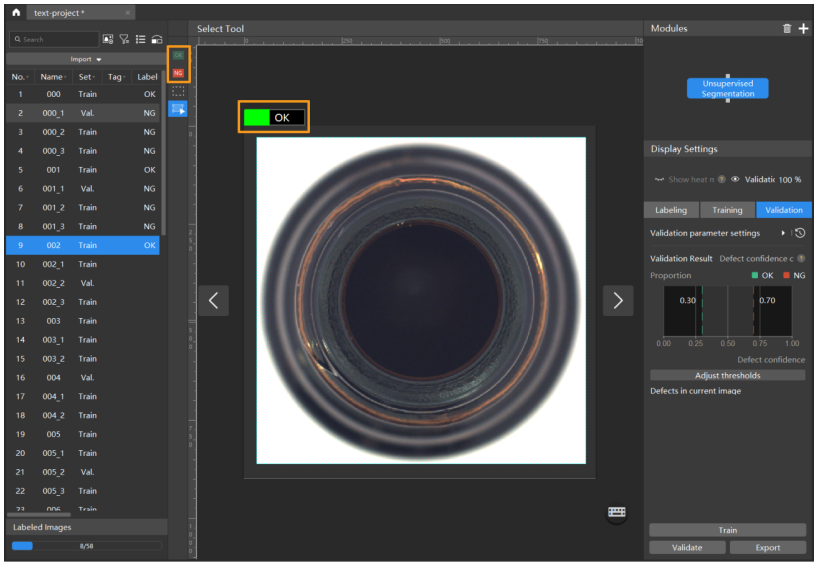


4. **Split the dataset into the training set and validation set**: By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. You can click and drag the slider to adjust the proportion.

© 2023 Mech-Mind Robotics Technologies Ltd.



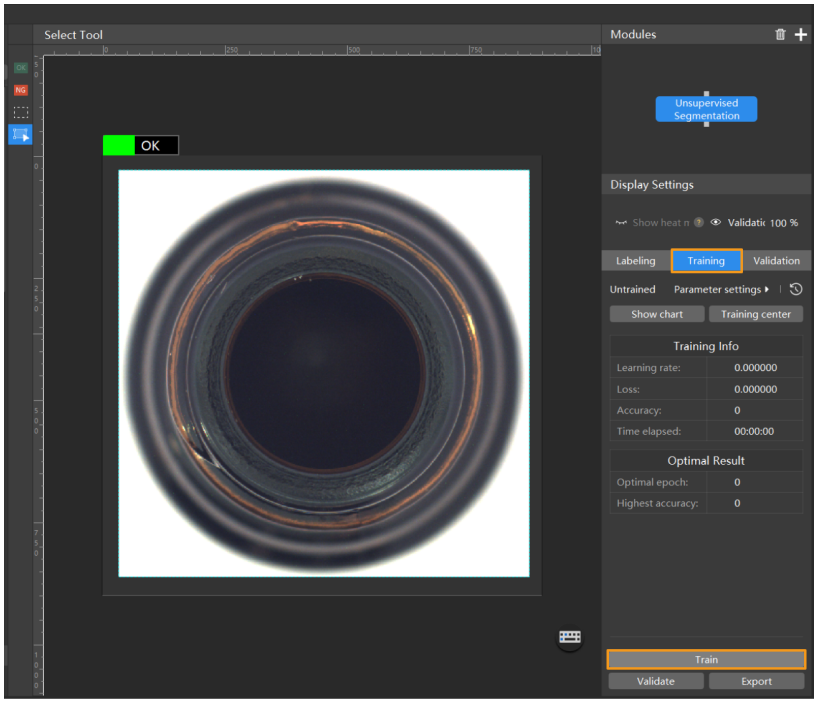
5. **Label images**: Use **OK Label** OK or **NG Label** NG on the toolbar to label images.



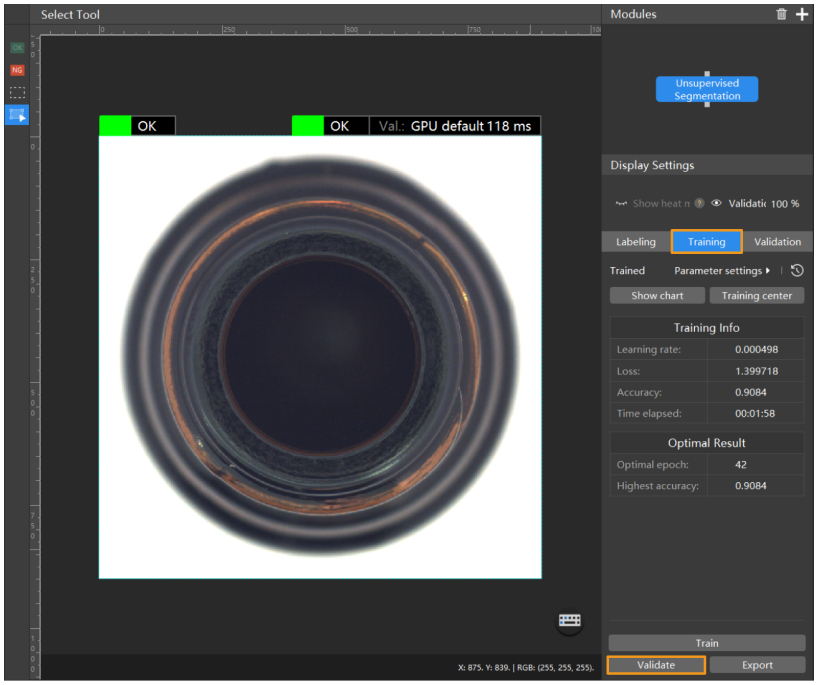
6. **Train the model**: Keep the default training parameter settings and click **[Train]** to start training the model.

If the training set contains images labeled as NG, these images will be automatically put into the validation set for validation during model training.

© 2023 Mech-Mind Robotics Technologies Ltd.



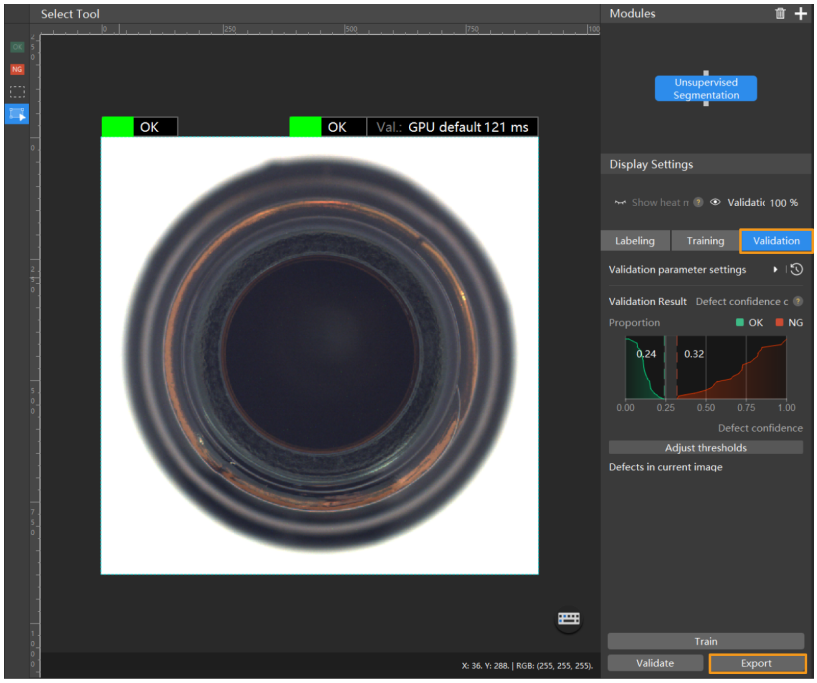
7. **Validate the model**: After the training is completed, click **[Validate]** to validate the model and check the results.



In the **Validation** tab, click **[Adjust thresholds]**, and in the pop-up dialog box, drag the vertical lines to adjust the thresholds. The green one is used to adjust the threshold of OK results, and the red one is used to adjust the threshold of NG results. Once the thresholds are adjusted, please re-validate the model.

8. **Export the model**: Click **[Export]** and select a directory to save the trained model.

© 2023 Mech-Mind Robotics Technologies Ltd.



The exported model can be used in Mech-Vision and Mech-DLK SDK. Click here to view the details.

2.8.3. Introduction to Labeling Tools

You can use labeling tools to add labels and hence provide data for deep learning training.

OK/NG Labels

Use the **OK Label** OK or the **NG Label** NG to label an image as OK or NG.

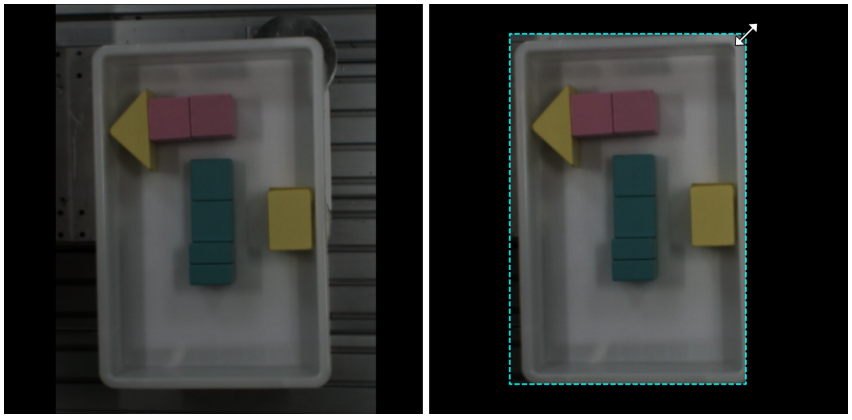
- Select an image from the image list and then click the **OK Label** or the **NG Label**. The image will be labeled as OK or NG.

[OK or NG label tool] | *dlk-software-manual/2.5.0/algorithm-modules/_images/unsupervised-segmentation/OK-or-NG-label-tool.gif*

ROI Tool

You can use the **ROI Tool** to set the region of interest.

Setting the ROI can avoid interferences from the background.



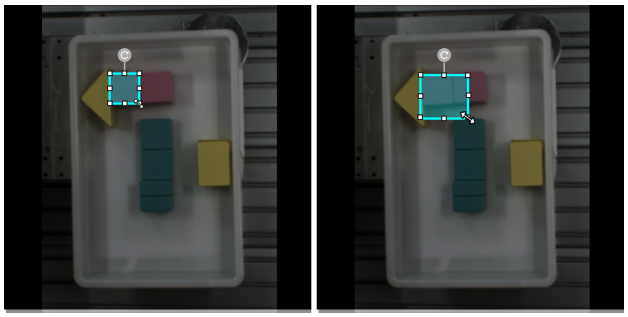
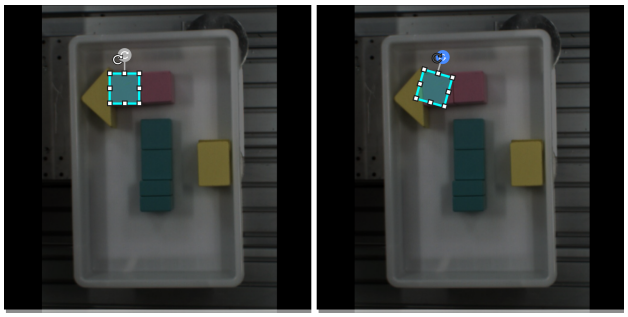
© 2023 Mech-Mind Robotics Technologies Ltd.

1. Click (or press O on the keyboard).
2. Adjust the ROI frame in the selection region.
3. Click **[Apply]** in the upper left corner of the selection region.

Select Tool

You can use the **Select Tool** to select, move, and adjust the selections.

1. Click (or press S on the keyboard).
2. Move the cursor in the selection region and then click the selection to be processed. Select multiple selections by pressing and holding the `Ctrl` key.

Adjust size	
Rotation	
Move	

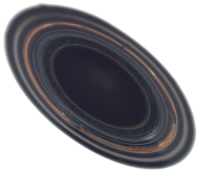
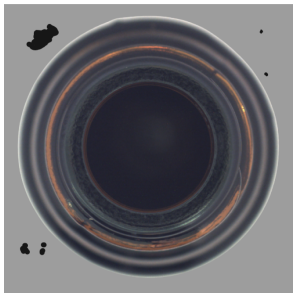
2.8.4. Train a High-Quality Model

This section introduces the factors that most affect the model quality and how to train a high-quality unsupervised segmentation model.

Ensure Image Quality

- Do not use **images in which objects have different shapes and size and are in different positions, or images with changing backgrounds**. Such situations will greatly impact model performance. Therefore, it is necessary to ensure the used OK images have small but important differences.

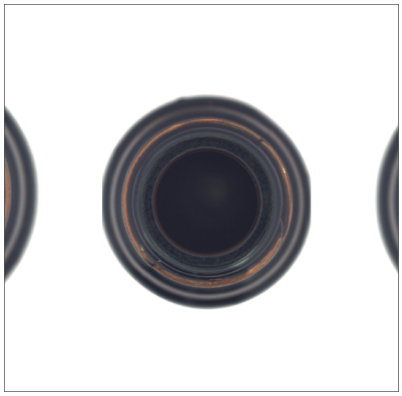
© 2023 Mech-Mind Robotics Technologies Ltd.

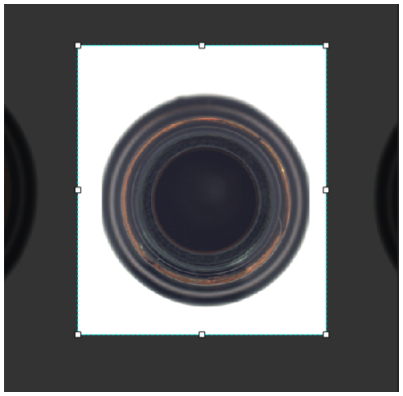
Most images	Incorrect examples		
			

If the image backgrounds can bring some inference, you can add an **Object Detection** module before **Unsupervised Segmentation**; if the objects in images have different orientations, add a **Fast Positioning** module before it. See the Use Cascaded Modules section for detailed instructions.

Set a Proper Region of Interest (ROI)

Setting the ROI can effectively eliminate the interference of the background, and the ROI boundary should be as close to the outer contours of the objects as possible.



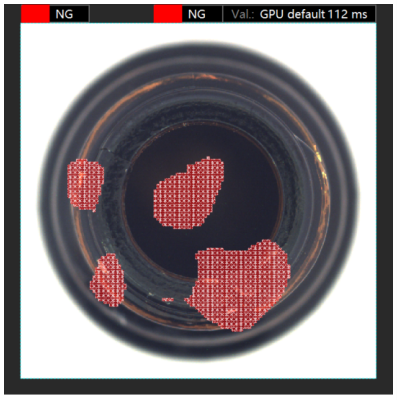
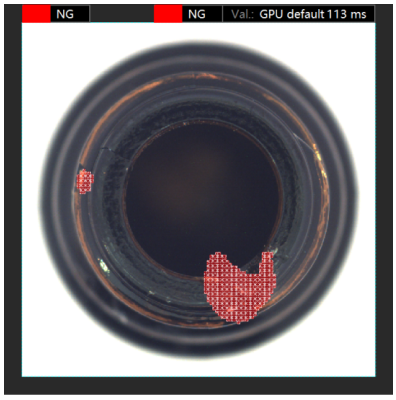


The same ROI setting will be applied to all images, so it is necessary to ensure that objects in all images are within the set ROI.

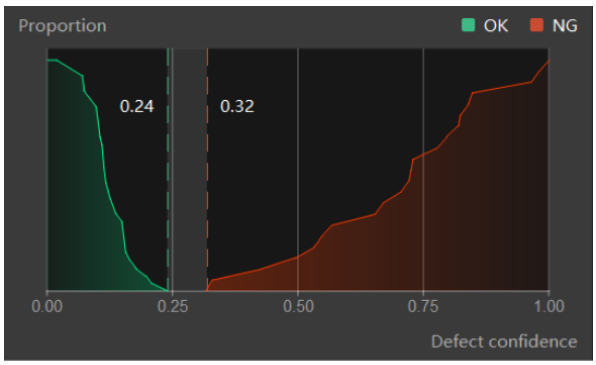
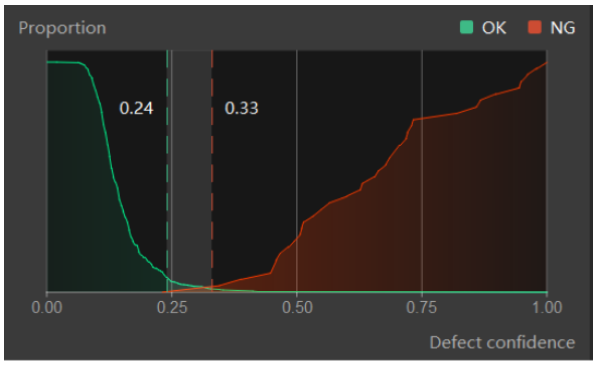
Set Appropriate Thresholds for OK and NG Results

Once thresholds are set for the defect confidence chart, the Unsupervised Segmentation module can judge accordingly whether an image is OK, NG, or Unknown. In addition, a greater threshold of OK results indicates a smaller defect area identified by the algorithm. It is necessary to set appropriate thresholds for OK and NG results according to actual situations to obtain the optimal performance of the trained model.

