Intel® Training and Learning Suite 2.0 (Intel® TLS2.0)

User Guide

November 2020

Intel Confidential

Document Number: 631070-1.2

Contents

Figures

Revision History

Date	Revision	Description
November 2020	1.2	Added Chapter 3 Section 3.3 Native Environment Installation
November 2020	1.1	Added Chapter 5.0 Frequently Asked Questions.
September 2020	1.0	Initial release.

§

Terminology

Table 1. Terminology

Term	Description
API	Application Programming Interface
CVAT*	Computer Vision Annotation Tool
CPU	Central Processing Unit
DL	Deep Learning
EIS	Edge Insight Software
E2E	End-to-end
GUI	Graphical User Interface
OpenVINO™ toolkit	Open Visual Inferencing and Neural Network Optimization
SSL	Secure Sockets Layer
TLS	Training and Learning Suite
UI	User Interface
IoT	Internet of Things

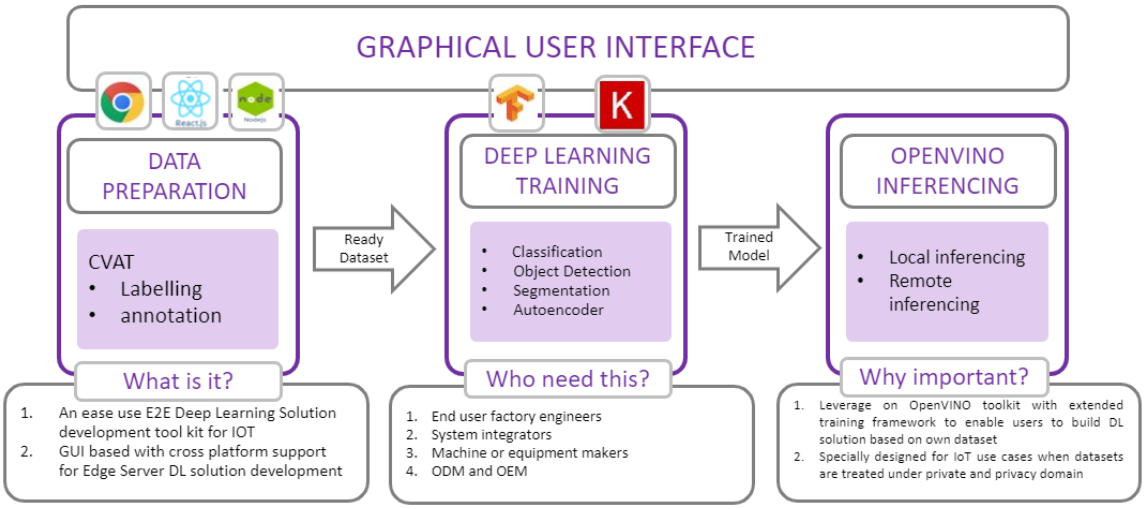
1.0 Introduction

The Intel® Training and Learning Suite 2.0 is a next generation of TLS1.0 with significant changes at the back-end and front-end. The goal is to enable Intel Customers with more popular tools and better user experience. TLS2.0 integrates the CVAT Annotation Tool, which is a popular annotation tool for data scientists to label uploaded images. The dataset created can be easily linked up with the TLS project creation and data scientists can start model training with TensorFlow* in just a few clicks.

1.1 Intel® TLS2.0 Architectures

TLS 2.0 comes with advanced features compared to TLS1.0. The CVAT feature is now enabled for data preparation that includes labeling and annotation tools. It is an easy to use the end-to-end (E2E) Deep Learning solution development toolkit for IoT. The GUI is also multi platform-based, and it can support Edge Server DL Solution development. After the data preparation, the data can be trained using 4 tasks, which are *Classification*, *Object Detection*, *Segmentation*, and *Auto-encoder*. The training will be performed using the Keras* API and TensorFlow*. TensorFlow is an end-to-end open-source platform for machine learning. It's a comprehensive and flexible ecosystem of tools, libraries, and other resources that provide workflows with high-level APIs. Keras, on the other hand, is a high-level neural networks library, which is running on the top of TensorFlow, Microsoft's* CNTK, and Theano*. Using Keras in deep learning allows for easy and fast prototyping as well as running them seamlessly on both CPU and GPU. The trained model can do the inferencing locally or remotely using the OpenVINO™ toolkit. The OpenVINO™ toolkit has an extended training framework to enable the users to build DL solutions using their own dataset.

Figure 1. TLS2.0 Overview of Major Components



1.2 TLS 2.0 Features List and Training Models

Figure 2. TLS2.0 New Features List

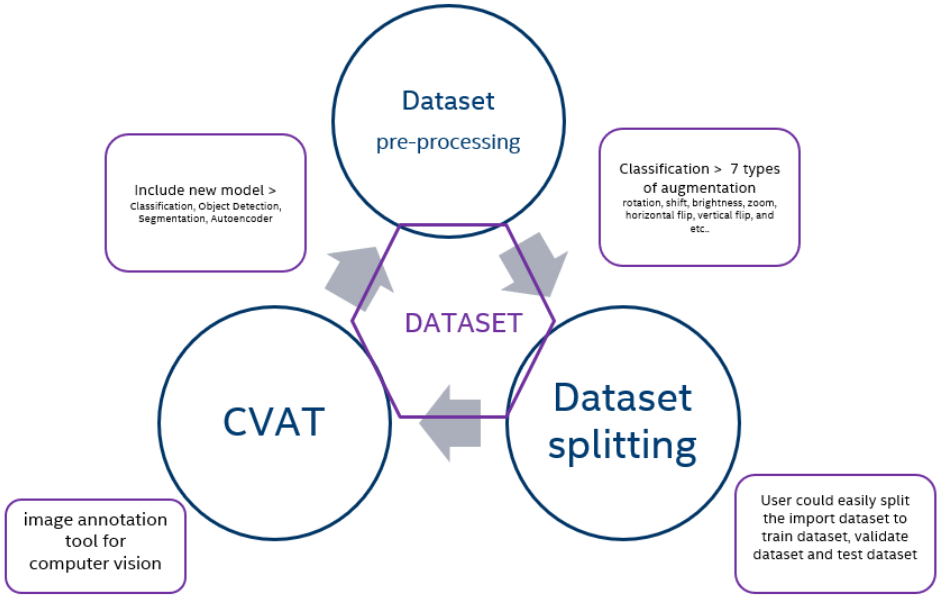
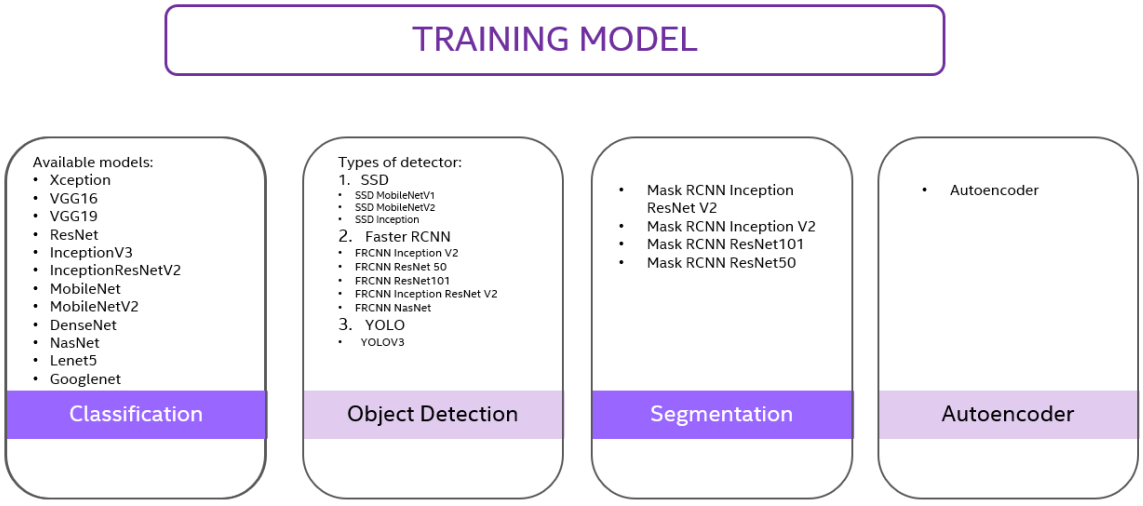
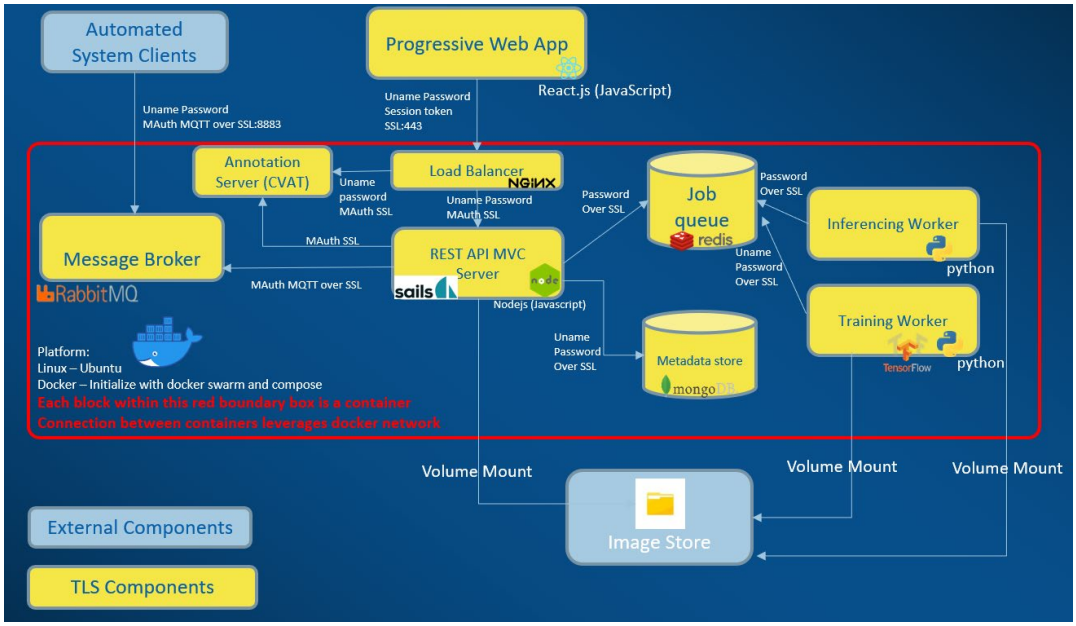