© 2023 Mech-Mind Robotics Technologies Ltd.

Original image	Threshold of OK results: 0.10	Threshold of OK results: 0.20
		

In the defect confidence chart, the y-coordinate of the green curve stands for the ratio of the number of OK images with defect confidence lower than the set threshold of OK results to the total number of OK images. The y-coordinate of the red curve stands for the ratio of the number of NG images with defect confidence greater than the set threshold of NG results to the total number of NG images. If the two curves do not intersect, the OK and NG images can be completely separated by thresholds. In other words, it is possible to leave no images labeled as Unknown. In this case, the farther apart the two lines are from each other, the better.

Good result	Poor result
	

Use Appropriate Data

- **Control the image quantity of training set**

  When you first use the **Unsupervised Segmentation** module, it is recommended to use 30 to 50 images for model training. If the validation results are not up to standard, you can then add more data.

- **Comprehensive and balanced OK images in training set**

  If OK images present multiple possible situations, please ensure all the possible situations are covered and evenly distributed in the training set.

- **Reduce inadequate data**

  Ensure that every image imported can facilitate the model training process. Adding highly similar images and repeated images makes a slight difference to model training, and it may slow down the training process.

© 2023 Mech-Mind Robotics Technologies Ltd.

2.9. General Parameters

2.9.1. Data Processing

Image Preprocessing

You can adjust the brightness, contrast, and color balance of images.

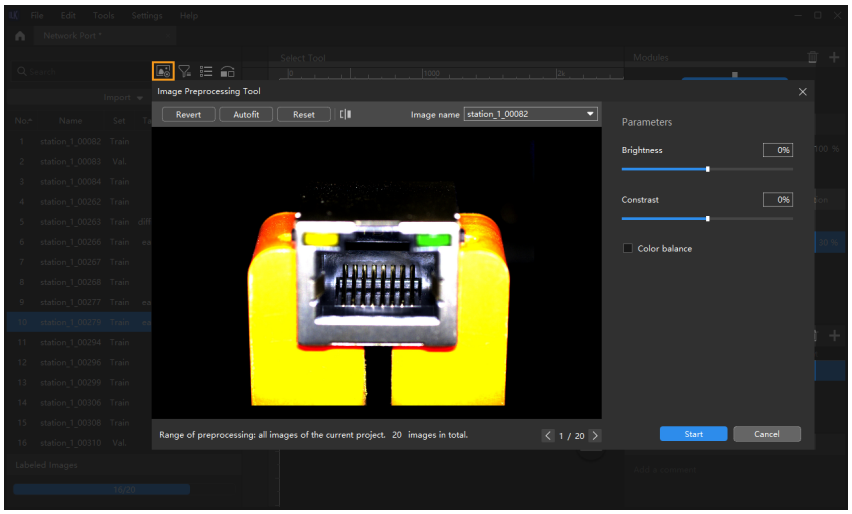
Typical Usage Scenarios

The original images captured by the camera might be dark or don't have obvious object features due to lighting or other factors. In such cases, you can preprocess images for better results.

Steps

1. Click to open the Image Preprocessing Tool.
2. Adjust the parameters to satisfy your needs.
3. Click **[Start]** and wait for the processing to complete.
4. Click **[OK]** in the pop-up window.

Range of preprocessing: All images of the current project. The preprocessed images will replace the original images in the training, test, and validation sets.



Filter Images

You can set the filtering conditions to conveniently select specific images from datasets.

Typical Usage Scenarios

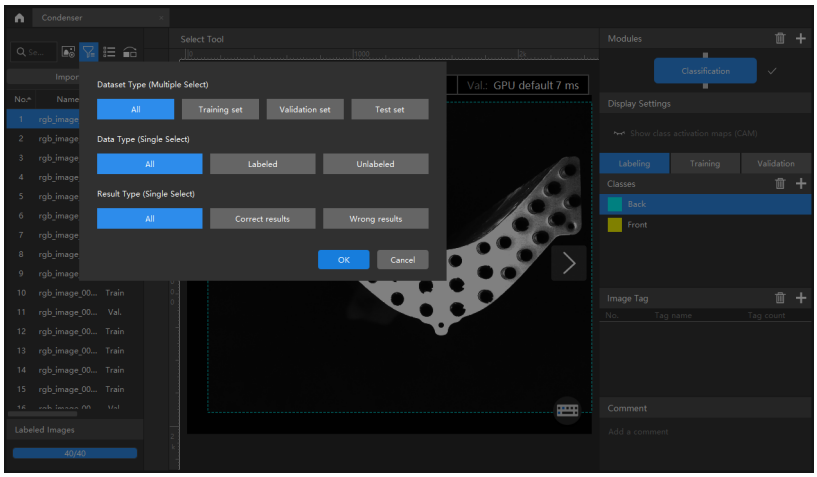
- View the results of the dataset division. When the images are divided into different datasets, you can filter images by dataset type to view the content of each dataset and check whether the division results need to be adjusted.
- Check the data labeling results. During labeling, you can view the labeling progress and labeling results by data type.
- Check the model prediction results. After validation, you can view the validation results by result type. For instance, when a defect segmentation model makes an incorrect prediction,

© 2023 Mech-Mind Robotics Technologies Ltd.

you can filter the images by selecting **False positive** and **False negative** for the result type and do a quick check. The incorrect prediction might result from wrong labels.

Steps

1. Click .
2. You can set the filtering conditions according to actual needs.
3. Click **[OK]**.



Switch to Preview Mode

The list view mode is the default mode, and you can click to view icons. You can drag the slider to change the display size of icons. Click it again to display the list.

Partition Training and Validation Sets

By default, 80% of the images in the dataset will be split into the training set, and the rest 20% will be split into the validation set. Please make sure that both the training set and validation set include **images in all different classes**, which will guarantee that the model can learn all different features and validate the images of different classes properly.

Steps

1. Click .
2. Drag the slider to adjust the proportion of the training set and validation set.

Right-click the image and click **[Switch to training set]**, **[Switch to validation set]**, or **[Switch to test set]** on the pop-up menu to modify the set to which the current image belong.

The test set is not involved in training and validation.

© 2023 Mech-Mind Robotics Technologies Ltd.

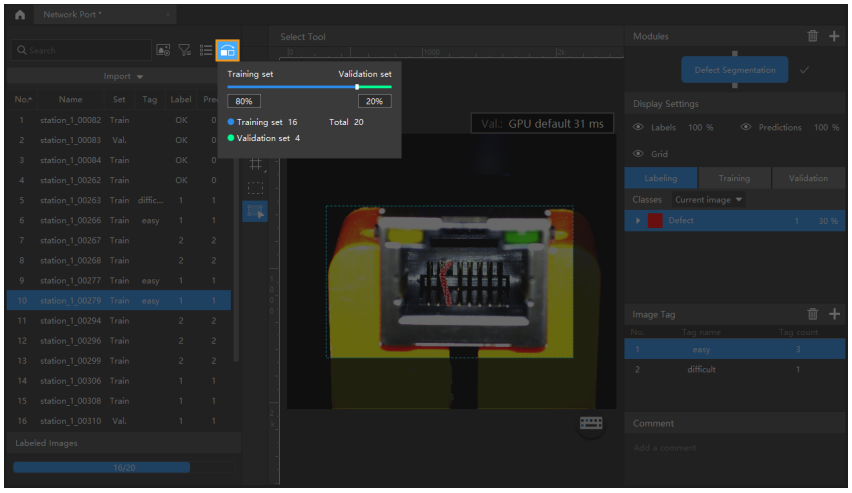


Image Tag

With the image tagging function, you can add tags to images and thereby manage massive image data in a centralized manner according to the usage scenarios of images.

Typical Usage Scenarios

- For using the Defect Segmentation module:

  If the objects in the images have multiple types of defects, the images need to be divided by defect types, and you can divide the images by adding tags.

  If it is hard to instantly determine whether the current image contains any defects, you can tag the image first to make a judgment on the image later.

- After validation, you can tag the images with poor validation results so that you can export the tagged images in groups for optimizing the model.

Steps

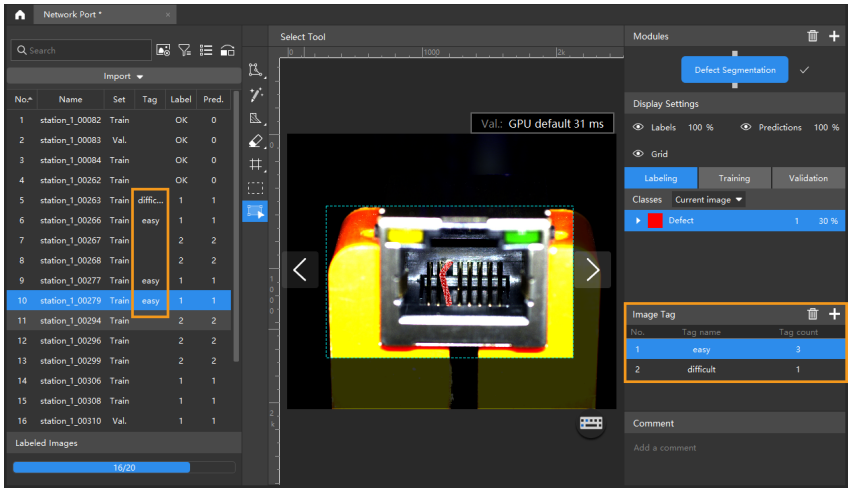
1. Create a new tag
   a. Click **[+]** on the management panel of Image Tag. (The management panel of Image Tag is in the lower right corner of the software interface)
   b. By default, the tag name is imageTag1. Double-click it to rename it.

   - Repeat the above steps to create more tags.
   - Select the tag and click to delete it. The tag added to the corresponding image(s) will be cleared once deleted.

2. Add tags to an image
   a. Click an image in the list or select multiple images.
   b. Click the corresponding tag on the management panel of Image Tag. The tag will be added to the image(s).

   - Press `Ctrl` or `Shift` on the keyboard to select multiple images.
   - Right-click the tagged image(s) and click **[Clear tags]** to clear the tags of the selected image(s).

© 2023 Mech-Mind Robotics Technologies Ltd.



Add a Comment

Click an image in the list and add a comment on the Comment bar.

2.9.2. Model Training

You can get started with model training upon labeling. On the parameter bar of Training, you can configure training parameters, train models, and view training information.

Training Parameters

Click **[Parameter Configuration]** to open the Training parameter configuration window.

Data Augmentation

The data for training the model needs to contain as much as possible all situations that may actually occur. If the site does not have the corresponding data-collecting conditions, you can adjust the **Data Enhancement** parameters to prepare data that can not be collected, thus enriching the training data. It must be ensured that the augmented image data should conform to the on-site situation. If there are no rotations on the site, then there is no need to adjust the parameter “Rotation”; otherwise, the model’s performance may be affected.

Hover the mouse cursor over ? to view the adjustment effect of each parameter.

- Lightness

  It refers to how much light is present in the image. When the on-site lighting changes greatly, by adjusting the **brightness** range, you can augment the data to have larger variations in brightness.

- Contrast

  Contradiction in luminance or color. When the objects are not obviously distinct from the background, you can adjust the **contrast** to make the object features more obvious.

- Translation

  Add the specified horizontal and vertical offsets to all pixel coordinates of the image. When the positions of on-site objects (such as bins and pallets) move in a large range, by adjusting the **translation** range, you can augment the data in terms of object positions in images.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Rotation

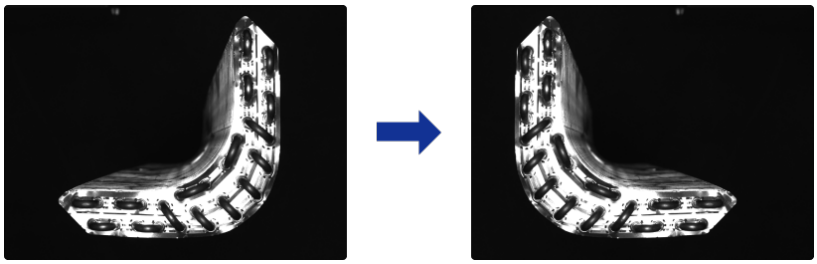
  Rotate an image by a certain angle around a certain point to form a new image. In general, keeping the default parameters can meet the requirements. When the object orientations vary greatly, by adjusting the **rotation** range, you can augment the image data to have larger variations in object orientations.

- Zoom

  Shrink or enlarge an image by a certain scale. When object distances from the camera vary greatly, by adjusting the **scale** range parameter, you can augment the data to have larger variations in object proportions in the images.

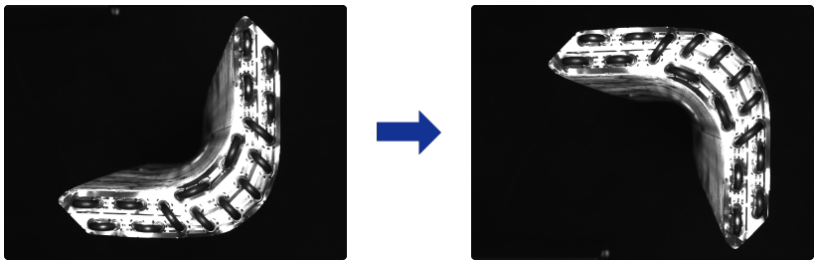
- Flip horizontally

  Flips the image 180° left to right. If the objects to be recognized have left-right symmetry, you can select the **Flip horizontally** check box.



- Flip vertically

  Flips the image 180° upside down. If the objects to be recognized have vertical symmetry, you can select the **Flip vertically** check box.



- Label expansion

  Only supported in the **Defect Segmentation** module. Expand the regions of defects selected in an image by a certain scale. For most scenarios, you do not need to check this option. If the regions of defects are too small, you can select the **Label expansion** checkbox to avoid influence on training when the defect regions are too small after image scaling.

Training Parameters

- Input image size

  The pixel-wise height and width of the image input to the neural network for training. It is recommended to use the default setting, but if the objects or defect regions in the images are small, you need to increase the **input image size**. The larger the image size, the higher the model accuracy, but the lower the training speed.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Batch size

  The number of samples selected for each time of neural network training. It is recommended to use the default settings; if you need to increase the training speed, you can appropriately increase the **batch size**. If the batch size is set too large, memory usage will increase.

- Model type

Defect Segmentation	Normal	Generally, it is recommended to use **Normal** mode
	Enhanced	You can choose the **Enhanced** mode when the model effect is not as expected or the accuracy requirement is high. This mode will decrease the training speed
Instance Segmentation	Normal (better with GPU deployment)	Select this option when the model is deployed on a GPU device
	Lite (better with CPU deployment)	Select this option when the model is deployed on a CPU device

- Eval. interval

  The number of epochs for each evaluation interval during model training. It is recommended to use the default setting. Increasing the **Eval. interval** can increase the training speed. The larger the parameter, the faster the training; the smaller the parameter, the slower the training, but a smaller value helps select the optimal model.

- Epochs

  The total number of epochs of model training. It is recommended to use the default setting. If the features of objects to be recognized are complex, it is necessary to increase the number of training epochs appropriately to improve the model performance, but increasing the number of epochs will lead to longer training time.

  It is not true that the bigger the number of epochs, the better. When the total number of epochs is set to be large, the model will continue to be trained after the accuracy stabilizes, which will result in a longer training time and the risk of overfitting.

- Learning rate

  The learning rate sets the step length for each iteration of optimization during neural network training. It is recommended to use the default setting. When the loss curve shows a slow convergence, you can appropriately increase the **learning rate**; if the accuracy fluctuates greatly, you can appropriately decrease the **learning rate**.

- GPU ID

  Graphics card information of the model deployment device. If multiple GPUs are available on the model deployment device, the training can be performed on a specified GPU.

- Model simplification

  This option is used to simplify the neural network structure. Unselected by default. When the training data is relatively simple, checking the option can improve the training and inference speeds.

© 2023 Mech-Mind Robotics Technologies Ltd.

- Max num of training objects

  Unselected by default. This option is to limit the maximum number of objects that can be recognized during training. In general, setting this parameter can speed up inference. Under certain circumstances, setting this parameter can help reduce GPU usage.

Model Finetuning

When a model is put into use for some time, it might not cover certain scenarios. At this point, the model should be iterated. Usually, using more data to re-train the model can do the job, but it could reduce the overall recognition accuracy and might take a long time. Hence, Model Finetuning can be used to iterate the model while maintaining its accuracy and saving time.

This feature only works under the Developer Mode. You can enable the Developer Mode by clicking Settings › Options.

Steps:

1. Collect images with poor recognition results and add them into the training and validation sets.
2. Enable Model Finetuning in the window of Training parameter settings and lower the learning rate accordingly; the number of training epochs can be reduced to 50–80.
3. Confirm the changes to the parameters and start the training.

For example, you can enable Model Finetuning and then select the path of Super Model in the Training parameter configuration window of the Instance Segmentation model to finetune the Super Model.

Training Center

You can view the training progress and memory usage in the **Training Center**.

If there is already a training task in progress, other training tasks will enter a waiting queue.

- Click to terminate the training task in progress.
- Click to remove the training task from the waiting queue.
- Click to stick the training task to the top of the waiting queue.

2.9.3. Model Validation

After model training, you can configure validation parameters, validate and view the recognition results of models on the Validation parameter bar. In addition, you can set the confidence in the Object Detection and Instance Segmentation modules to filter results.

Validation parameters

Click **[Validation parameter settings]** to open the window for validation parameter settings.

- Hardware type
  - CPU: Use CPU for deep learning model inference, which will increase inference time and reduce recognition accuracy compared with GPU.
  - GPU (default): Do model inference without optimizing according to the hardware, and the model inference will not be accelerated.

© 2023 Mech-Mind Robotics Technologies Ltd.

    - GPU (optimization): Do model inference after optimizing according to the hardware. The optimization only needs to be done once and is expected to take 5–15 minutes. The inference time will be reduced after optimization.

- GPU ID

  The graphics card information of the device deployed by the user. If multiple GPUs are available on the model deployment device, the model can be deployed on a specified GPU.

- Float precision
    - FP32: high model accuracy, low inference speed.
    - FP16: low model accuracy, high inference speed.
- Max num of inference objects (only visible in the **Instance Segmentation** module and **Object Detection** module)

  The maximum number of inference objects during a round of inference is 100 by default.

- Character limit (only visible in the **Text Recognition** module)

  The maximum quantity of characters that can be recognized from an image, which is 50 by default.

After parameter setting, click OK › Validate and wait for the validation to complete.

© 2023 Mech-Mind Robotics Technologies Ltd.

3. Application Guide of Mech-DLK

3.1. Usage Scenarios of Deep Learning

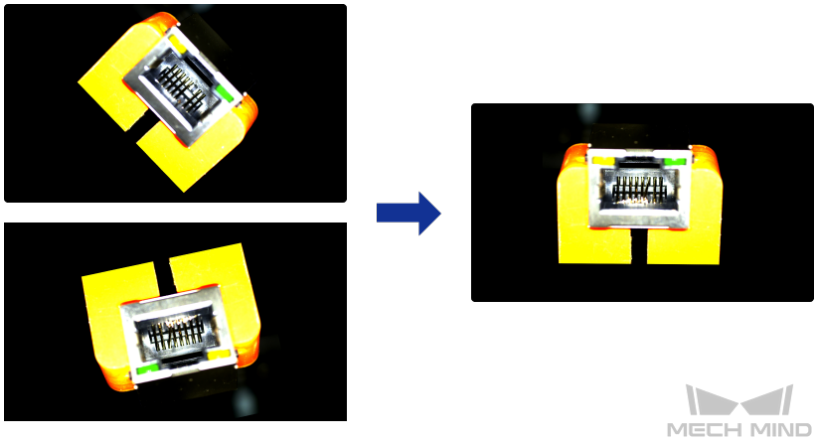
2D Camera + Deep Learning

The following show the usage scenarios of a 2D camera + deep learning. Use these deep learning modules according to actual needs.

Fast Positioning

You can use the **Fast Positioning** module to correct the object orientations.

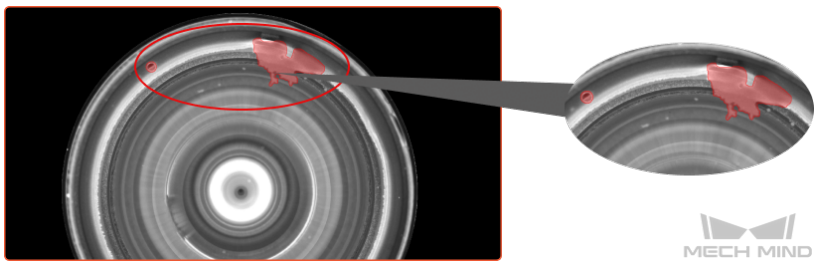
- Recognizes object orientations in images and rotates these images so that objects in them have a uniform orientation.



Defect Segmentation

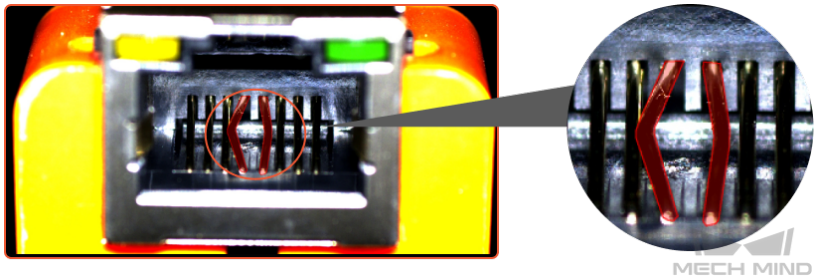
You can use the **Defect Segmentation** module to detect all types of defects. These defects include surface defects, such as stains, bubbles, and scratches, and positional defects, such as bending, abnormal shape, and absence. Moreover, this module can be applied even under complex conditions, such as small defects, complicated background, or random object positions.

- Detects air bubbles and glue spills on lenses.



- Detects bending defects of parts.

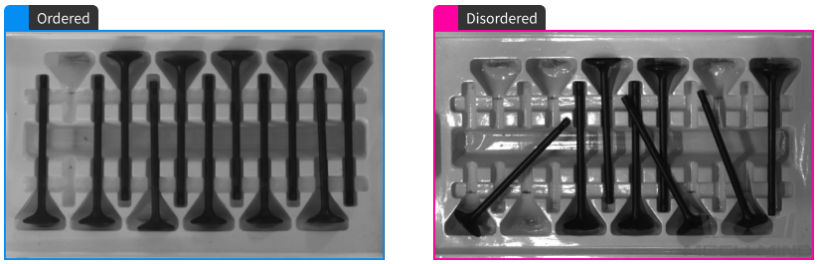
© 2023 Mech-Mind Robotics Technologies Ltd.



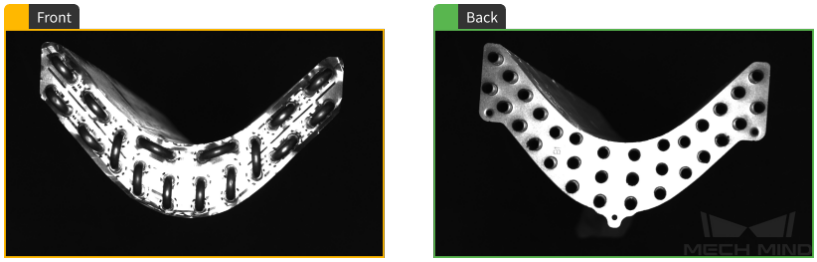
Classification

You can use the **Classification** module to recognize the fronts and backs of objects, object orientations, and defect types and to judge whether objects are missing or neatly arranged.

- Recognizes whether objects are neatly arranged or scattered.



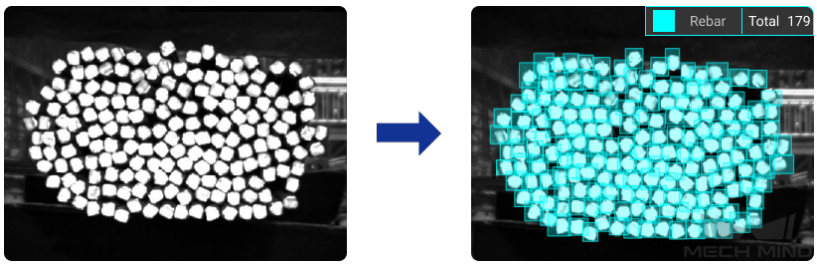
- Recognizes the fronts and backs of objects.



Object Detection

You can use the **Object Detection** module to detect the absence of objects in fixed positions, such as missing components in a PCB. You can also use it for object counting.

- Counts all rebars.

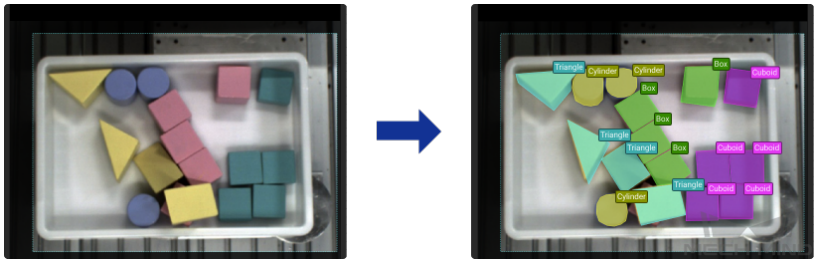


Instance Segmentation

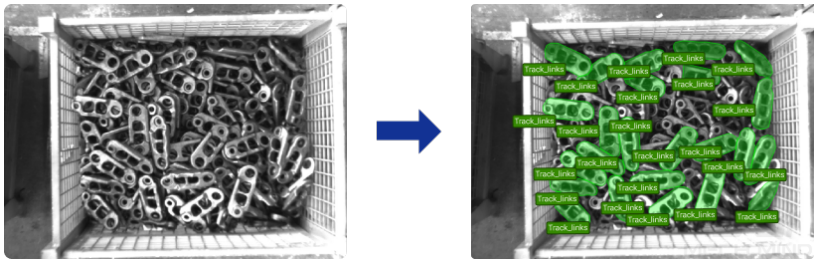
You can use the **Instance Segmentation** module to distinguish objects of a single type or multiple types and segment their contours.

© 2023 Mech-Mind Robotics Technologies Ltd.

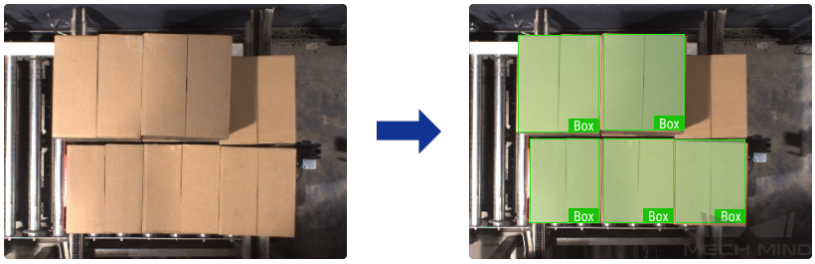
- Segments blocks of various shapes.



- Segments scattered and overlapping track links.



- Segments cartons closely fitted together.

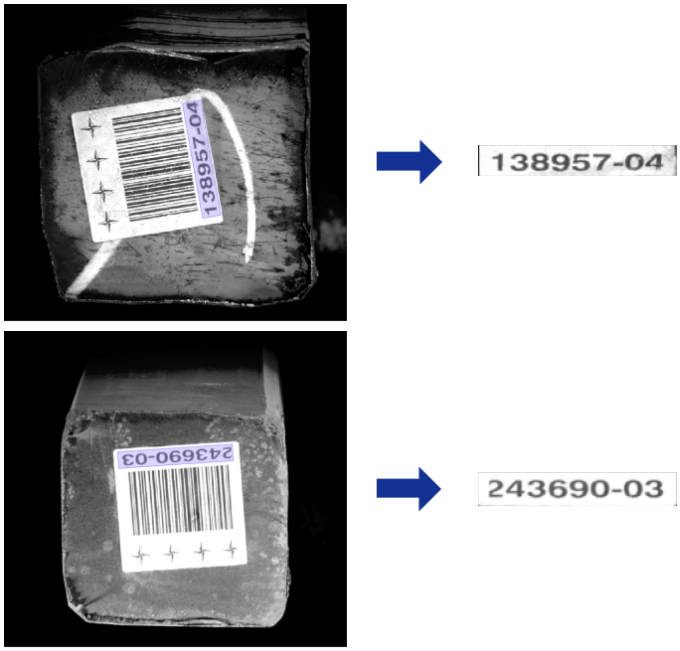


Text Detection

You can use the **Text Detection** module to detect the text area in an image and thus reduce the interference from the background.

- Detects the text areas of an image.

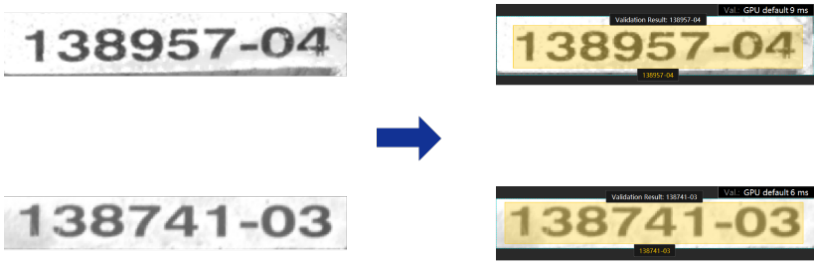
© 2023 Mech-Mind Robotics Technologies Ltd.



Text Recognition

You can use the **Text Recognition** module to recognize the text in an image, including letters, numbers, and some special symbols.

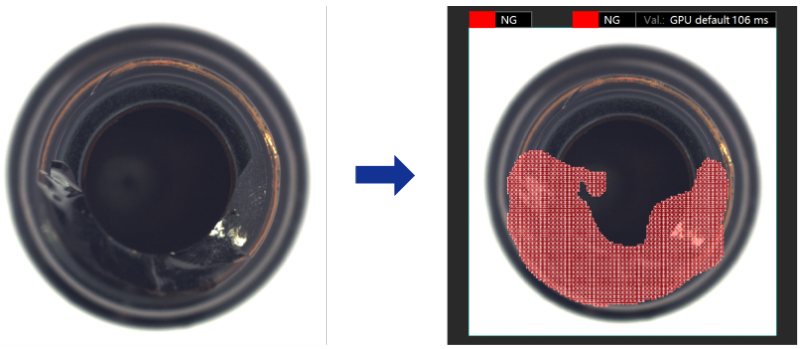
- Recognizes the characters in an image.



Unsupervised Segmentation

You can use the **Text Recognition** module to train a model on the basis of only OK images as the training set. The trained model can judge whether an image is OK, NG, or Unknown.

- Judges whether an image is OK, NG, or Unknown according to set thresholds and displays the possible areas with defects.



© 2023 Mech-Mind Robotics Technologies Ltd.

3D Camera + Deep Learning

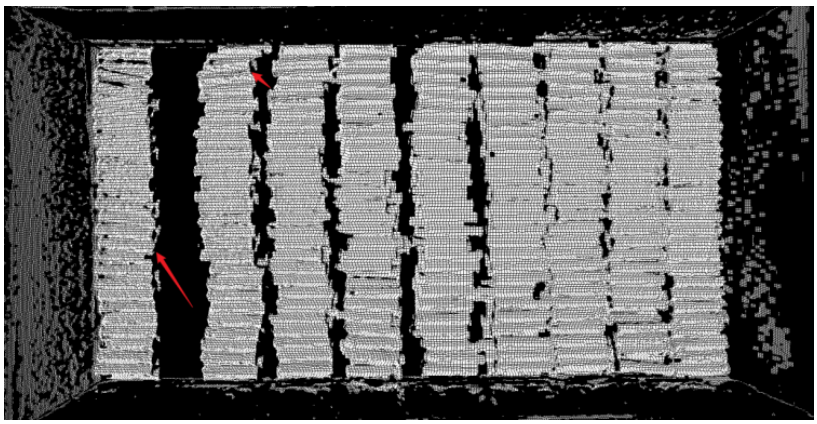
In the following scenarios, the data of point clouds cannot assure accurate recognition and positioning of objects. Therefore, we use 3D matching + deep learning to recognize and position objects.