Figure 3. TLS2.0 Training Models



1.3 Security Information

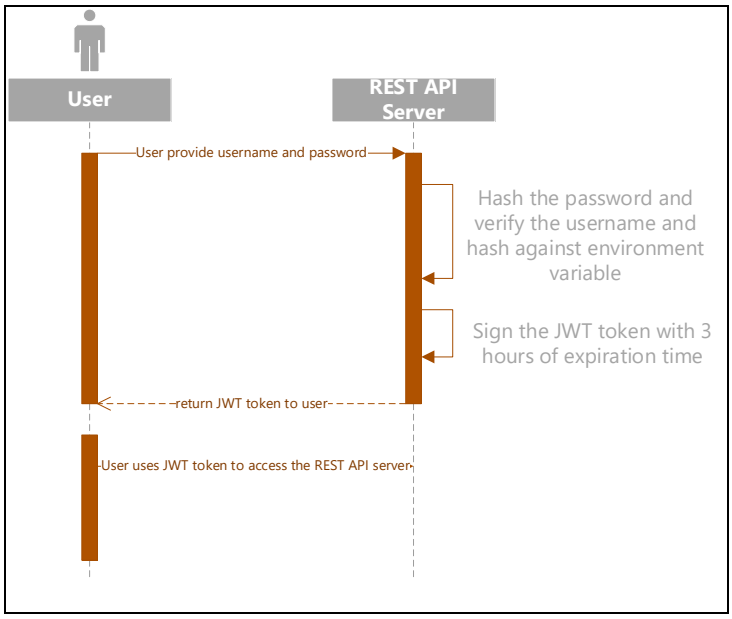
Figure 4. TLS2.0 Security Architecture



TLS 2.0 is a Dockerized web application that simplifies the process of training custom deep learning models based on the user's own datasets. It is a GUI-based application that allows users to upload their dataset, annotate data, and train their models. It also comprises of APIs for automated system clients to retrieve trained model and upload their inferencing results.

There are a lot of cryptographic features for security implementation in this application. The objective is to prevent non-privileged users from accessing the solution. It is also to prevent non-privileged software from impersonating as the load balancer to intercept or modify the user's dataset, to prevent unauthorized software from connecting to the message broker, and to detect impersonation of message broker in automated system clients. Some of the cryptographic uses are encryption, hashing, MAC-ing, signing, and key derivation. It is to make sure it provides the specific algorithm details such as key size, mode, and padding scheme. The figure below shows an example of authentication flow.

Figure 5. Authentication Flow

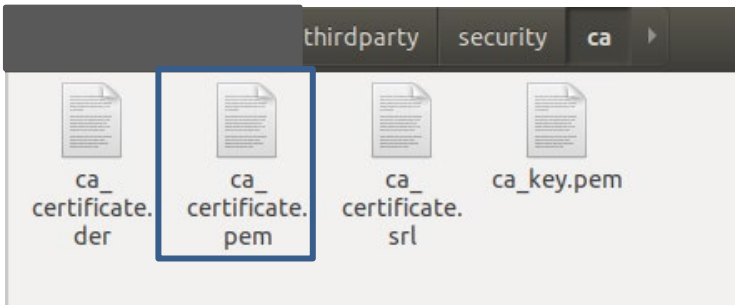


1.3.1 SSL and HTTPS Security Implementation

The Secure Sockets Layer (SSL) is a common building block for encrypted communications between clients and servers. It's possible that an application might use SSL incorrectly, such that malicious entities may be able to intercept an app's data over the network. To help user ensure that this does not happen to the application, we provide the guide to secure the HTTPS browser. We need to add a trusted CA certificate to the browser.

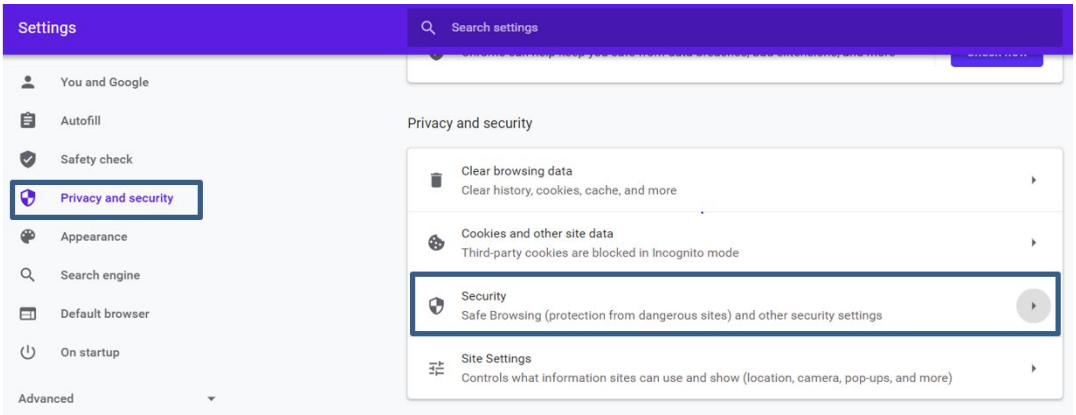
1. First, go inside the ***[work_dir]/training-learning-suite-2.0/thirdparty /security/ca*** folder that we installed and copy the ***ca_certificate.pem*** file to your host machine.

Figure 6. Certificate Folder



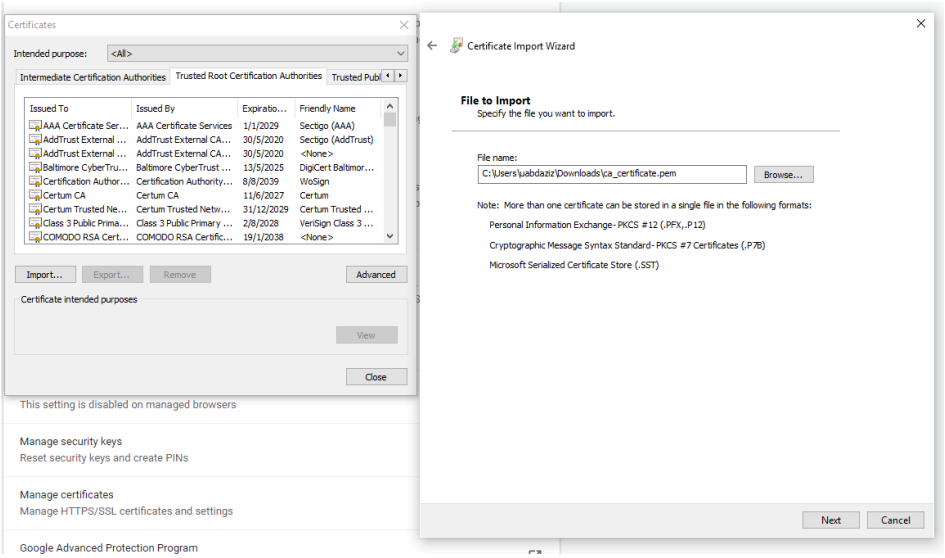
2. Then, go the web browser setting and choose the **Privacy and security** tab, then select **Security**.

Figure 7. Privacy and Security Webpage



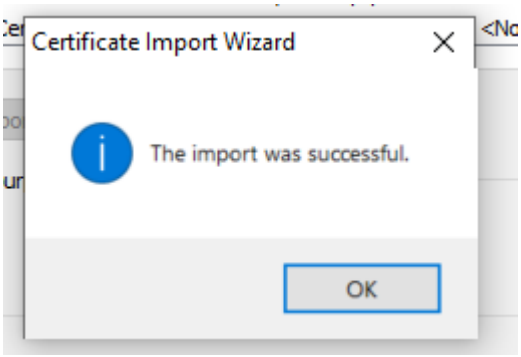
3. Next, click on **Manage certificates**, where the task will pop up, and the user can click the Trusted Root Certification Authorities to import the certification file that has been copied previously.

Figure 8. Import the Certificate



4. Once you have successfully imported the certificate file, a confirmation notice will appear.

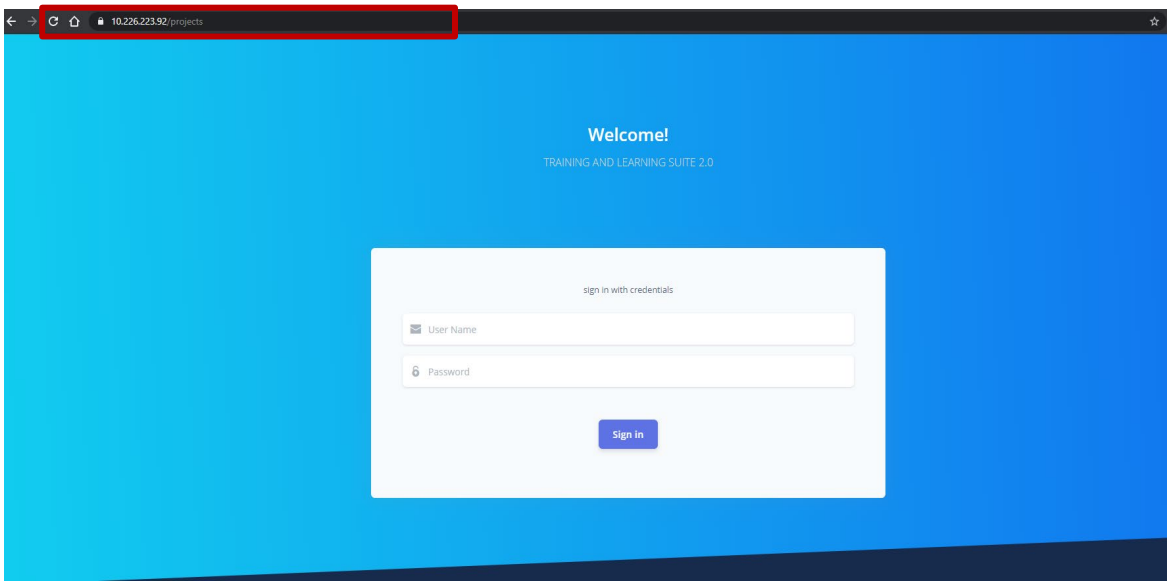
Figure 9. Successful Import Certificate Notice



5. The TLS2.0 is now secured, and ready for use.

Note: Empty up your cache to be able to access TLS2.0 in secure mode.

Figure 10. TLS2.0 Secured



§

2.0 System Set Up and Requirements

2.1 Required Hardware

i. 6th Generation Intel® Core™ processor onwards, OR
6th Generation Intel® Xeon® processor onwards

ii. At least 32GB RAM

iii. An internet connection

iv. Ubuntu* 18.04

Note: *** input images are auto resized; no manual resizing required.

Recommended training batch size for 32GB / 64GB system	System with 32GB Memory	System with 64GB Memory	Validated Image Resolution
Training Parameters	Batch Size	Batch Size	
Classification:			224 x 224*** 299 x 299***
Xception *	32	32	
VGG16 *	32	32	
VGG19*	32	32	
ResNet*	32	32	
InceptionV3*	32	32	
InceptionResNetV2*	32	32	
MobileNet*	32	32	
MobileNetV2*	32	32	
DenseNet*	32	32	
NasNet*	32	32	
Lenet5*	32	32	
GoogLeNet*	32	32	
Object Detection:			300 x 300***
SSD MobileNetV1	32	32	
SSD MobileNetV2	32	32	
RCNN Inception V2	32	32	
FRCNN ResNet 50	32	32	
FRCNN ResNet101	8	16	
FRCNN Inception ResNet V2	8	16	
FRCNN NasNet	2	2	
YOLOv3*	32	32	

Segmentation:			150 x 300***
Mask RCNN Inception ResNet V2	2	4	
Mask RCNN Inception V2	4	8	
Mask RCNN ResNet101	4	8	
Mask RCNN ResNet50	4	8	
Auto-encoder:			64 x 64***
Auto-encoder	32	32	

2.2 Prerequisite Software Configuration

- Ubuntu* 18.04 version
- OpenVINO™ toolkit
- Docker
- TensorFlow Framework V1.14
- CVAT Annotation Tool

- [Optional] *setup **proxy for docker***.

  Note: If proxy server is used, this setting should be implemented.

  1. Add environment settings as below:
     - Set http_proxy to http:// user:pass@my.proxy.server_port
     - Set https_proxy to http:// user:pass@my.proxy.server_port
     - Example command:

       export http_proxy=http://<user.pass>@<company>:<server_port>

§

3.0 Intel® TLS2.0 Installation and Setup

3.1 Installation and Setup

The setup script located in ***[work_dir] /training-learning-suite-2.0/thirdparty/setup_TLS_docker.sh*** is designed in a way to be a single-step process (to complete all dependencies packages download and installation).

List of major software components to be download and install:

- OpenVINO™ toolkit
- Docker*
- TensorFlow* Framework V1.14
- CVAT* Annotation Tool

3.2 TLS2.0 Auto-Installation Step-by-Step Guide for Docker Environment

User is highly recommended to install using the docker installation. User needs to follow the following guide to install it.

1. Set up proxy setting

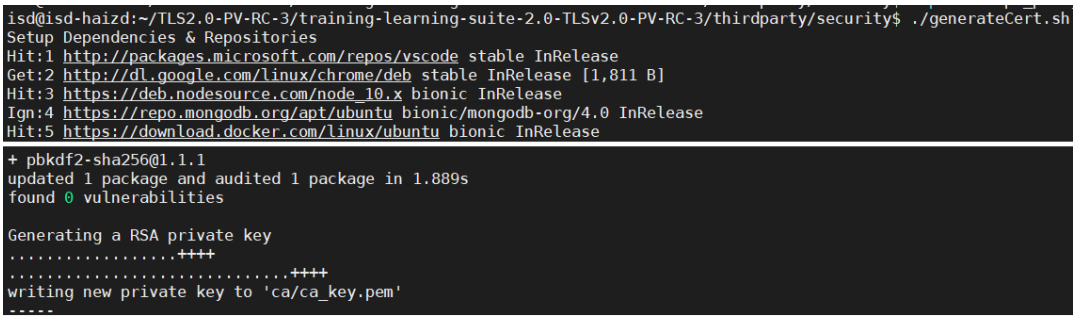
   Example:
   ```
   export http_proxy=http://<user.pass>@<company>:<server_port>
   ```

2. Enable SSL by go to ***[work_dir]/training-learning-suite-2.0/thirdparty/security*** folder.

3. Execute script from this location:

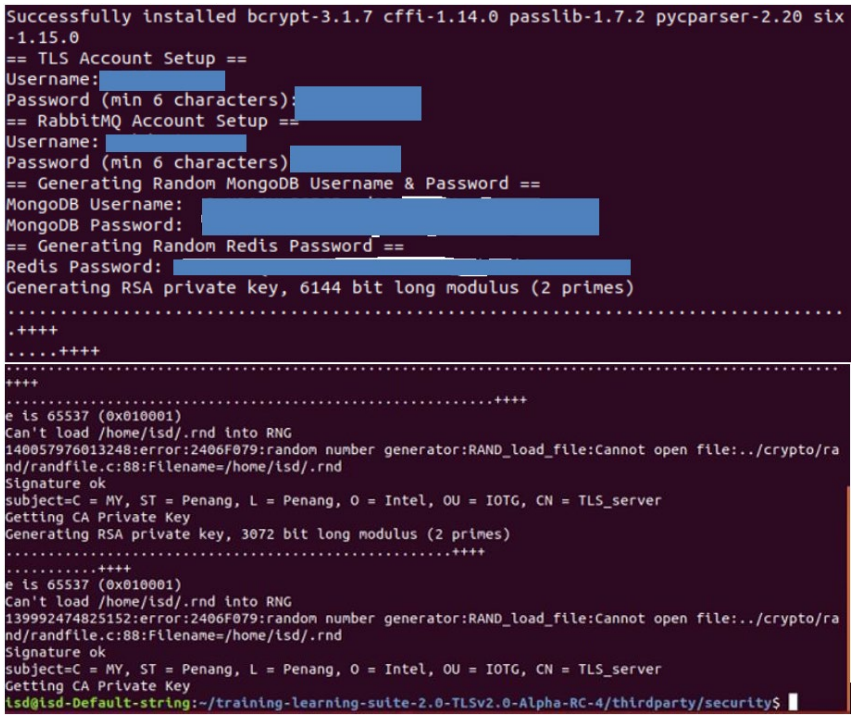
   ```
   $ cd [work_dir]/training-learning-suite-2.0/thirdparty/security/

   $ ./generateCert.sh
   ```

Figure 11. TLS2.0 Certificate Generate



4. User needs to key in the username for TLS and RabbitMQ* account setup. The username and password for MangoDB* will be generated automatic for later step in installing phase.

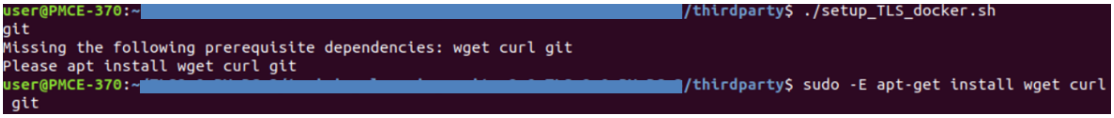
Figure 12. TLS2.0 Account Setup



5. If user installs TLS2.0 on a new fresh system, user needs to install *wget cURL git* first by running the command below:

```
$ sudo -E apt install wget curl git
```

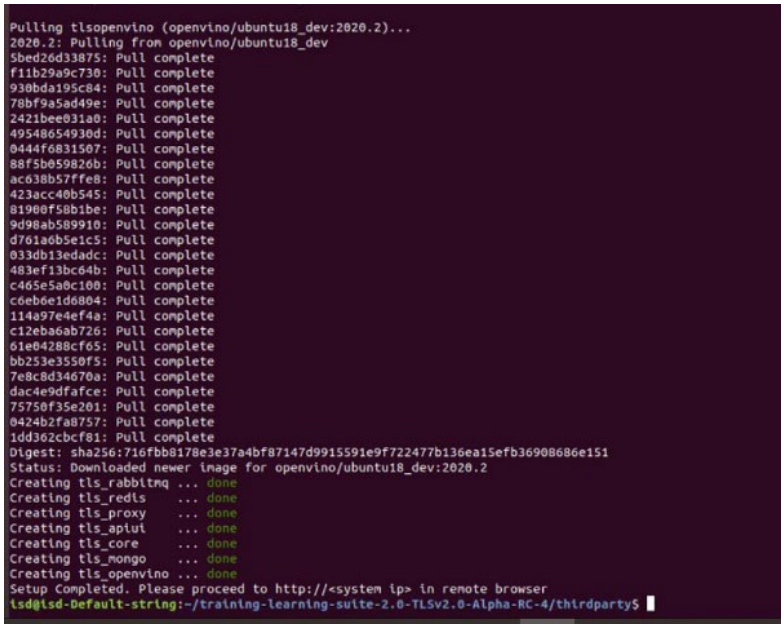
Figure 13. Install *wget cURL Git*



6. Then, execute the script from location:

```
$   cd [work_dir]/training-learning-suite-2.0/thirdparty/
$ ./setup_TLS_docker.sh
```

Figure 14. TLS2.0 Setup Completed



3.3 TLS2.0 Auto-Installation Step-by-Step Guide for Native Environment

The native installation is for **development** purpose. If user does not have the intention to modify or customize TLS 2.0 functionalities through own development efforts, then user should install the TLS2.0 in docker mode (refer to Section 3.2) and use as it is. User needs to follow the following steps to install TLS2.0 in Native environment.

1. Set up proxy setting

   Example:
   `export http_proxy=http://<user.pass>@<company>:<server_port>`

2. Enable SSL by going to ***[work_dir]/training-learning-suite-2.0/thirdparty/security*** folder.

3. Execute script from this location:

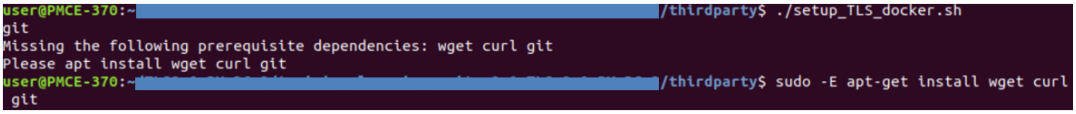
```
$ cd [work_dir]/training-learning-suite-
2.0/thirdparty/security/

$ ./generateCert.sh
```

4. User will be prompt to setup Account for TLS. User needs to key in the username for TLS and RabbitMQ* account setup. The username and password for MangoDB* will be generated automatic later at the installing phase.