Seriously Incomplete Point Clouds

The workobjects in the following figures are used as an example.

1. 2D image: In the figure below, large quantities of reflective workobjects are in contact with each other, but their edges and shape features are clear.

2. Point cloud: As the workobjects are reflective, the point clouds are incomplete. Analysis shows that missing point clouds are mostly along the workobject axis.



If point clouds are incomplete along the workobject axis, 3D matching using the point clouds may be inaccurate and thus results in largely deviated picking poses. As the workobjects are in contact with each other, the point clouds may not be correctly segmented and thus lead to incorrect matching. In addition, the quantity of workobjects is large, the vision cycle time will be long.

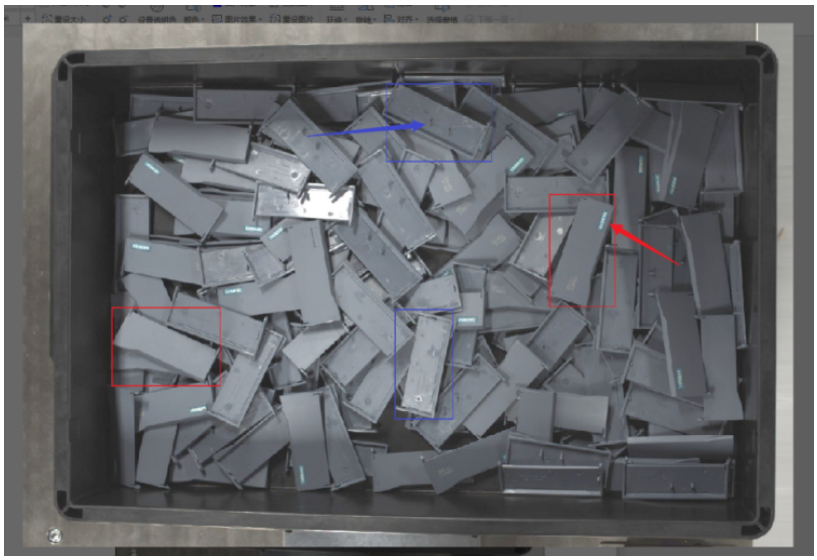
In such scenarios, you can use the Instance Segmentation module to train the corresponding model and then use the deep learning-related Steps in Mech-Vision to recognize the workobjects. Then, extract the point cloud corresponding to the mask of the workobject and calculate its Pose A on the basis of 3D matching. Finally, calculate Pose B on the basis of the extracted point cloud and correct the X and Y components of Pose A.

© 2023 Mech-Mind Robotics Technologies Ltd.

Key Features Missing from Point Cloud

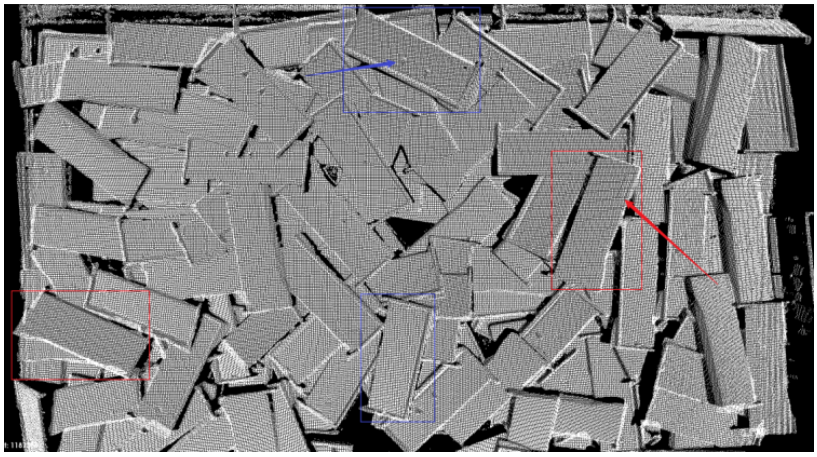
The workobjects in the following figures are used as an example.

1. 2D image: In the figure below, the workobjects in the red boxes have their fronts facing up, while the workobjects in the blue boxes have their backs facing up. The arrows point to key features used to distinguish between fronts and backs.



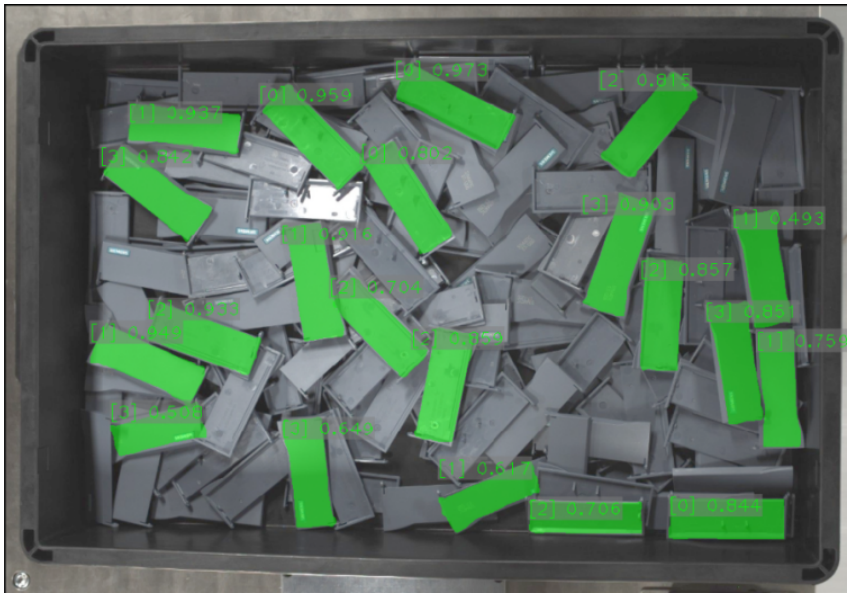
2. Point cloud: In the point cloud, the key features used to distinguish between fronts and backs of workobjects are missing.

© 2023 Mech-Mind Robotics Technologies Ltd.



As the key features used to distinguish workobject types are very small (even missing from the point cloud), when 3D matching is used to recognize different workobject types, the matching result may be incorrect, which results in the wrong classification of the workobjects.

In such scenarios, you can use the Instance Segmentation module to train the corresponding model and set corresponding labels for different types of workobjects. When this model is used in the deep learning-related Steps in Mech-Vision, the Steps will not only extract the masks of each workobject but also output the type labels of workobjects.

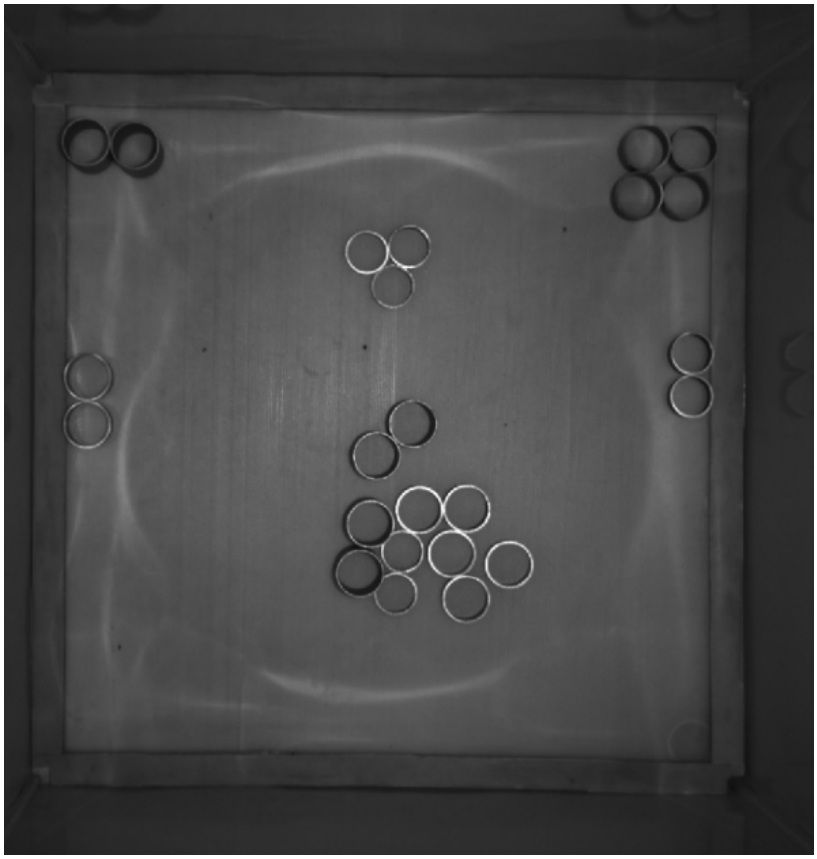


Almost No Point Cloud of Workobjects

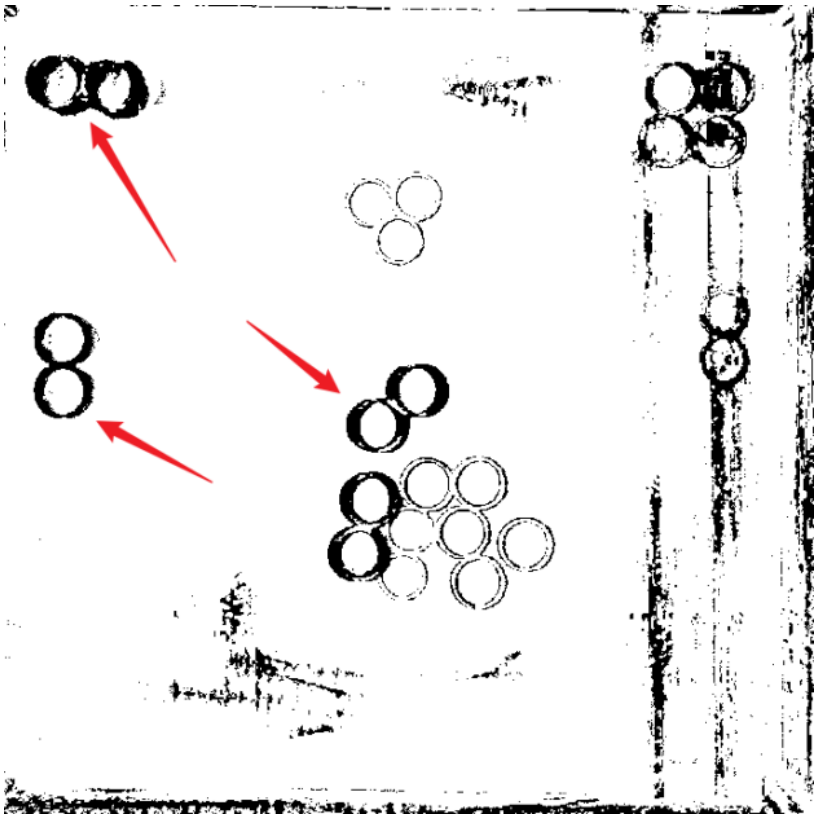
The workobjects in the following figures are used as an example.

1. 2D image: Wave washers are reflective and are placed close to each other in the bin.

© 2023 Mech-Mind Robotics Technologies Ltd.



2. Point cloud: The point clouds of the workobjects are unstable. Chances are that the points of some workobjects are completely missing.



As many object features are missing from the point clouds, it is impossible to use the point clouds

© 2023 Mech-Mind Robotics Technologies Ltd.

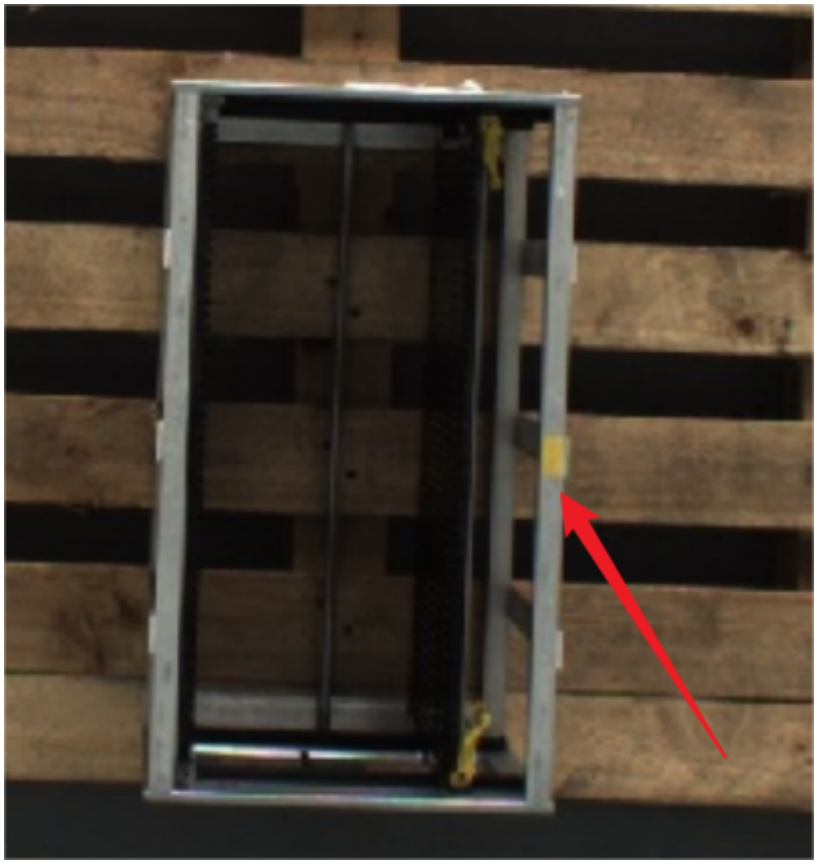
for positioning the workobjects and calculating their poses by 3D matching. When 3D matching is used, incorrect matching with the bin bottom is likely to occur.

In such scenarios, although the workobjects are reflective, their edges are clear in the 2D images. Therefore, you can use the Instance Segmentation module to train the corresponding model and use this model in the deep learning-related Steps in Mech-Vision. The Steps output workobject masks, on the basis of which point clouds of workobjects are extracted. The poses of these point clouds are then calculated and used as the picking pose.

Recognize Patterns and Colored Region on Workobject Surface

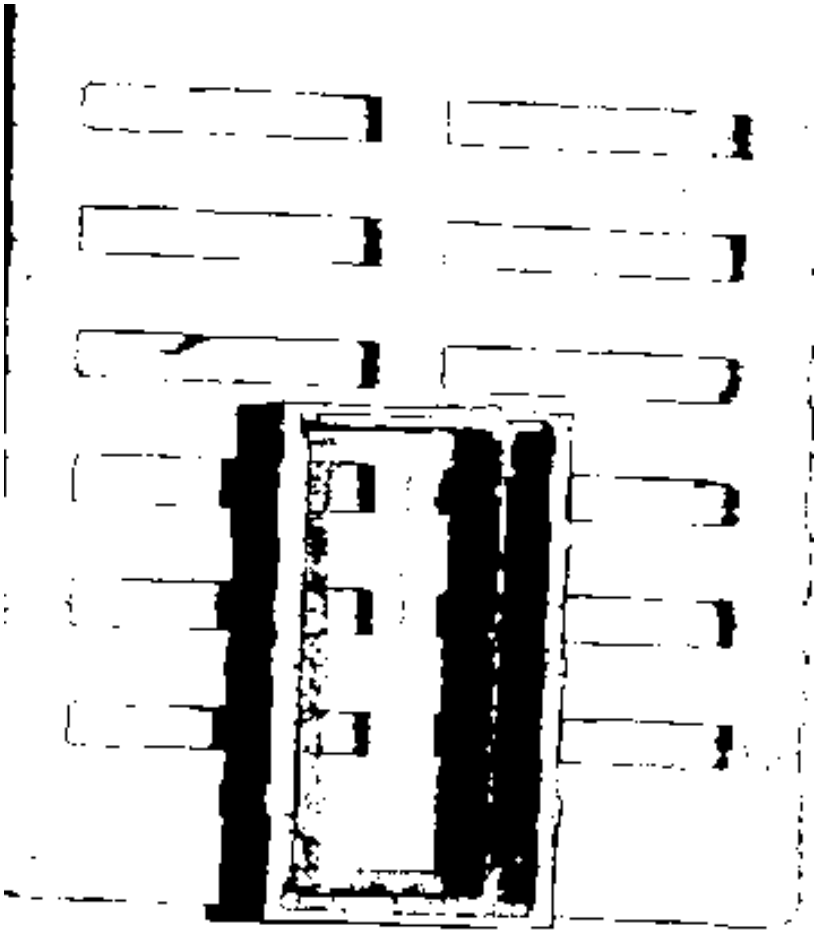
The workobjects in the following figures are used as an example.

1. 2D image: A piece of yellow tape is attached to one side of the aluminum bin, used to mark the orientation of the bin.



2. Point cloud: The point clouds are good, but the yellow tape cannot be reflected by the point clouds.

© 2023 Mech-Mind Robotics Technologies Ltd.



As the target feature is a colored region and is only visible in the color image, the orientation of the bin cannot be recognized through the point clouds.

In such scenarios, as long as the rough location of the yellow tape is determined, the orientation of the aluminum bin can also be determined. You can use the Object Detection module to train the corresponding model and use this model in the deep learning-related Steps in Mech-Vision to recognize the workobjects.

© 2023 Mech-Mind Robotics Technologies Ltd.

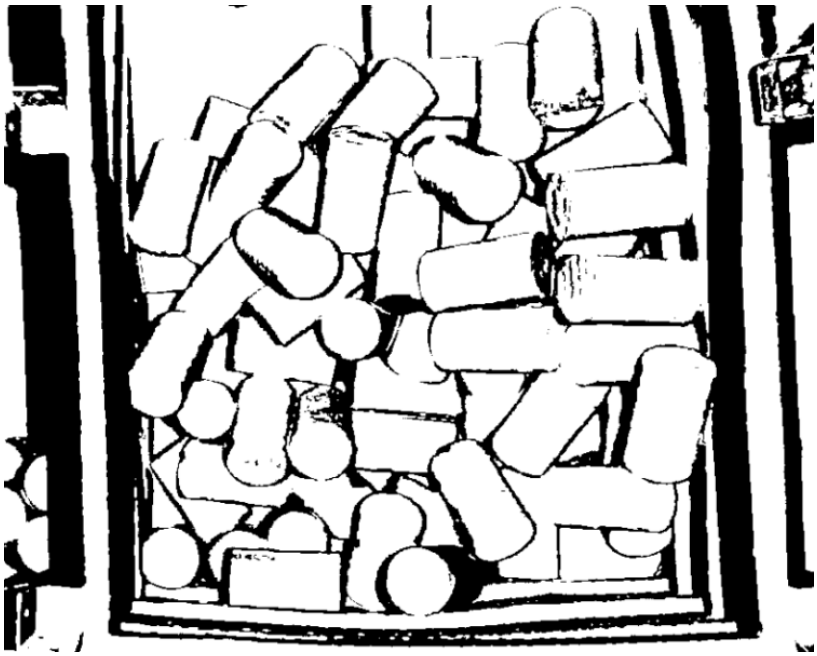
Random Picking from Deep Bin

The workobjects in the following figures are used as an example.

1. 2D image: Steel bars are randomly piled in the bin, and some regions of some steel bars reflect light. In addition, the steel bars overlap each other.

© 2023 Mech-Mind Robotics Technologies Ltd.

2. Point cloud: For workobjects not overlapped, their point clouds are good. For overlapped workobjects, it is difficult to render the point cloud of each workobject by clustering, and the clustering performance is unstable.

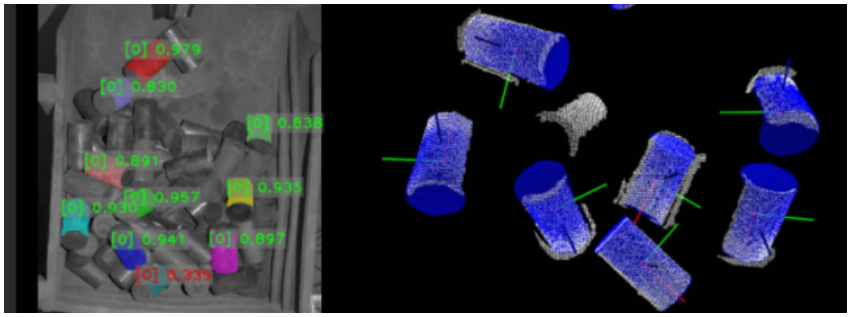


Point cloud clustering cannot stably cluster out the point cloud of each workobject. It is difficult to train a suitable model for 3D matching in that the orientations of workobjects vary greatly. 3D

© 2023 Mech-Mind Robotics Technologies Ltd.

matching might also output incorrect matching results, which leads to inaccurate pose calculation. In addition, using only point cloud models for global matching will result in a very long vision cycle time.

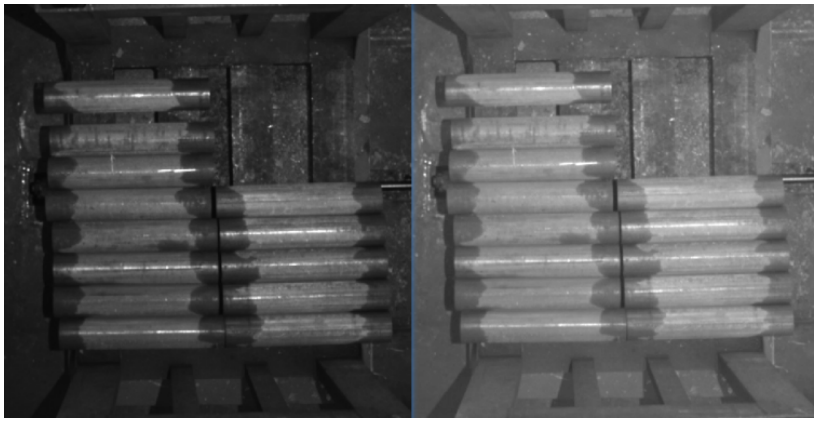
In such scenarios, you can use the Instance Segmentation module to train the corresponding model and use this model in the deep learning-related Steps in Mech-Vision to extract the masks of individual workobjects. Then, extract the point clouds corresponding to the masks and then use the point clouds for the matching of individual workobjects.



Point Cloud Clustering Fails to Separate Point Clouds of Overlapped Workobjects

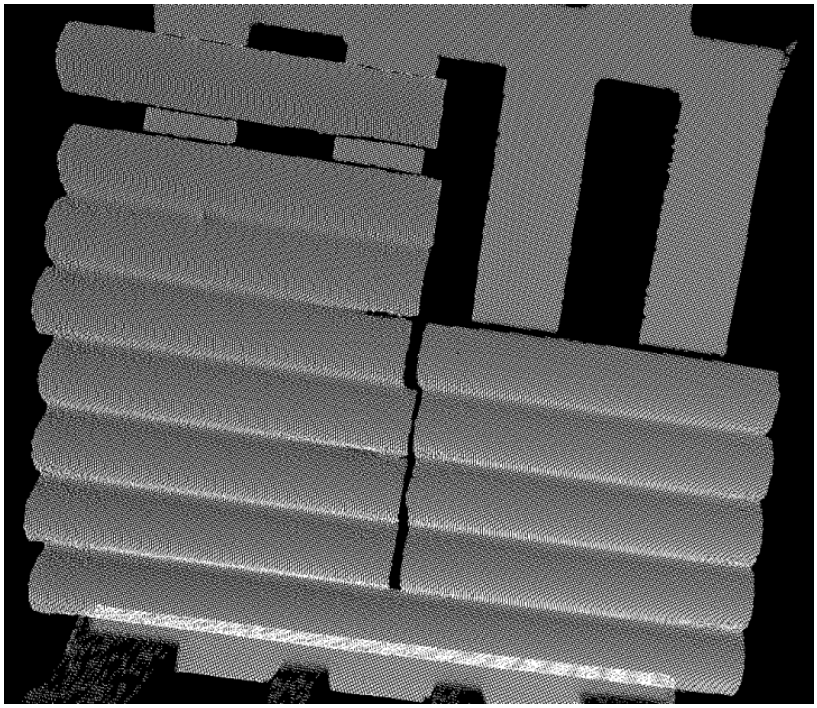
The workobjects in the following figures are used as an example.

1. 2D image: The obtained 2D images are too dark, and workobject edges may not be discernible. After the image brightness is enhanced, the workobject edge, size, and grayscale value can be obtained.

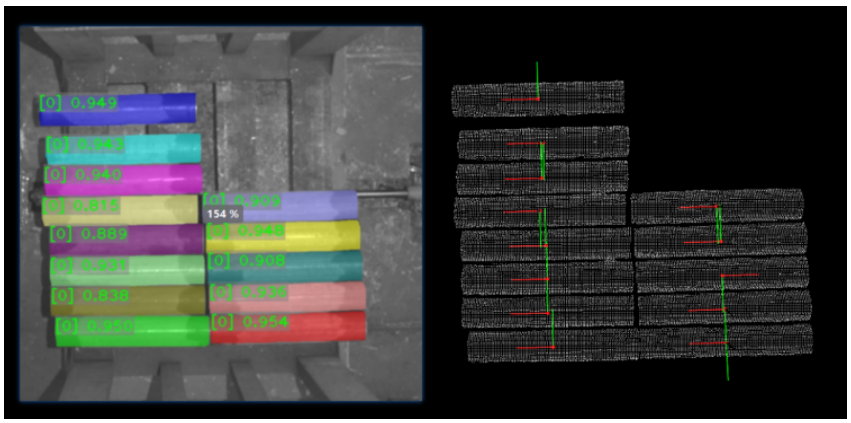
2. Point cloud: The point clouds are good, but the workobjects are in contact with each other. Therefore, the edges of different workobjects may not be correctly separated.

© 2023 Mech-Mind Robotics Technologies Ltd.



Although the point clouds are good, but the point clouds of workobjects stick to each other, so point cloud clustering cannot be used to segment the point clouds of individual workobjects. Global 3D matching might output incorrect matching results or even matching to the bin.

In such scenarios, you can use the Instance Segmentation module to train the corresponding model. The trained model can be used in the deep learning-related Steps in Mech-Vision to extract the workobject point clouds and then perform matching.



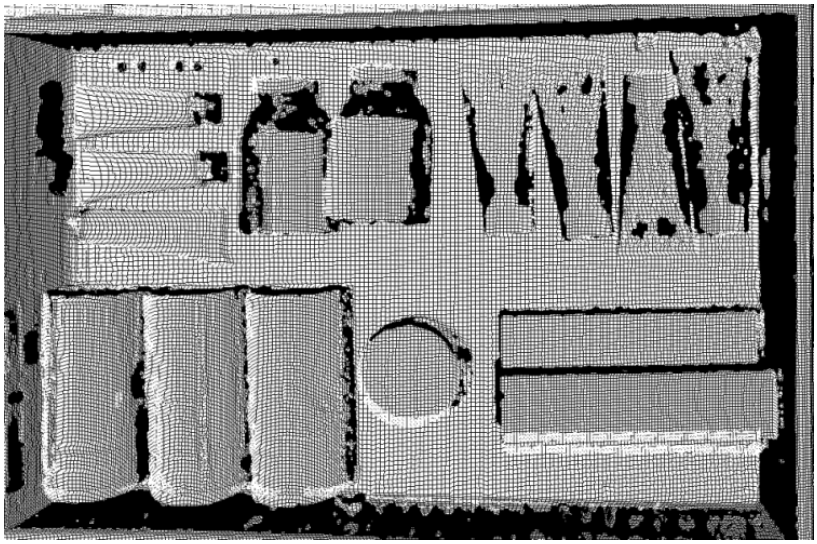
Recognize and Pick Workobjects of Different Types

The workobjects in the following figures are used as an example.

1. 2D images: The image on the left is obtained at the station where the workobjects are randomly placed. The image on the right is obtained at the station where secondary judgment is performed. The quality of the 2D images at both stations is good, and the features of different types of workobjects are distinctive.

© 2023 Mech-Mind Robotics Technologies Ltd.

2. Point cloud: The point clouds are good, and the shape features of the workobjects are clearly reflected in the point clouds.



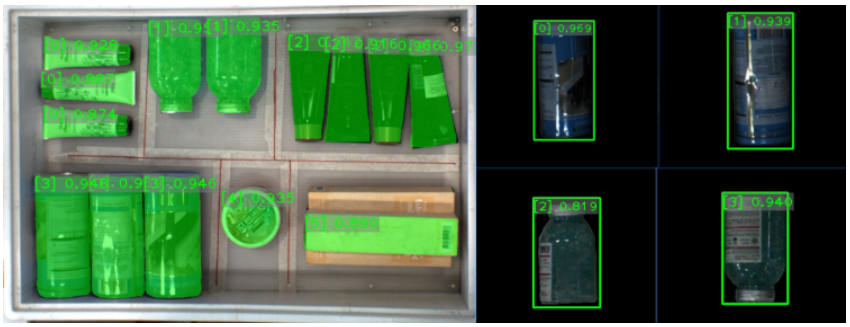
For cylindrical workobjects, however, the point clouds cannot reflect the orientation or the type of the workobject, so using only 3D matching cannot distinguish the workobject orientation and type.

For the above 2D image on the left, you can use the Instance Segmentation module to train the corresponding model and use the model in the deep learning-related Steps in Mech-Vision to recognize and segment the workobjects. The Steps will output the corresponding workobject masks, which can be used for subsequent point cloud processing.

For the above 2D image on the right, you can use the Instance Segmentation module to recognize

© 2023 Mech-Mind Robotics Technologies Ltd.

individual workobjects and then use the Object Detection module to determine the orientations of workobjects according to the shape and surface features. (The left figure below shows the instance segmentation result, and the right figure below shows the object detection result of determining the orientations of workobjects.)



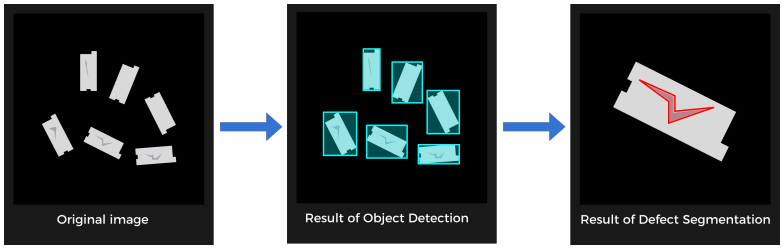
3.2. Cascade Modules

Cascading modules can help detect defects in complex scenarios such as changing positions of incoming materials and detection points and the scenarios requiring the classification of defects.

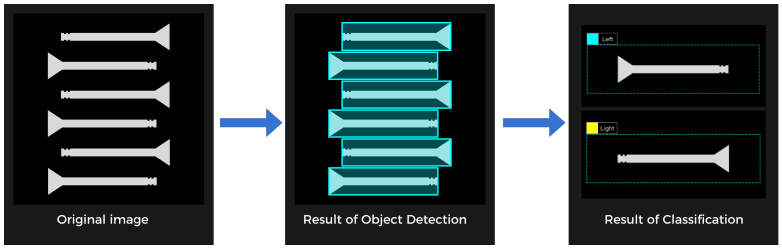
Common Cascading Combinations

The following are the common combinations:

1. Object Detection–Defect Segmentation
   - Feature: Position the to-be-detected objects in an image and then detect defects.
   - Applicable scenarios: There are many objects to be detected in the original image, and the position and number of objects are random; the shapes of defects vary.



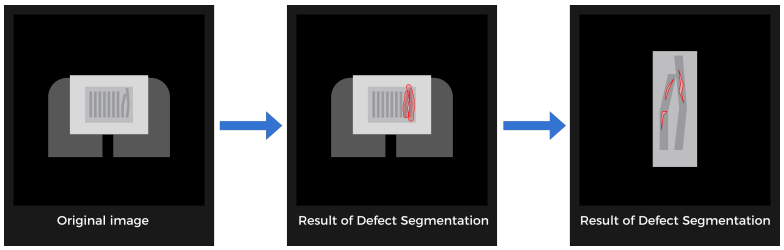
2. Object Detection–Classification
   - Feature: Position the to-be-detected objects in an image and then determine the orientations and colors of objects.
   - Applicable scenarios: There are multiple objects to be detected in the original image, and the number and positions of objects are random; the objects need to be classified.