5. If user installs TLS2.0 on a fresh new system, user needs to install *wget cURL git* first by running the command below:

```
$ sudo -E apt install wget curl git
```

Figure 15. Install wget cURL Git



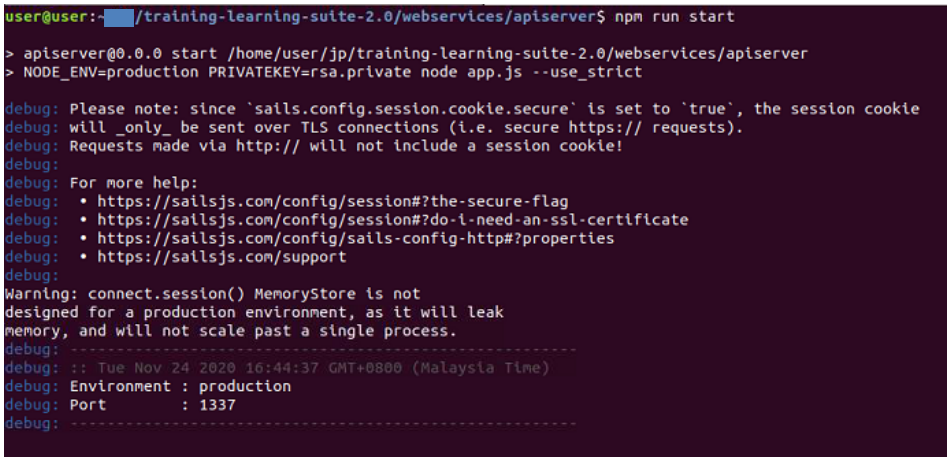
6. Then, execute the script from location:

```
$   cd [work_dir]/training-learning-suite-2.0/thirdparty/
$ ./setup_TLS_native.sh
```

7. After the installation is completed, user needs to go to the following repository to run the TLS.20

```
$   cd [work_dir]/training-learning-suite
2.0/webservices/apiserver

$ npm run start
```

Figure 16. Output on apiserver Terminal



8. Then, go to the following repository to execute the command below to build up the UI.

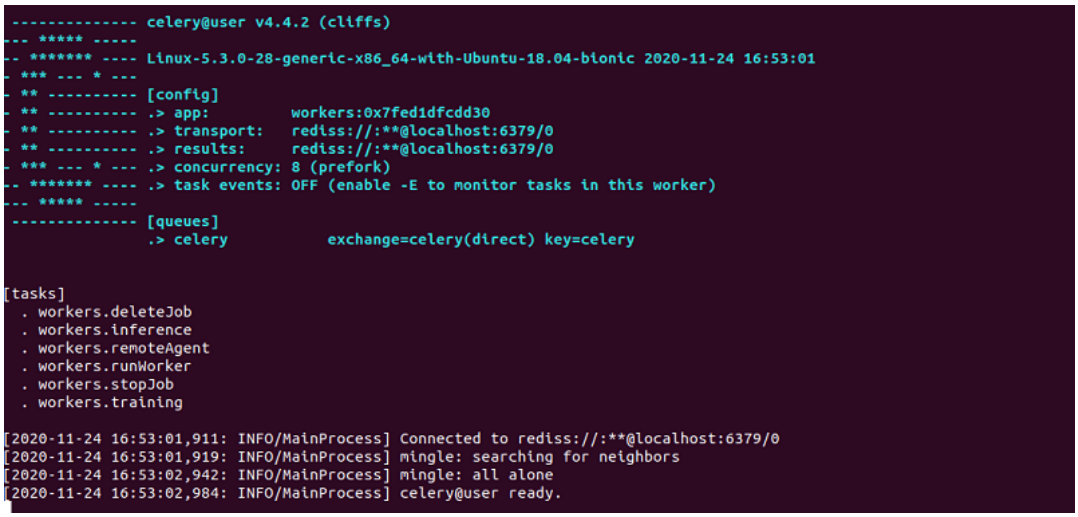
```
$    cd [work_dir]/training-learning-suite
2.0/webservices/ui

$ npm run start
```

Figure 17. Output on UI Terminal

9. Lastly, user needs to go to the following repository to execute the command below.

```
$    cd [work_dir]/training-learning-suite
2.0/nnframework/tf

$ ./startNN.sh
```

Figure 18. Successfully Run the File on Terminal

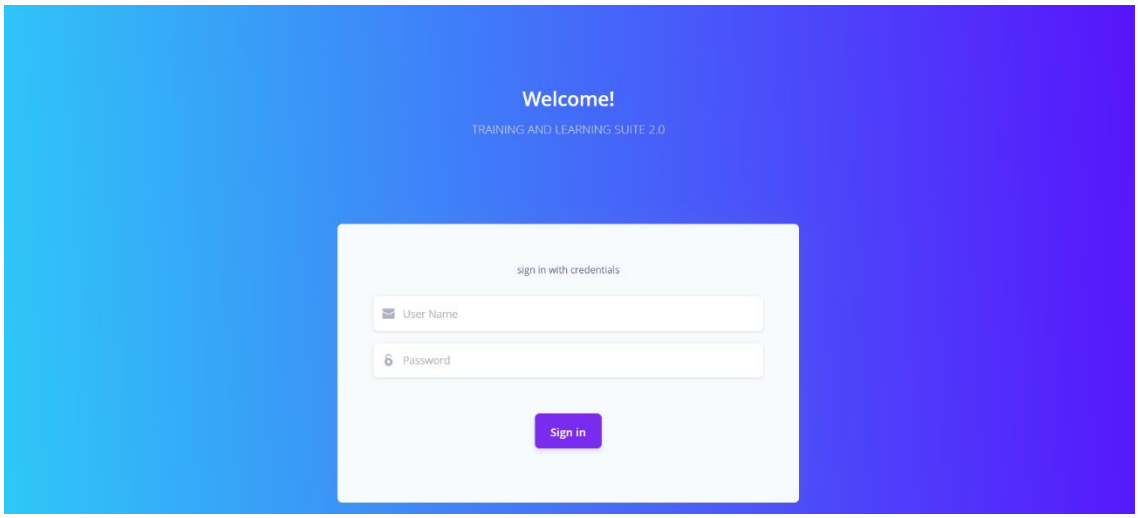


3.4 TLS2.0 Login Page

After the installation and setup is completed, the user can proceed to the UI page on a browser at **https://<System IP>**.

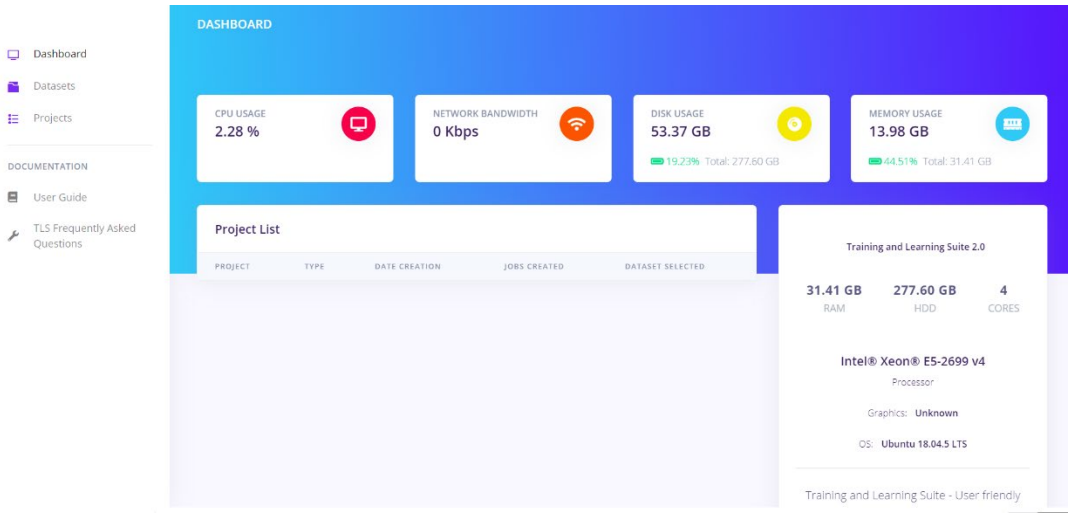
The Login Page will appear. User needs to enter the TLS account username and password that was created during the installation process.

Figure 19. TLS2.0 Login Page



After that, the dashboard of the TLS 2.0 will appear, as shown in the figure below and user may proceed to do the training.

Figure 20. TLS.20 Dashboard



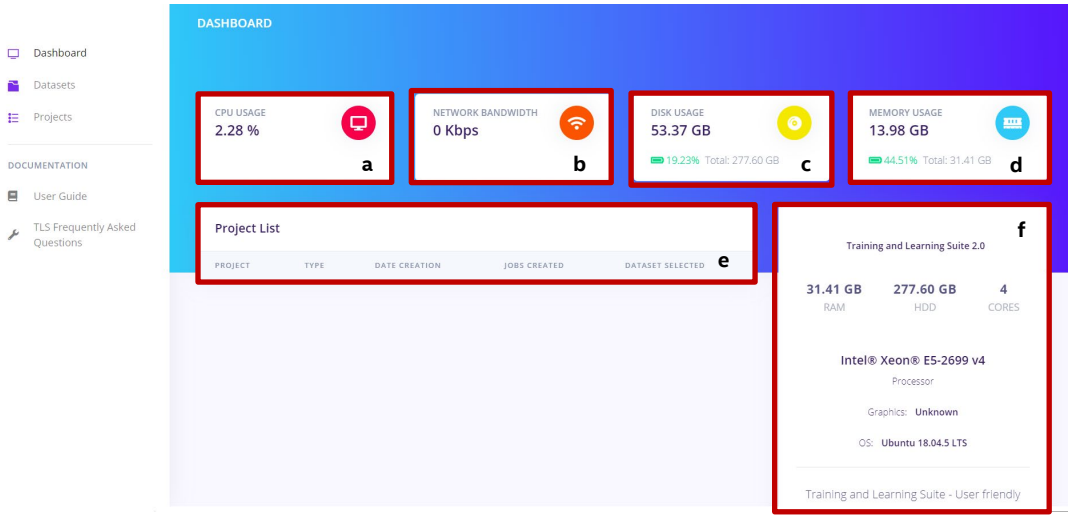
§

4.0 Model Training & Inferencing with TLS2.0

4.1 Introduction to TLS2.0 UI

The dashboard of the TLS2.0 can be accessed after user logging in. It will show several applications that are running on the installed system.

Figure 21. Right-side Dashboard Information



The red boxes on the right-side of the dashboard represent:

a. Box '**a**' shows the loading CPU usage in the system.

b. Box '**b**' shows the loading network bandwidth in the system.

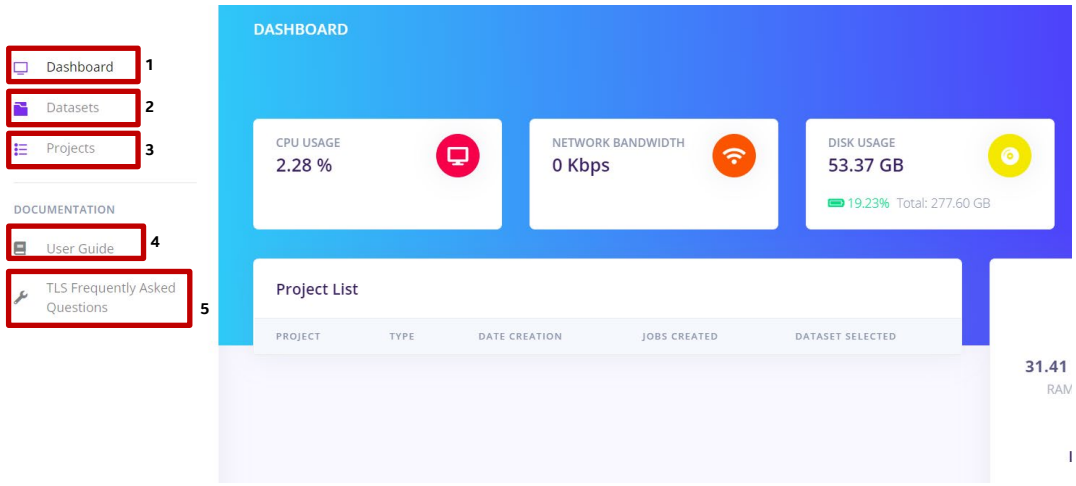
c. Box '**c**' shows the loading the usage of the disk that being used in TLS2.0

d. Box '**d**' shows the memory usage in the system.

e. Box '**e**' shows the project list that the user created on TLS2.0.

f. Box '**f**' shows the information of RAM, Hard Disk, cores, processor, graphics, and operating system information.

Figure 22. Left-side Dashboard Information



The box on the left-side of the dashboard represents:

a. Box '**1**' shows the tab for TLS2.0 dashboard.

b. Box '**2**' shows dataset tab which users can upload their dataset and do annotation on CVAT.

c. Box '**3**' shows the Project tab which users can create a project and train a model.

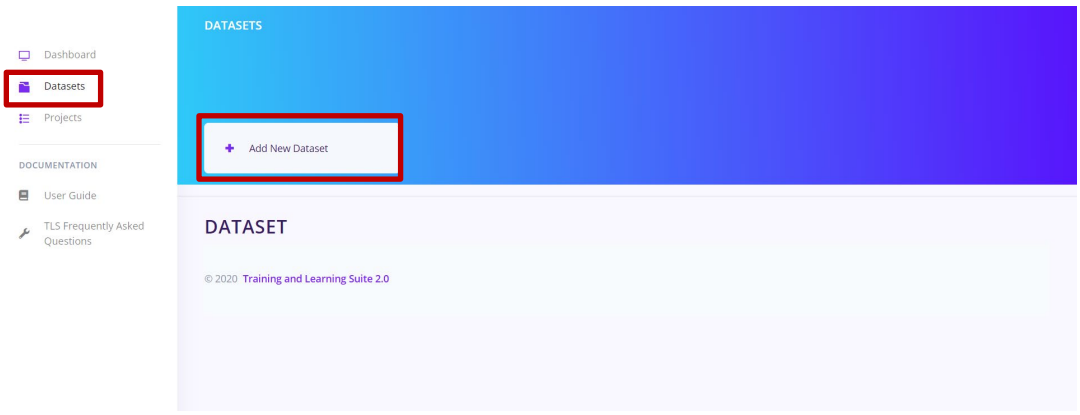
d. Box '**4**' shows where user can download the user guide for TLS2.0.

e. Box '**5**' is where users can read the TLS Frequently Asked Questions.

4.2 Steps to Add Dataset, Upload Images, and labelling

Users can add their dataset by clicking on the Dataset tab.

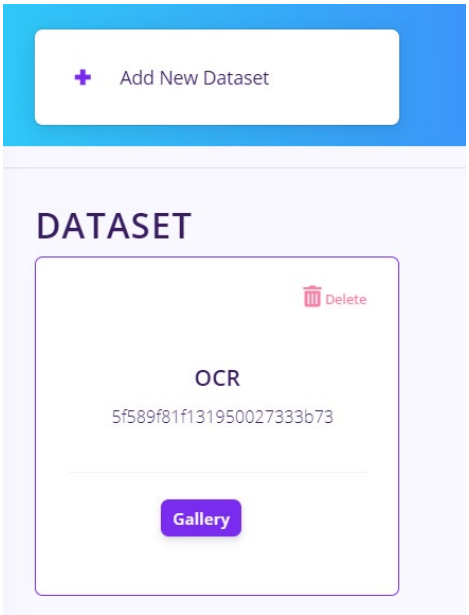
Figure 23. Dataset Tab



4.2.1 Steps to Add Dataset

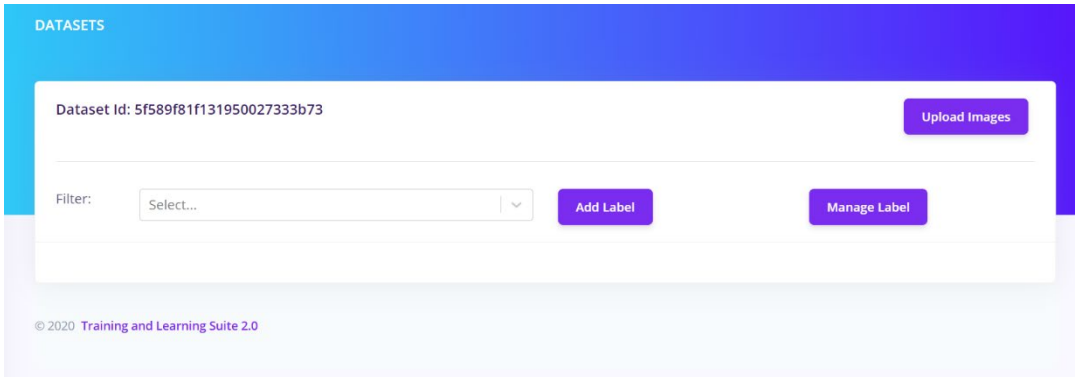
1. From TLS2.0 Dataset tab, click **+ Add New Dataset** button.
2. Users need to add Dataset Name and click **Create**. The dataset will be created as shown in the picture below . User needs to click on the **Gallery** button to upload the images. The dataset ID will appear below the dataset's name.

Figure 24. Add New Dataset



3. After clicking on **Gallery**, the image is ready to be uploaded into the dataset.

Figure 25. New Dataset is Created



4.2.2 Steps for Labeling on the Model Selection

For validation purpose, there are 4 categories of dataset provided:

a. Classification
b. Object Detection
c. Segmentation
d. Autoencoder

Selection of Whole Image and Box/Segmentation depends on model:

a. Classification or autoencoder used **Whole Image**

b. Object Detection or Segmentation used **Box/Segmentation**

Classification/Auto-encoder

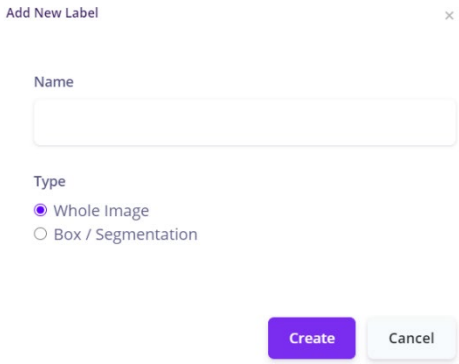
1. For Classification and Autoencoder, users need to create a label and choose **Whole Image**. It will not go into CVAT as the labeling is on the whole image.

2. Users need to click **Add Label** button.

Figure 26. Add Label Button

3. Add the label name and choose Whole Image.

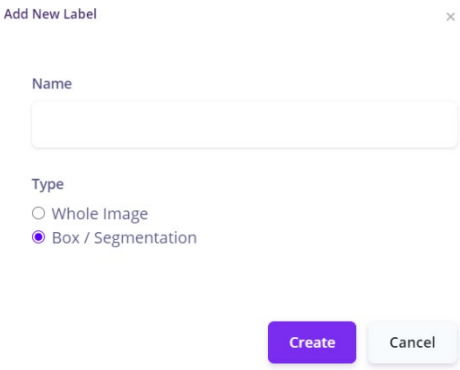
Figure 27. Label Type for Classification and Auto-encoder



Object Detection/Segmentation

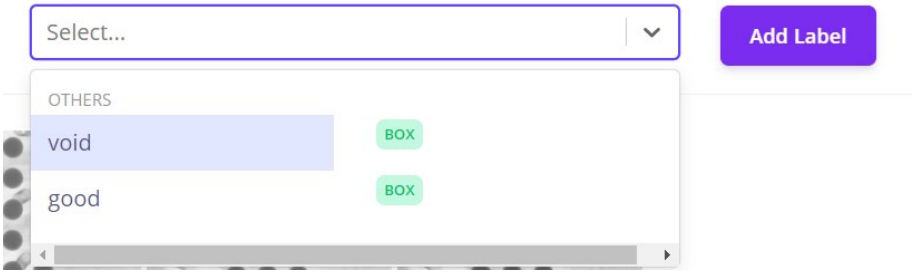
1. For object detection and segmentation, users need to create a label and choose **Box / Segmentation**. It will go into CVAT as users need to do annotation on the images.