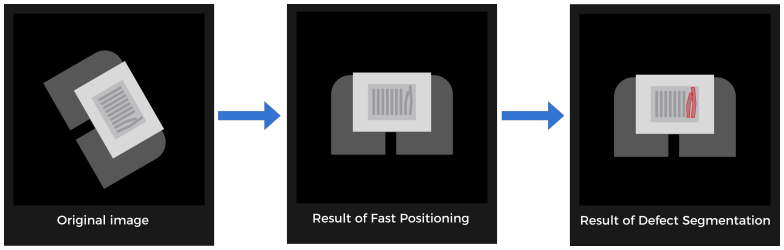
3. Defect Segmentation–Defect Segmentation

© 2023 Mech-Mind Robotics Technologies Ltd.

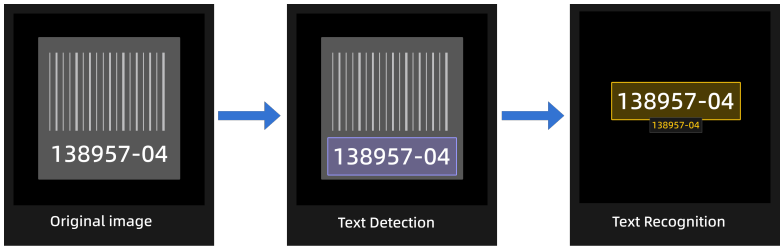
- Feature: The Defect Segmentation module segments the region to be detected and the background, and then the Defect Segmentation module performs defect detection on the extracted region.
- Applicable scenarios: Complex background, small or inconspicuous defects. The to-be-detected region should be extracted before fine defect detection.



4. Fast Positioning–Defect Segmentation
   - Feature: Single objects with relatively random positions and angles are relatively random. Thus, the images should be rotated to a specified angle and position before defect detection.
   - Applicable scenarios: single objects, random incoming materials, and changing defect shapes.



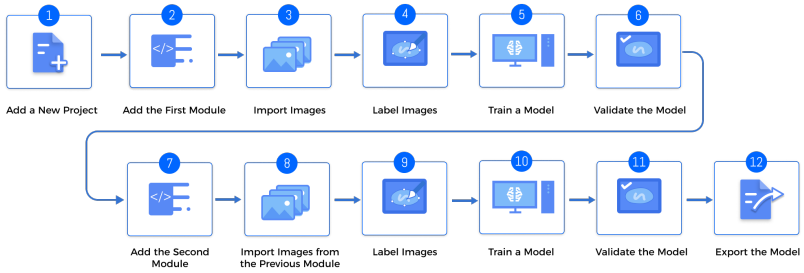
5. Text Detection–Text Recognition
   - Feature: The **Text Detection** module can locate and extract the text areas of images to reduce the interference from the background and the text orientations. The **Text Recognition** module can recognize characters in images, and it cannot be followed by any other module.
   - Applicable scenarios: complex backgrounds and different text orientations.



Use Cascaded Modules

First, confirm whether the dongle rights include the module cascading function by clicking the + sign in the upper-right corner of the Modules section. If you can click it, the function is enabled. If not, please contact Mech-Mind Sales to upgrade the authorization version.

© 2023 Mech-Mind Robotics Technologies Ltd.



1. Train the first module

   Train the first module by the training methods of single modules. After training, check whether the model is applicable.

2. Train the second module

   Click **[+]** in the upper-right corner of the software and add a module according to actual needs.

3. Import the data from the previous module

   The result of the previous module will be the input for the current module.

   a. Click **[Import]** and select Import from previous module.
   b. Select images.
   c. Modify Import Configuration.
      - Expand by: Expand the imported images by a specified number of pixels. The expansion can prevent data loss caused by insufficient accuracy of the previous module. Default value: 0 px
      - Rectify image(s): Rectify all imported images to the positive direction (0°).
      - Keep background: This option determines whether or not to save the pixels other than masks. Only masked images are imported when this option is turned off.
   d. Click **[Import]**.

4. Label and train the current module

   Complete the labeling and training of the current module.

5. Validate and export the model

   After training, validate the model and export it. The exported model can be used in Mech-Vision and Mech-DLK SDK.

   If the previous module is updated, the following module needs to be re-trained.

3.3. Use Models

Use Models in Mech-Vision

Usage Instructions

The exported models can be used in the Deep Learning Model Package Inference Step in Mech-Vision.

Compatibility

- It is recommended that the models exported from Mech-DLK 2.5.0 or later versions should be used in conjunction with Mech-Vision 1.8.0 or above.
- It is recommended that single models exported from Mech-DLK 2.4.1 or later versions should be used in Mech-Vision 1.7.0 or above.
- Single models exported from Mech-DLK 2.4.1 or later versions can only be used in Mech-Vision in version 1.7.2 or above.
- When the **Deep Learning Model Package Inference** Step performs inference on a model package exported from Mech-DLK 2.2.0 and early versions in Mech-Vision 1.7.2, defect determination rules configured for this model package are invalid. You need to configure them again in Mech-DLK in version 2.4.1 or above before exporting the model package.
- For model packages of object detection exported from Mech-DLK 2.4.1 or later versions, when the **Max num of inference objects** is set to 1, and the hardware type of the deep learning model package management tool is CPU, the inference speed becomes very low. It is recommended that the **Max num of inference objects** be greater than 1. (Only for model packages imported into Mech-Vision 1.7.x.)

▼ *Click here to view the details on compatibility.*

Instance Segmentation

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.4.0	1.4.0	Instance Segmentation (please start the deep learning server for the Step)	1.4.0	.pth/.py
1.5.x	2.0.0/2.1.0	Instance Segmentation (please start the deep learning server for the Step)	1.4.0	.pth/.py
	2.0.0/2.1.0	Instance Segmentation (please start the deep learning server for the Step)	2.0.0/2.1.0	.dlkmp/.dlkcfg
1.6.0	2.0.0/2.1.0	Instance Segmentation (please start the deep learning server for the Step)	1.4.0	.pth/.py
	2.0.0/2.1.0	Instance Segmentation (please start the deep learning server for the Step)	2.0.0/2.1.0	.dlkmp/.dlkcfg
	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpack

© 2023 Mech-Mind Robotics Technologies Ltd.

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.6.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.6.2	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.7.0	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.2	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.7.4	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

Classification

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.4.0	1.4.0	Classification (please start the deep learning server for the Step)	1.4.0	.pth/.json
1.5.x	2.0.0/2.1.0	Classification (please start the deep learning server for the Step)	1.4.0	.pth/.json
	No deep learning environment required	Deep Learning Inference	2.0.0/2.1.0	.dlkpack

© 2023 Mech-Mind Robotics Technologies Ltd.

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.6.0	2.0.0/2.1.0	Classification (please start the deep learning server for the Step)	1.4.0	.dlkpack
	No deep learning environment required	Deep Learning Inference (Mech-DLK 2.1.0/2.0.0)	2.0.0/2.1.0	.dlkpack
	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpack
1.6.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.6.2	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.7.0	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.2	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.7.4	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

Object Detection

© 2023 Mech-Mind Robotics Technologies Ltd.

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.4.0	1.4.0	Object Detection (please start the deep learning server for the Step)	1.4.0	.pth/.py
1.5.x	2.0.0/2.1.0	Object Detection (please start the deep learning server for the Step)	1.4.0	.pth/.py
	No deep learning environment required	Deep Learning Inference	2.0.0/2.1.0	.dlkpack
1.6.0	2.0.0/2.1.0	Object Detection (please start the deep learning server for the Step)	1.4.0	.dlkpack
	No deep learning environment required	Deep Learning Inference (Mech-DLK 2.1.0/2.0.0)	2.0.0/2.1.0	.dlkpack
	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpack
1.6.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.6.2	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpackC/.dlkpack
1.7.0	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.2	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.7.4	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

© 2023 Mech-Mind Robotics Technologies Ltd.

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

Defect Segmentation

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.4.0	1.4.0	Defect Segmentation (please start the deep learning server for the Step)	1.4.0	.pth/.py
1.5.x	No deep learning environment required	Deep Learning Inference	2.0.0/2.1.0	.dlkpack
1.6.0	No deep learning environment required	Deep Learning Inference (Mech-DLK 2.1.0/2.0.0)	2.0.0/2.1.0	.dlkpack
	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpack
1.6.1	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpack
1.6.2	No deep learning environment required	Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.1+	.dlkpack
1.7.0	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.1	No deep learning environment required	Deep Learning Model Package CPU Inference/Deep Learning Model Package Inference (Mech-DLK 2.2.0+)	2.2.0+	.dlkpackC/.dlkpack
1.7.2	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

© 2023 Mech-Mind Robotics Technologies Ltd.

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.7.4	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.2.0+	.dlkpackC/.dlkpack

Text Detection

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.5.0+	.dlkpackC/.dlkpack

Text Recognition

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.5.0+	.dlkpackC/.dlkpack

Unsupervised Segmentation

Mech-Vision Version	Deep Learning Environment Version	Mech-Vision Step	Compatible Mech-DLK Version	Model File Extension(s)
1.8.0	No deep learning environment required	Deep Learning Model Package Inference	2.5.0+	.dlkpackC/.dlkpack

Use Models in Mech-DLK SDK

Mech-DLK SDK is a software development kit specifically designed to be used with Mech-DLK. Its main purpose is to help developers easily do deep learning inference in their software systems. With Mech-DLK SDK, developers can rapidly deploy deep learning models trained in Mech-DLK and flexibly integrate deep learning functionality into their own applications without reliance on Mech-Vision.

© 2023 Mech-Mind Robotics Technologies Ltd.

Usage Instructions

If you need to use Mech-DLK SDK, see Mech-DLK SDK User Manual for details.

You can also visit the Download Center of Mech-Mind to get related resources.

Compatibility

Currently, Mech-DLK SDK only supports the inference based on models trained by Mech-DLK (version 2.4.2 or above).

3.4. Model Iteration

After a certain period of usage, you may find that the trained model may not be applicable to certain scenarios. At this point, you should iterate the model. It is common practice to re-train the model with more data, but such an effort could reduce the overall recognition accuracy and might take a long time. Hence, it is recommended to use the Model Finetuning function for model iteration so as to maintain accuracy and save time.

General Model Iteration

1. Collect images that report poor recognition results.
2. Use Mech-DLK to open the project the model belongs to.
3. Enable the Developer Mode by clicking Settings › Options.
4. Add the collected images into the training and validation sets.
5. Label the newly added images.
6. Click Training parameter settings › Model Finetuning and then enable **Finetune**.
7. In the **Training Parameters** tab, lower the Learning rate properly. The Epochs can be reduced to 50–80.
8. Complete model training and export the model.

Super Model Iteration

1. Collect images that report poor recognition results.
2. Open Mech-DLK, create a new project, and add the Instance Segmentation module.
3. Enable the Developer Mode by clicking Settings › Options.
4. Add the collected images into the training and validation sets.
5. Label the newly added images.
6. Click Training parameter settings › Model Finetuning and then enable **Finetune**.
7. Select **Super model finetuning** and click to select the super model.
8. In the **Training Parameters** tab, lower the Learning rate properly. The Epochs can be reduced to 50–80.
9. Complete model training and export the model.

© 2023 Mech-Mind Robotics Technologies Ltd.

4. Appendix

4.1. Menu Bar

File

New Project	Create a new project
Open Project	Select and open an existing project folder
Save Project	Save changes to the current project
Save As	Save the project in a specified location
Import	Import images, folders, or datasets into the current project. Under the cascade mode, you can choose to add from the previous module. The source folder shouldn't be subject to any change during the import procedure of a folder.
Export Dataset	Export the dataset of the current project to a specified location. Under the cascade mode, only the dataset of the first module is exported.

Edit

Undo	Undo the **Labeling** action
Redo	Redo the action that you've undone

Tools

Operation Mode	You can perform batch-wise testing on the dataset under the Operation Mode. This feature is for the Defect Segmentation module and the Classification module.
Training Center	Train models in batches

Settings

Options	Change the display language and turn on/off the Developer Mode

Help

Manual	Open the web page of User Manual
Quick Guide	Quick guide to start the software
What's New	View the release notes in a web interface

© 2023 Mech-Mind Robotics Technologies Ltd.

Logs	Open the folder containing the running logs
Device Check	Check and view the CPU and GPU information of the deployment device
About	View the software version

4.1.1. Perform Dataset Testing in Batches under Operation Mode

Introduction

You can perform batch-wise testing on the dataset in the **Operation Mode**. This feature is for the **Defect Segmentation** module and the **Classification** module. It is unavailable under the cascade mode.

After model training, select Tools › Operation Mode in the menu bar to open the window of **Operation Mode**.

- **Defect Segmentation**: Once data is imported, you can select the automatic check by presetting images as OK or NG, or you can choose to perform a manual check.
- **Classification**: Once data is imported, you need to perform a manual check.

After data check, click the **Export report** button to get a report on accuracy, GPU usage, inference time, etc.

Use the Operation Mode

- Use the Operation Mode in Defect Segmentation Module
- Use the Operation Mode in Classification Module

Defect Segmentation

Steps

1. Select data source

   Regarding **Data Source**, you can select **Mech-DLK** or **Folder**.

Mech-DLK	Use all images in the current project to demonstrate the performance of the trained model.
Folder	Use massive new data for model testing. After selecting the image folder path, import all images in the folder as a new dataset. The new dataset imported will be independent of the original datasets in the project.

   If some new images were imported into the current project, in addition to the training set and validation set, these new images will also be imported when you select **Mech-DLK** as the data source.

2. Load the model

   Click **[Load Model]**. After model loading was completed, click **[Next]** to enter the inference interface.

3. Preset data

© 2023 Mech-Mind Robotics Technologies Ltd.

When **Preset as OK** or **Preset as NG** is selected, an automatic check will be performed after inference. When **Check manually** is selected, a manual check should be performed after inference.

4. Inference

   After presetting, click to start the inference. Once inference is completed, relevant data such as inference time and GPU usage will be displayed under **Running information**.

5. **Perform a manual check** (Skip this step if data are preset)

   Perform a manual check on inference results. Click when the inference result is correct and click when the inference result is incorrect. Upon the check, you can view the statistics of false positives (FPs) and false negatives (FNs) under **Running information**.

6. Export the report

   After the data check, you can click **[Export report]** to learn about accuracy, FN rate, FP rate, etc.

Running Information

- Inference time

  Display of **Image inference time** and **Average inference time**.

- Num of FN/FP images
  - When images are preset as OK, the image inferred to be NG will be counted as an FP; when images are preset as NG, the image inferred to be OK will be counted as an FN.
  - During the manual check, if the image inferred to be OK is actually NG, which means the inference result is incorrect, the image will be counted as an FN; if the image inferred to be NG is actually OK, also an incorrect inference result, the image will be counted as an FP.
- GPU usage

  Display of **Current GPU usage**.

Classification

Steps

1. Select data source

   Regarding **Data Source**, you can select **Mech-DLK** or **Folder**.

Mech-DLK	Use all images in the current project to demonstrate the performance of the trained model.
Folder	Use massive new data for model testing. After selecting the image folder path, import all images in the folder as a new dataset. The new dataset imported will be independent of the original datasets in the project.

   If some new images were imported into the current project, in addition to the training set and validation set, these new images will also be imported when you select **Mech-DLK** as the data source.

© 2023 Mech-Mind Robotics Technologies Ltd.

2. Load the model

   Click **[Load Model]**. After loading successfully, click **[Next]** to enter the inference interface.

3. Inference

   Click to start the inference. Once inference is completed, relevant data such as inference time and GPU usage will be displayed under **Running information**.

4. Perform a manual check

   Perform a manual check on inference results. Click when the inference result is correct and click when the inference result is incorrect.

5. Export the report

   After the manual check, you can click **[Export report]** to learn about the inference time, GPU usage, accuracy, etc.

Running Information

- Inference time

  Display of **Image inference time** and **Average inference time**.

- GPU usage

  Display of **Current GPU usage**.

4.2. Obtain a Trial Software License

Mech-Mind uses CodeMeter from Wibu-Systems as the license system for its software. We provide trial software licenses that do not require a license dongle.

Trial software licenses are temporary and are given only when the license dongle cannot be received in time. Once the license dongle is received, please connect it with the IPC and update software license.

Obtain a Ticket

Please contact Mech-Mind Sales to obtain a **ticket** code before proceeding. The ticket code sent to you via email is a 25-character string made up of numbers, alphabets, and hyphens.

Install CodeMeter

Please run the CodeMeter installer received from Mech-Mind to install CodeMeter.

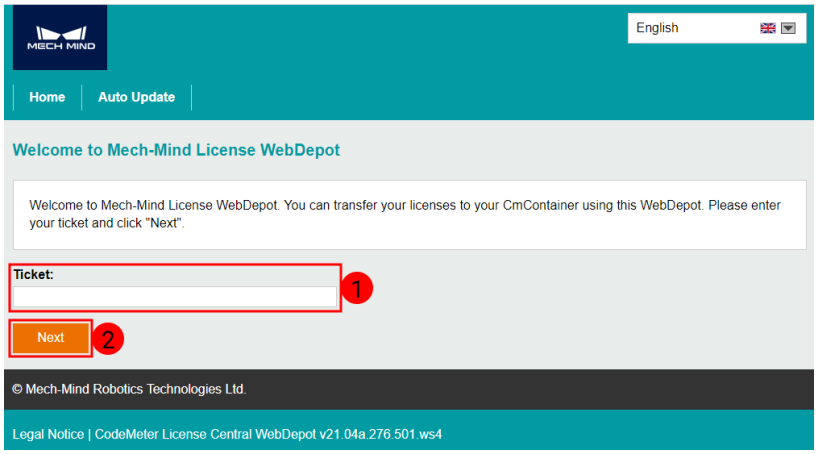
Activate the License

Please activate the license in Mech-Mind License WebDepot.

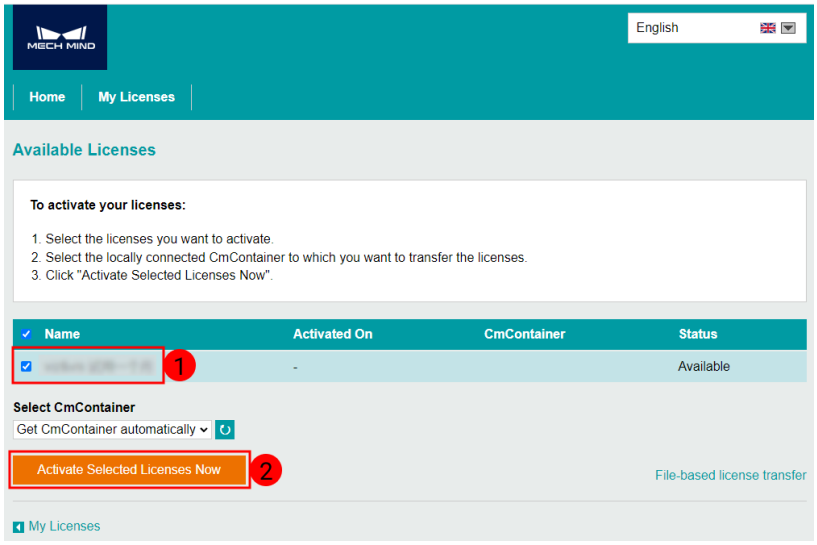
The activation process is as follows:

1. Copy and paste the ticket code into the **Ticket field**, and click **[Next]**.

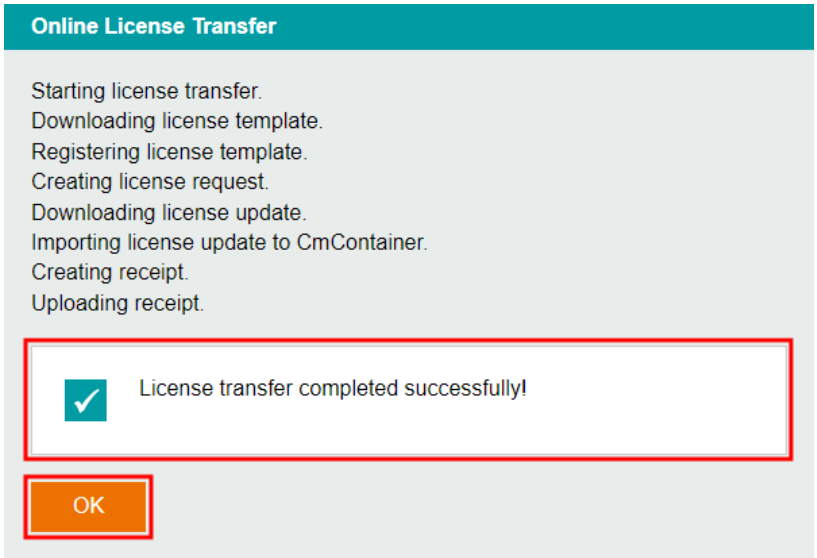
© 2023 Mech-Mind Robotics Technologies Ltd.



2. Select the license and click [Activate Selected Licenses Now] to download the license.

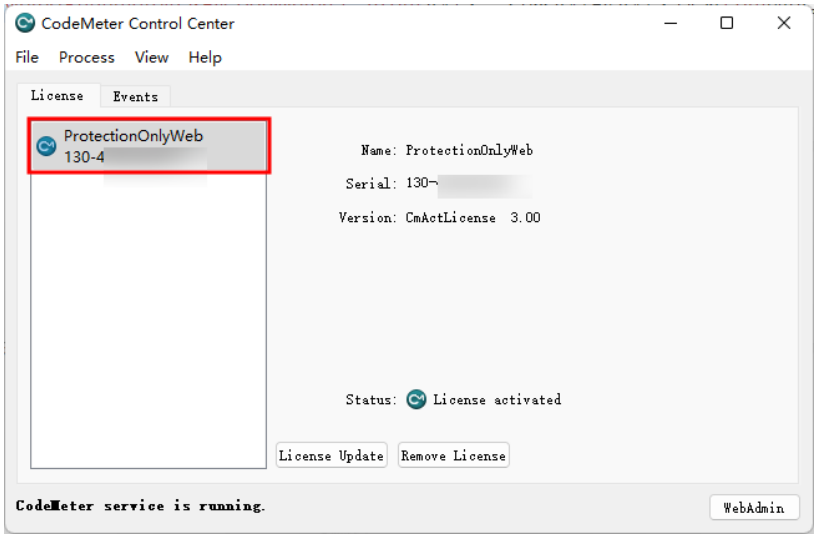


3. Once downloaded, the following information will be displayed, and then click [OK].



4. Open CodeMeter Control Center, and you should see the license displayed.

© 2023 Mech-Mind Robotics Technologies Ltd.



4.3. Update Software License

CodeMeter will notify you if your license(s) is about to expire. In such case, please contact Mech-Mind Technical Support (hereafter referred to as Mech-Mind) to request an update.

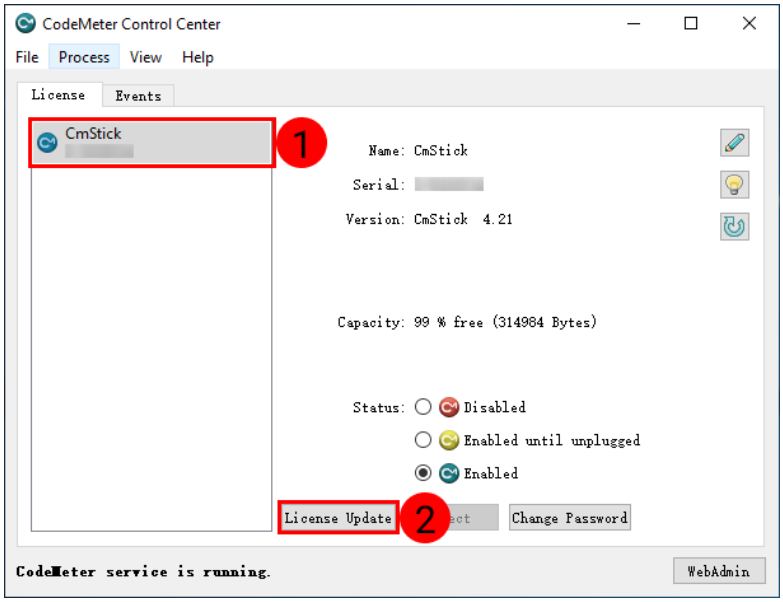
Once the update has been arranged, you'll need to export a license request file to send to Mech-Mind first. Then, Mech-Mind will send back a license update file that extends the duration of your license.

Export License Request File

Before proceeding, please make sure that:

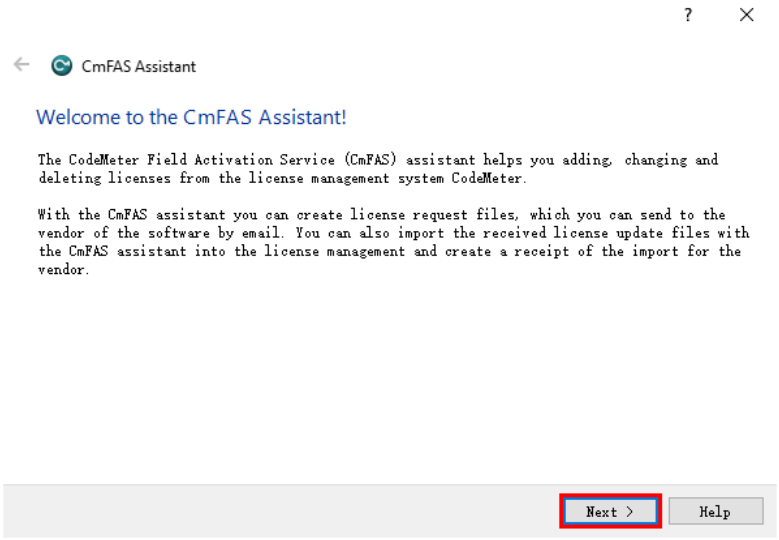
- CodeMeter is installed.
- License dongle(s) is inserted.

1. Click in the system tray to open CodeMeter Control Center.
2. Select the license dongle whose license needs to be updated, and click **[License Update]**.

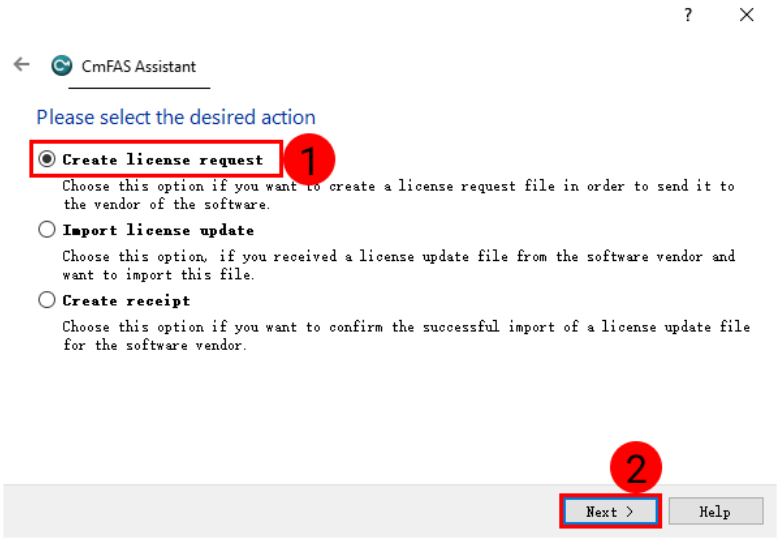


© 2023 Mech-Mind Robotics Technologies Ltd.

3. Click [**Next**] in the pop-up window.

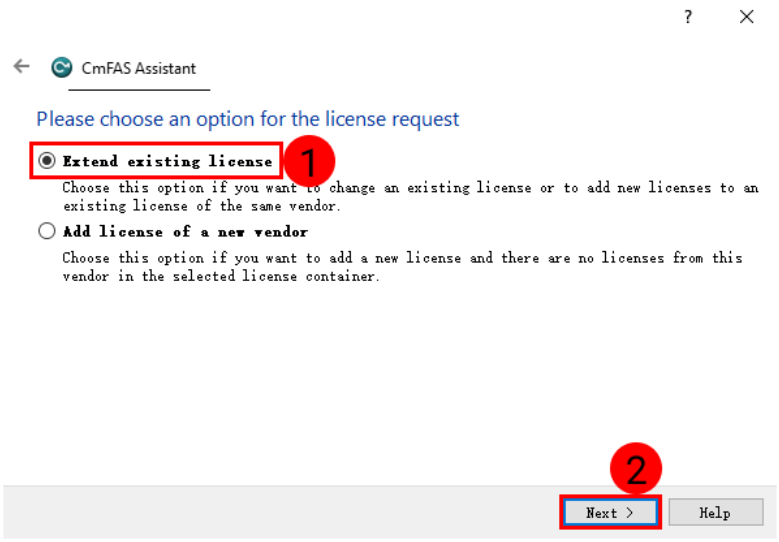


4. Select **Create license request**, and then click [**Next**].

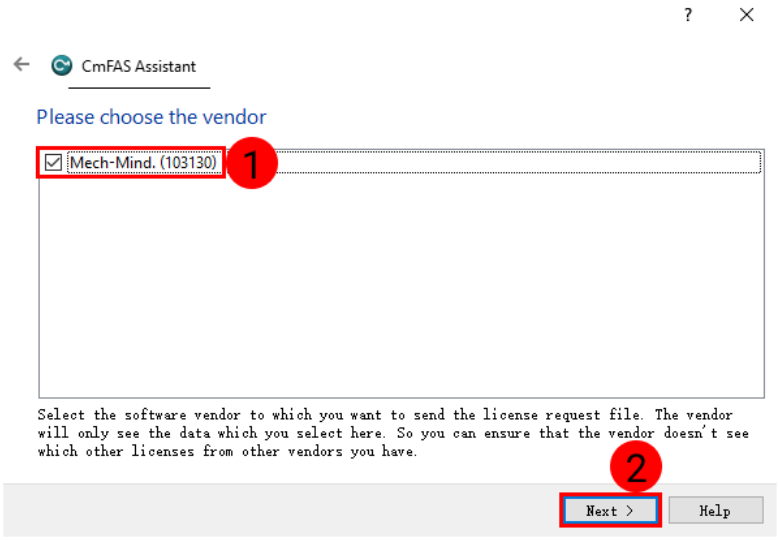


5. Select **Extend existing license**, and then click [**Next**].

© 2023 Mech-Mind Robotics Technologies Ltd.



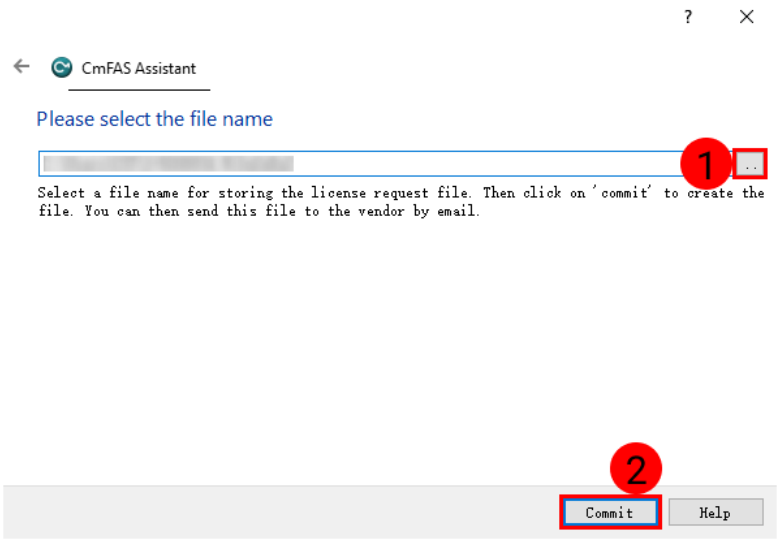
6. Choose **Mech-Mind** as the vendor, and then click **[Next]**.



The company code may differ.

7. On the next page, click **[..]** to select a location for saving the license request file, and then click **[Commit]**.

© 2023 Mech-Mind Robotics Technologies Ltd.