Figure 28. Label Type for Object Detection and Segmentation



2. Example of label added for Object detection as shown in the figure below.

Figure 29. Example of Labels for Object Detection

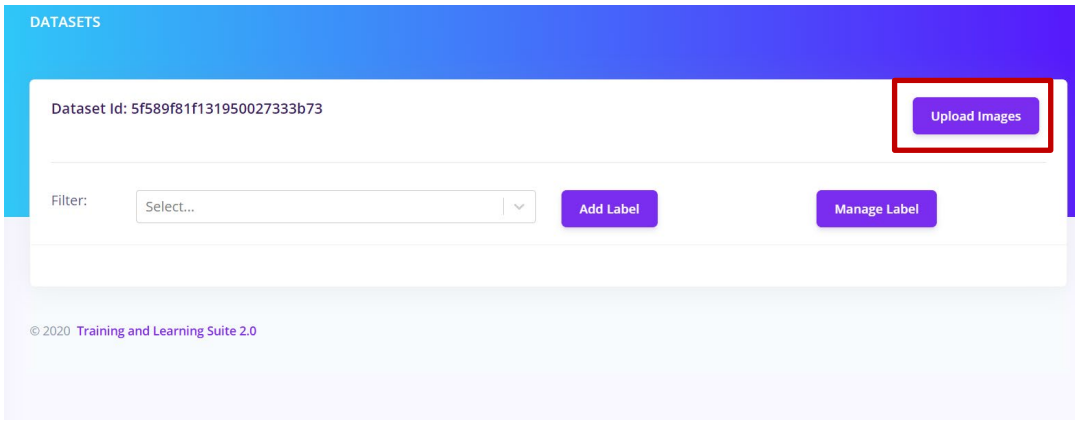


4.2.3 Steps to Upload Images into Dataset

After creating dataset and labels, users need to upload their images into dataset.

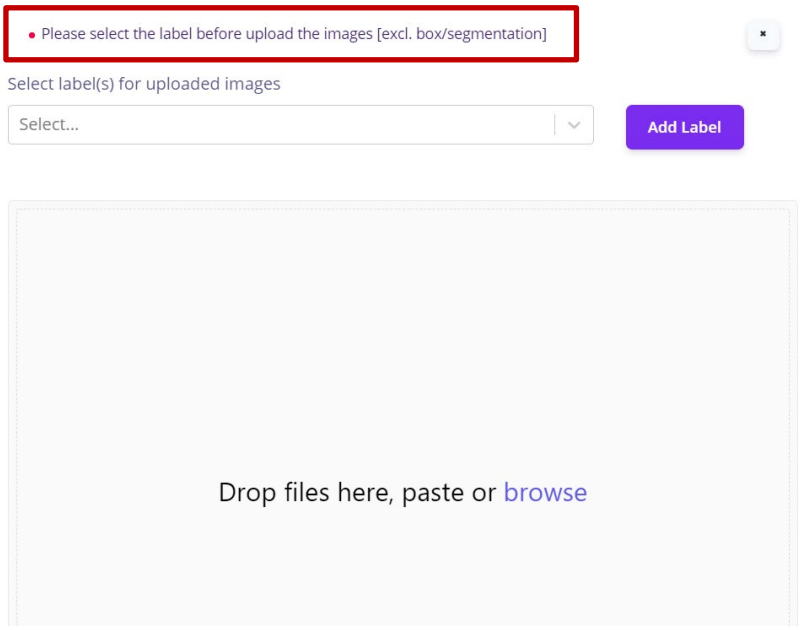
1. User needs to click on the **Upload Images** button.

Figure 30. Upload Images Button



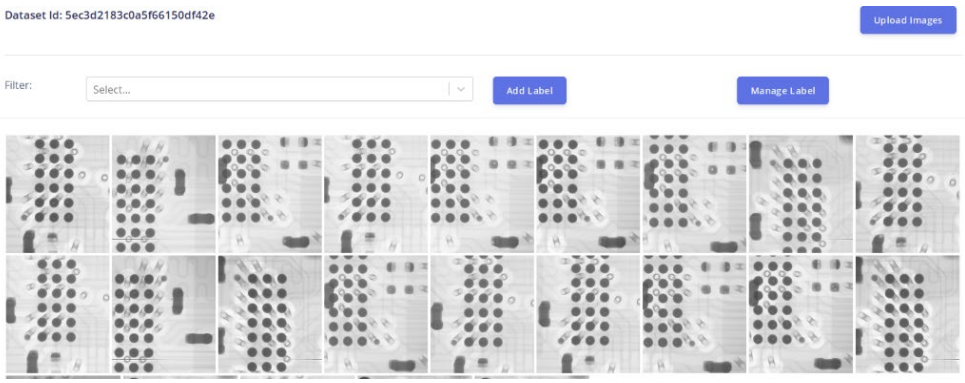
2. Drag and Drop to from Upload Image dialog.
3. For **whole image labeling**, user needs to select the label first before uploading the images.

Figure 31. Drag and Drop Images



4. Click upload and review images dataset is added.

Figure 32. Review Dataset

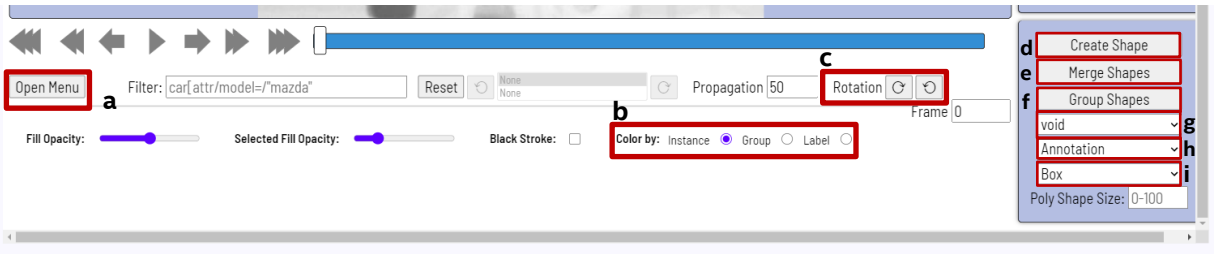


Note: ***It is recommended to upload maximum of 1000 images with 500x400pixels at one time.*** Users may repeat multiple times for large numbers of images. The TLS2.0 stores the image dataset based on the system disk-space. Please check Dashboard – Disk Usage for the information.

4.3 Steps to do Annotation on Dataset by CVAT

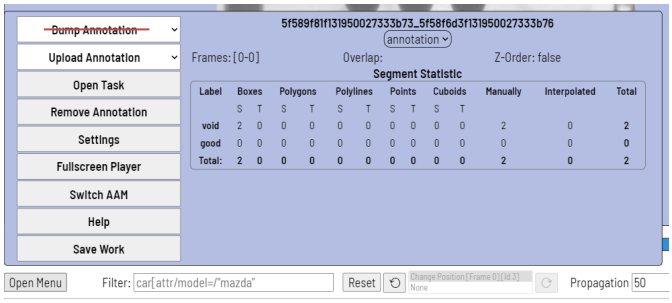
After the dataset is uploaded, the annotation can be manually annotated. The CVAT is only available for Object Detection and Segmentation models. The highlighted boxes below are the CVAT features function that applicable on TLS2.0.

Figure 33. CVAT Features Available on TLS2.0



a. Box '**a**' shows the Open Menu button. There have several functions on it. Note that the **Dump Annotation is currently unavailable to use on TLS2.0**.

Figure 34. Open Menu Button



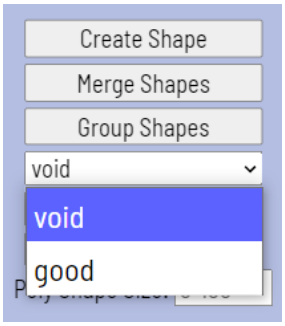
b. Box '**b**' shows the annotation color that can be chosen for either Instance, Group or Label.

c. Box '**c**' is for rotation. Users can rotate the images by using this feature.

d. Box '**d**', '**e**', and '**f**' are the features for users to create, merge and group the shapes when doing annotation on the dataset.

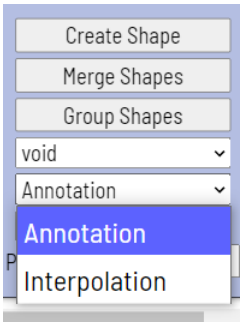
e. Box '**g**' shows the labels that users created for annotation on dataset.

Figure 35. Labels Selection



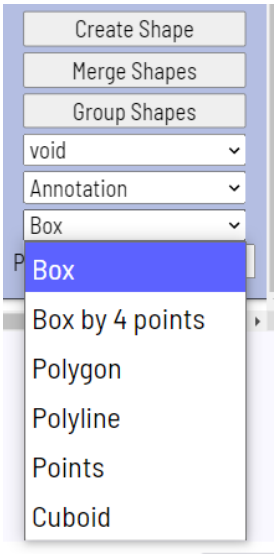
f. Box '**h**' shows the type of geometric bounding box that users can choose from either Annotation or Interpolation.

Figure 36. Type of Geometric Bounding box



g. Box '**i**' shows that there are five shapes which users can use to annotate images with Box, Box by 4 points, Polygon, Polyline, Points, and Cuboid.

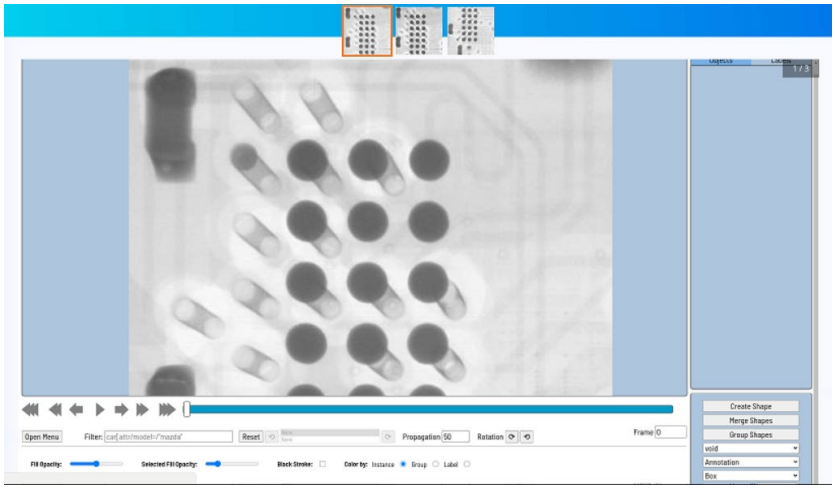
Figure 37. Type of Shapes to do Annotation



Follow the steps below to annotate on the image dataset.