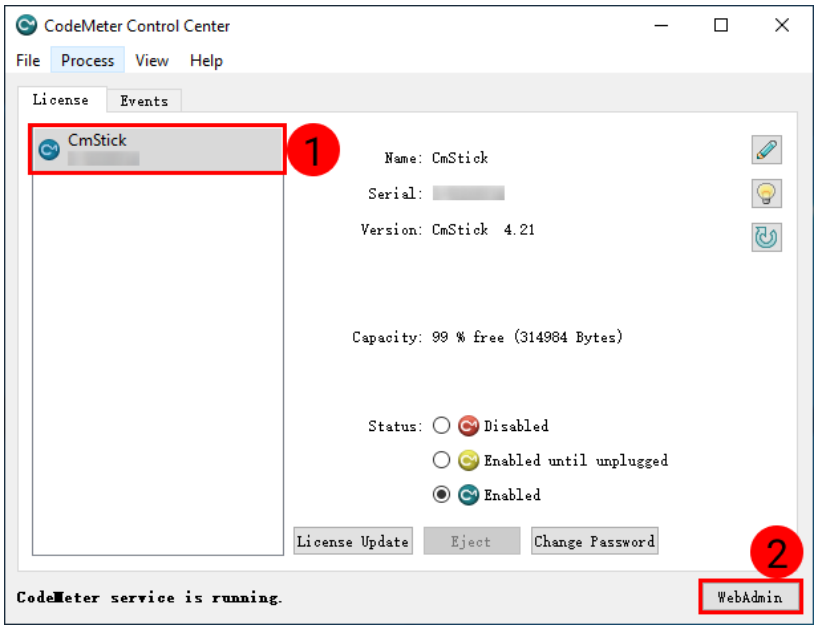
If licenses on multiple license dongles need to be updated, please repeat steps 2 to 7 to export a license request file for each dongle.

8. Send the license request file to Mech-Mind.

Update the License

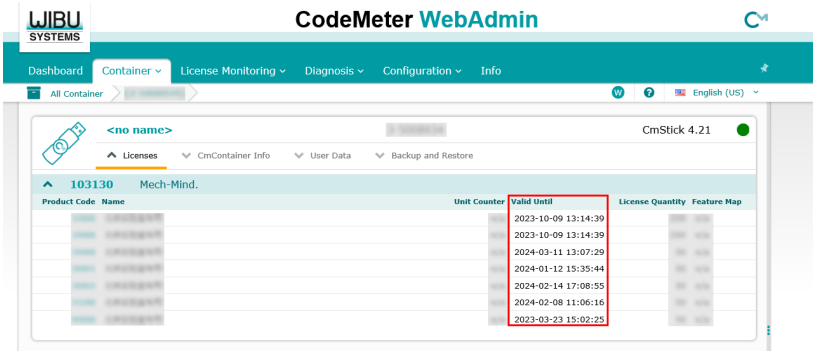
After you send the license request file, Mech-Mind will send back a license update file in WIBUCMRAU format. Double-click this file to update the corresponding license. Follow the steps below to check if the license has been successfully updated.

1. Click in the system tray to open CodeMeter Control Center.
2. Select the license dongle you'd like to check, and then click **[WebAdmin]** in the lower right.



3. The **CodeMeter WebAdmin** webpage will be opened, and you can check the date under **Valid Until** to see if the license has been updated.

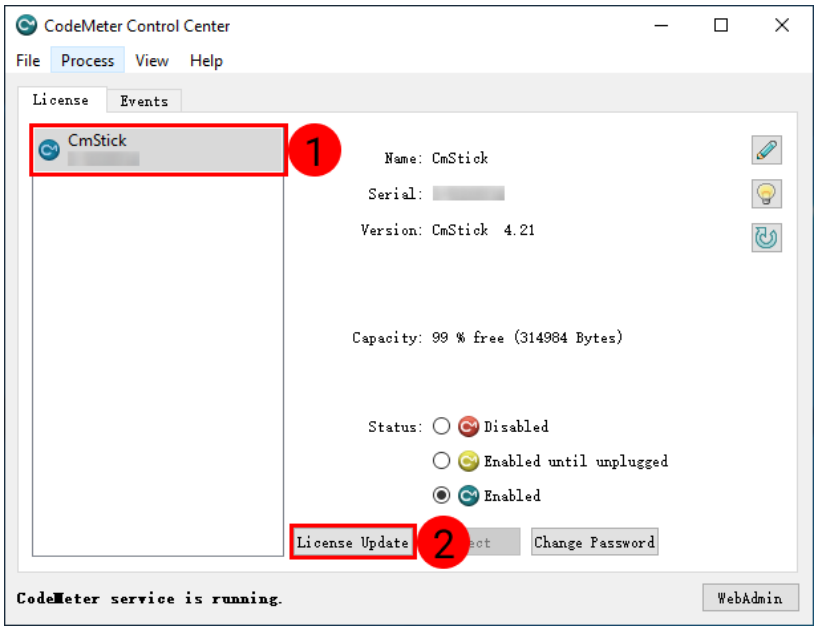
© 2023 Mech-Mind Robotics Technologies Ltd.



- The **Valid Until** date for a perpetual license is **n/a**.
- You can select a different license dongle to view by clicking on the **[Container]** tab at the top.

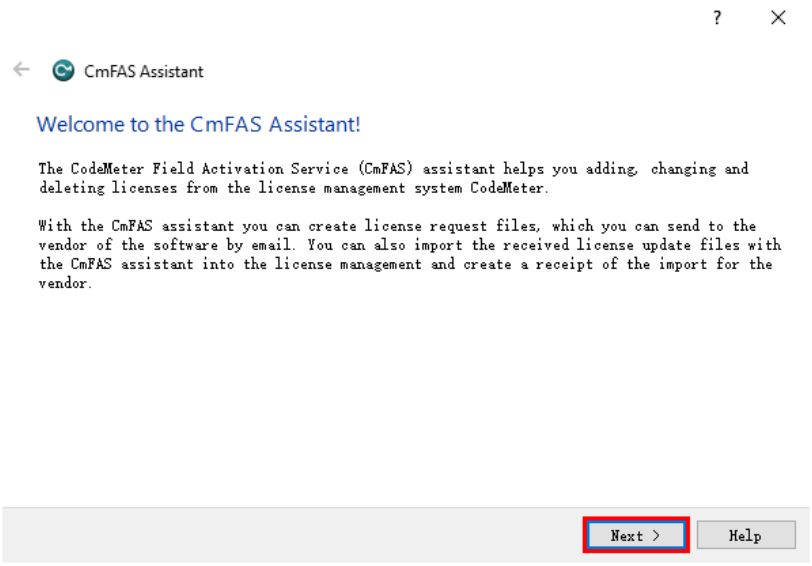
If for some reason, double-clicking on the WIBUCMRAU file does not update your license, please follow the steps below to manually import the license update file.

1. Click in the system tray to open CodeMeter Control Center.
2. Select the license dongle whose license needs to be updated, and click **[License Update]**.

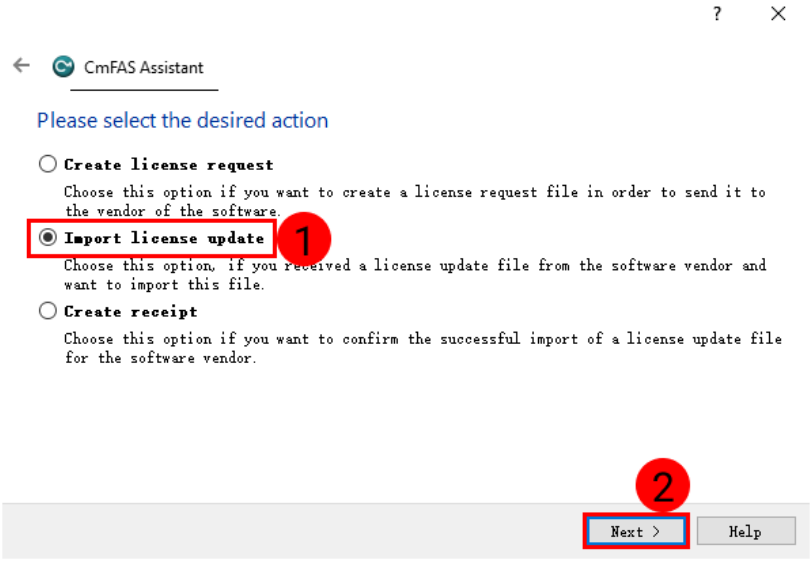


3. Click **[Next]** in the pop-up window.

© 2023 Mech-Mind Robotics Technologies Ltd.

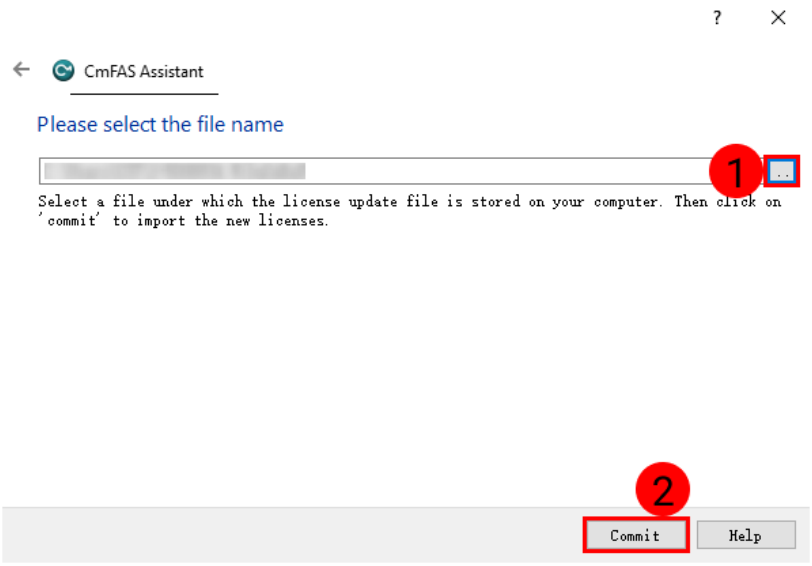


4. Select **Import license update**, and then click **[Next]**.



5. Click **[..]** to select the license update file, and then click **[Commit]**.

© 2023 Mech-Mind Robotics Technologies Ltd.



4.4. Keyboard Shortcuts

In the lower right corner of the selection region, click to open the floating window of keyboard shortcuts. The window can be placed anywhere on the interface and will not be hidden during other operations. Click [×], click , or press Esc on the keyboard to close the floating window. In addition, you can hover over the buttons of labeling tools to view the corresponding keyboard shortcuts.

All the keyboard shortcuts work in both lowercase and uppercase letters.

Project

Feature	Shortcut	Description
New Project	Ctrl + N	Create a new project.
Save the project	Ctrl + S	
Open Project	Ctrl + O	

Edit

Feature	Shortcut	Description
Undo	Ctrl + Y	
Redo	Ctrl + Z	
Copy	Ctrl + C	
Paste	Ctrl + V	
Select All	Ctrl + A	
Delete	Delete	Please move the cursor to the labeling section / dataset section and select the item to delete.

© 2023 Mech-Mind Robotics Technologies Ltd.

Tools

Feature	Shortcut	Description
Ellipse Tool	L	
Polygon Tool	P	
Free Rectangle Tool	D	
Rectangle Tool	R	
Brush Tool	B	
Autofill Lasso Tool	A	
Labeling Eraser Tool	E	
Auxiliary Labeling Tool	T	
Smart Labeling Tool	M	
Apply current smart labeling	Enter	
Cancel current smart labeling	Q	
Mask Polygon Tool	Shift + P	
Finish Polygon Drawing	Enter	
Mask Brush Tool	Shift + B	
Mask Lasso Tool	Shift + A	
Mask Eraser Tool	Shift + E	
Template Tool	C	
Select template	Ctrl + left click	
Select template	Shift + scroll wheel	
Grid Cutting Tool	U	
Grid Selection Tool	I	
Feature Group Labeling Tool	F	
ROI Tool	O	
Selection Tool	S	
Adjust Template Tool/Brush size	⬅, ➡	
Switch between neighboring images	⬆, ⬇	
Text Recognition Tool	G	

View

Feature	Shortcut	Description
Switch labeling display	Ctrl + L	

© 2023 Mech-Mind Robotics Technologies Ltd.

Feature	Shortcut	Description
Switch validation display	Ctrl + P	

4.5. FAQ

How to troubleshoot the reason why a defect segmentation model is not effective?

a. Check the labels for errors.

b. Check that all kinds of defects are included in the training set.

c. Check that the input image size is reasonable. If the defect is too small it may not be effective for training the model.

When to use the Defect Segmentation module and the Unsupervised Segmentation module?

Generally speaking, both modules can recognize the defect areas in images, but they remain quite different from each other. You may want to choose one by the following considerations:

- The **Defect Segmentation** module aims to segment the defects; in other words, a high requirement of accuracy is put on this module in the position, shape, and size of defects. The **Unsupervised Segmentation** module, however, is designed to judge whether there is any defect in an image and display the possible areas with defects for NG images.
- For the former, all types of defects should be labeled in the labeling process. The latter, however, poses no such requirement as defect labeling, and NG images are unnecessary for model training, namely that only OK images will be used as the training set.
- The latter can only show a rough defect area of an image and cannot finely segment a defect. If you want to segment defects in an image, please use the **Defect Segmentation** module.

Does it work if I simulate changes in lighting conditions during data collection by manually adjusting the camera exposure or adding supplemental light?

No. Simulated lighting conditions may not reflect the actual conditions accurately, and thus image data collected under such conditions cannot provide accurate object features to train the model. Therefore, if the lighting conditions on-site change over the day, please collect image data respectively under different conditions.

The camera is fixed, and the incoming objects' positions vary slightly. Does it work if I simulate the position changes of the objects by moving the camera during data collection?

No. The camera should be fixed in position before any data collection. Moving the camera during data collection will affect the extrinsic parameters of the camera and the training effect. Setting a larger ROI can help fully capture the changes in object position.

If the previously used camera has unsatisfactory imaging quality and is replaced by a new camera, is it necessary to add the images taken by the old camera to the dataset?

No. After camera replacement, all data used for model training should come from the new camera. Please conduct data collection again using the new camera and use the data for training.

Will changing the background affect model performance?

Yes. Changing the background will lead to recognition errors, such as false recognition or failure to recognize a target object. Therefore, once the background is set in the early stage of data collection, it is best not to change the background afterward.

© 2023 Mech-Mind Robotics Technologies Ltd.

Does it work if I use the image data collected with different camera models at different heights together to train one model?

Yes, but please work on the ROI settings. Select different ROIs for images taken at different heights to reduce the differences among images.

For highly reflective metal parts, what factors should I consider during data collection?

Please avoid overexposure and underexposure. If overexposure in parts of the image is inevitable, make sure the contour of the object is clear.

If the model performs poorly, how to identify the possible reasons?

Factors to consider: quantity and quality of the training data, data diversity, on-site ROI parameters, and on-site lighting conditions.

a. Quantity: whether the quantity of training data is enough to make the model achieve good performance.
b. Quality: whether the data quality is up to standard, whether images are clear enough and are not over-exposed/under-exposed.
c. Data diversity: whether the data cover all the situations that may occur on-site.
d. ROI parameters: whether the ROI parameters for data collection are consistent with those for the actual application.
e. Lighting conditions: Whether the lighting conditions during the actual application change, and whether the conditions are consistent with those during data collection.

How to improve unstable model performance due to complicated on-site lighting conditions, e.g., objects are covered by shadows?

Please add shading or supplemental light as needed.

Why does the inconsistency between the ROI settings of on-site data and training data affect the confidence values of instance segmentation?

The inconsistency will result in objects being out of the optimal recognition range of the model, thus affecting the confidence. Therefore, please keep the ROI settings of the on-site data and training data consistent.

What are super models for cartons?

Super models (click here to download) are provided for carton palletizing and depalletizing. They can be used directly for most project sites to correctly segment most cartons without collecting additional image data or training.

What scenarios can the super models for cartons be applied to?

It is suitable for palletizing/depalletizing boxes of single or multiple colors and surface patterns. However, please note that this Super Model is only applicable to boxes placed in horizontal layers and not at an angle to the ground.

How to collect data for the Super Model for boxes?

Please test the Super Model first. If it cannot segment correctly sometimes, collect about 20 images of situations where the model does not perform well and train the Super Model.

ROI position deviations may occur when opening old projects with the newer-version Mech-DLK.

The ROI will be corrected after you click **[Validate]**.

When training a model in Mech-DLK, what should I do if the error message “ModuleNotFoundError: No module named ‘onnxruntime’” shows?

© 2023 Mech-Mind Robotics Technologies Ltd.

Go to the “users” folder in OS (C:) and open the folder of the current user of the computer. Check if the folder “AppData/Roaming/Python/Pythong36/site-packages” is empty. If not, please delete all contents in the folder.

Whether AMD CPUs can run CPU models?

AMD CPUs do not support running CPU models.

© 2023 Mech-Mind Robotics Technologies Ltd.