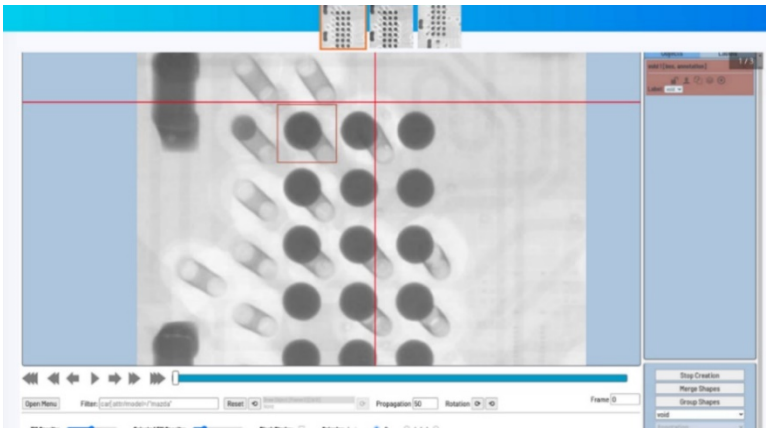
1. Click on uploaded image of dataset. The CVAT UI will appear after clicking on the dataset.

Figure 38. CVAT Annotation UI



2. User can press "**n**" on keyboard or click on **Create Shape** button for annotation. The object/label box on the right side will be appeared and user can choose the labels they have created before.

Figure 39. Shape Creation for Annotation

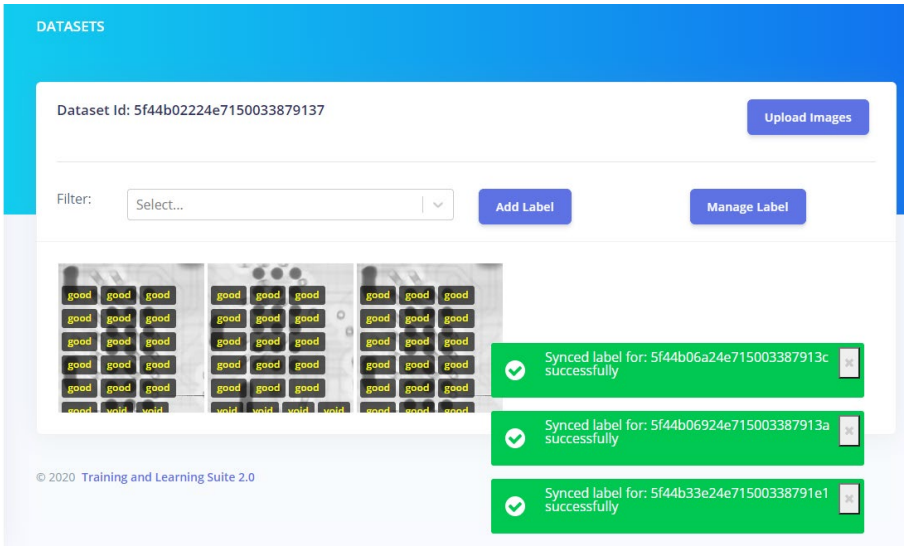


3. User needs to save all the changes after doing the annotation by clicking **Open Menu** button and choose **Save Work**.

Figure 40. Open Menu Button

4. The green alert pops up once annotation is completed successfully.

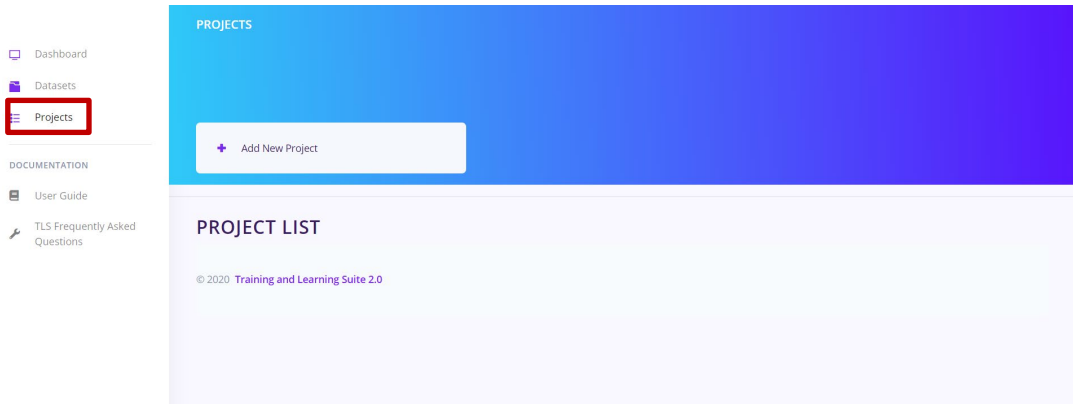
Figure 41. Alert Notification



4.4 Steps to Add New Project, Select Target Model, and Set Training Parameters

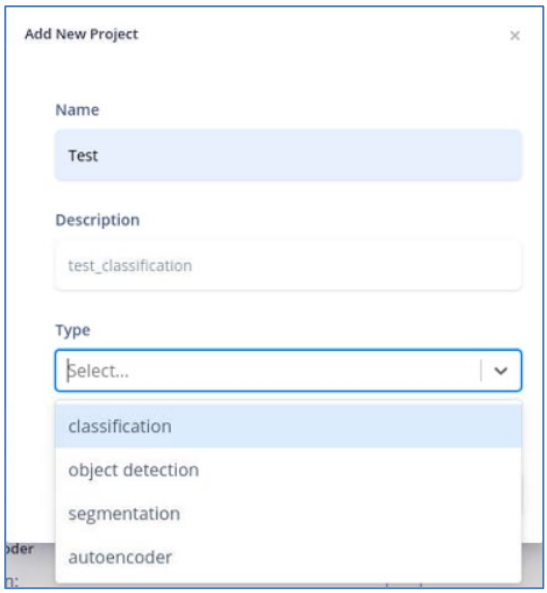
To start to do model training, user needs to go to the Project tab.

Figure 42. Project Tab



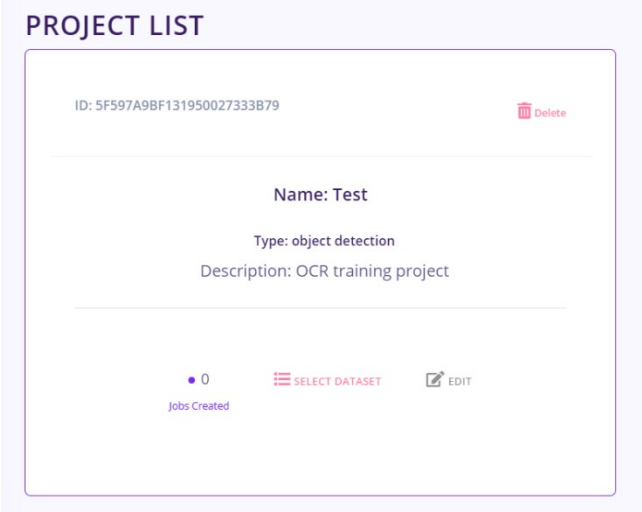
1. From Projects Dashboard, create a new training project by clicking the **+Add New Project** button. Enter the name, description, and choose the training type. The project will be created as shown below.

Figure 43. New Project Creation



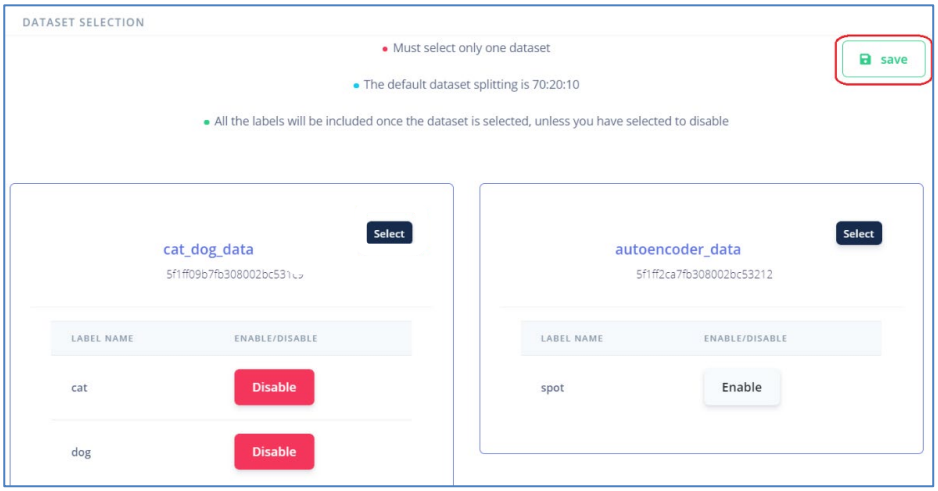
2. Once project is created, click on **SELECT DATASET**.

Figure 44. New Project Created



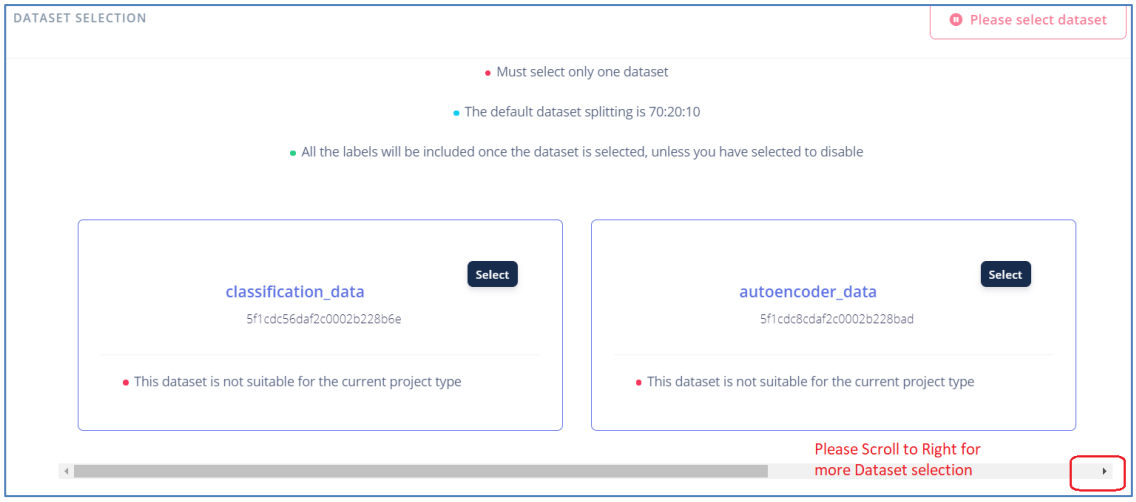
3. Select the suitable dataset from the dashboard. Click **"Select"** on your dataset then save your changes.

Figure 45. Dataset Selection



4. In case users can't find dataset that has been created, scroll to the right to see all the available dataset.

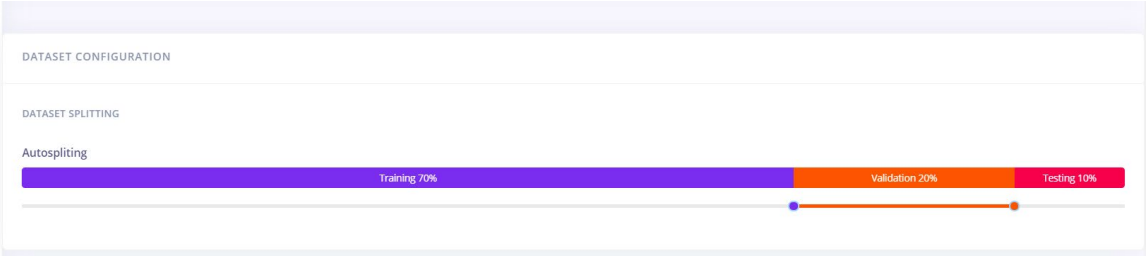
Figure 46. Available Dataset Selection



Note: *Scroll to the right to see all the available dataset.*

5. After selecting the dataset, scroll down and there will be a dataset configuration. Users can split their dataset by 3 categories, which is for training, validation, and testing. Users can use adjustment bar to split the dataset into categories.

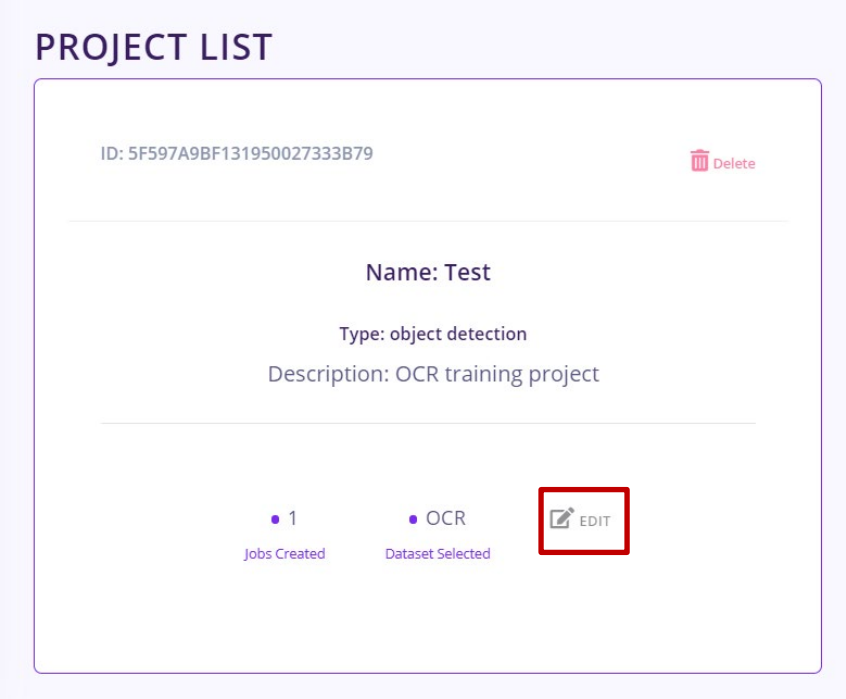
Figure 47. Dataset Splitting



Note: Once the dataset is selected and split, it is unchangeable for the specific project. This is to ease the users to do the comparison between different parameter configuration and model selection with the same dataset.

6. Then, save the selection dataset, Click on **Edit** button to start creating model training.

Figure 48. Click on Edit Button



7. **Augmentation**: Users need to click on Edit button to edit the parameters for the model training. Choose the preferences by clicking the box for data augmentation. ***Data augmentation is only available for classification***. Otherwise, user can skip the augmentation part.

Figure 49. Augmentation on Object Detection

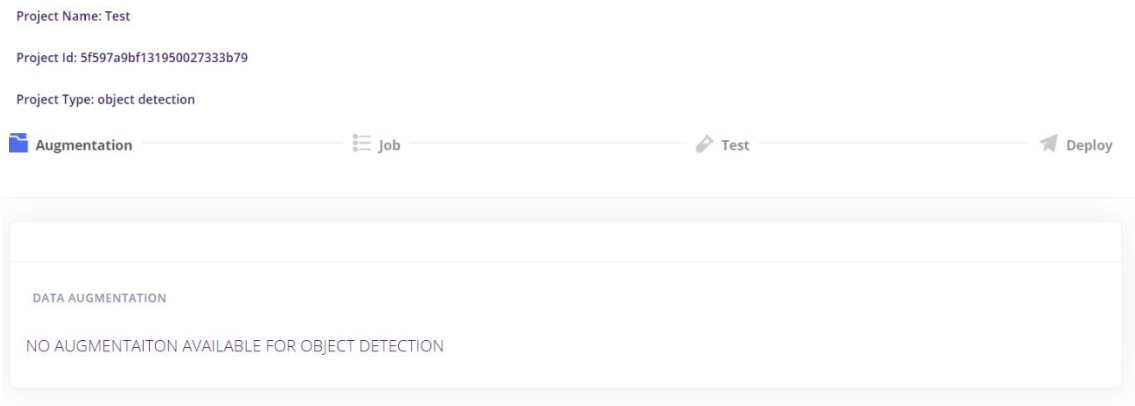
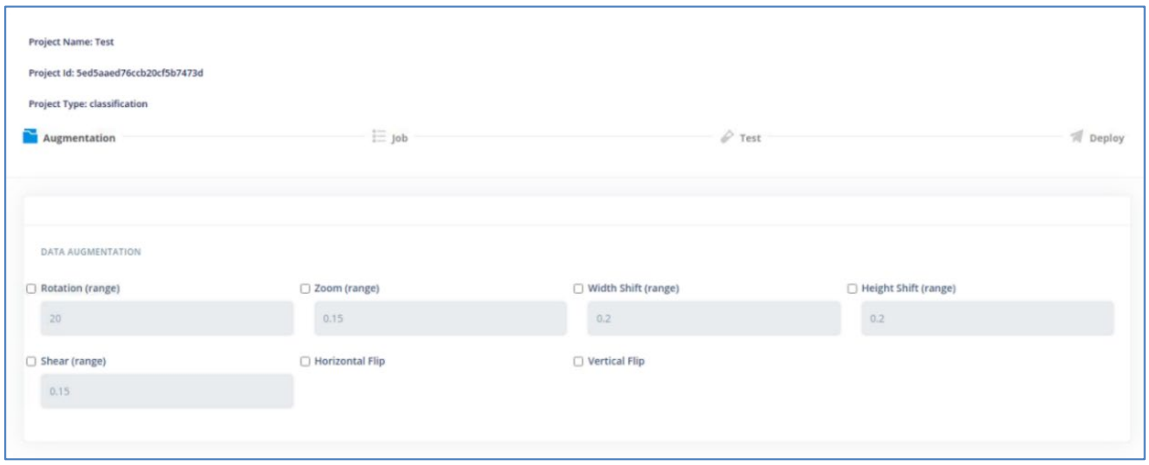
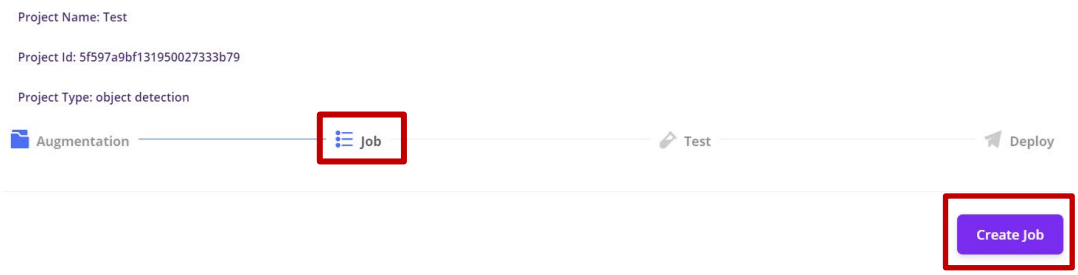


Figure 50. Augmentation Parameter on Classification



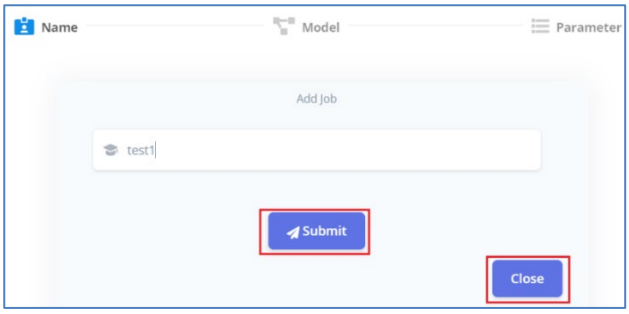
8. Next is to create a job. Click on **Create Job** button and enter your data. Users need to enter the name of the job, select Pertained model, and enter the parameter of the model.

Figure 51. Job Creation



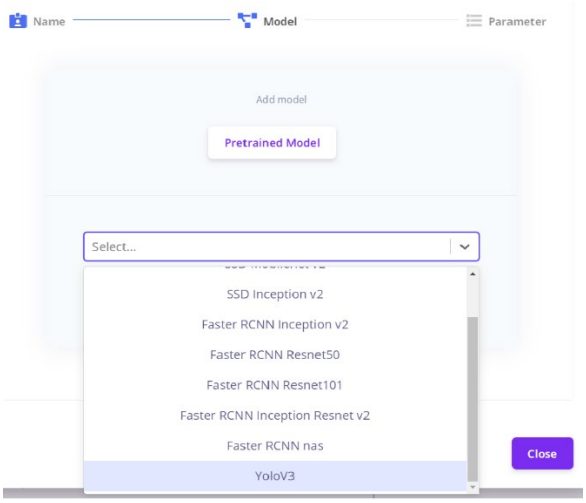
9. Click the ***Submit*** button for every data that has been entered. User can create multiple jobs in one project.

Figure 52. Submit Button



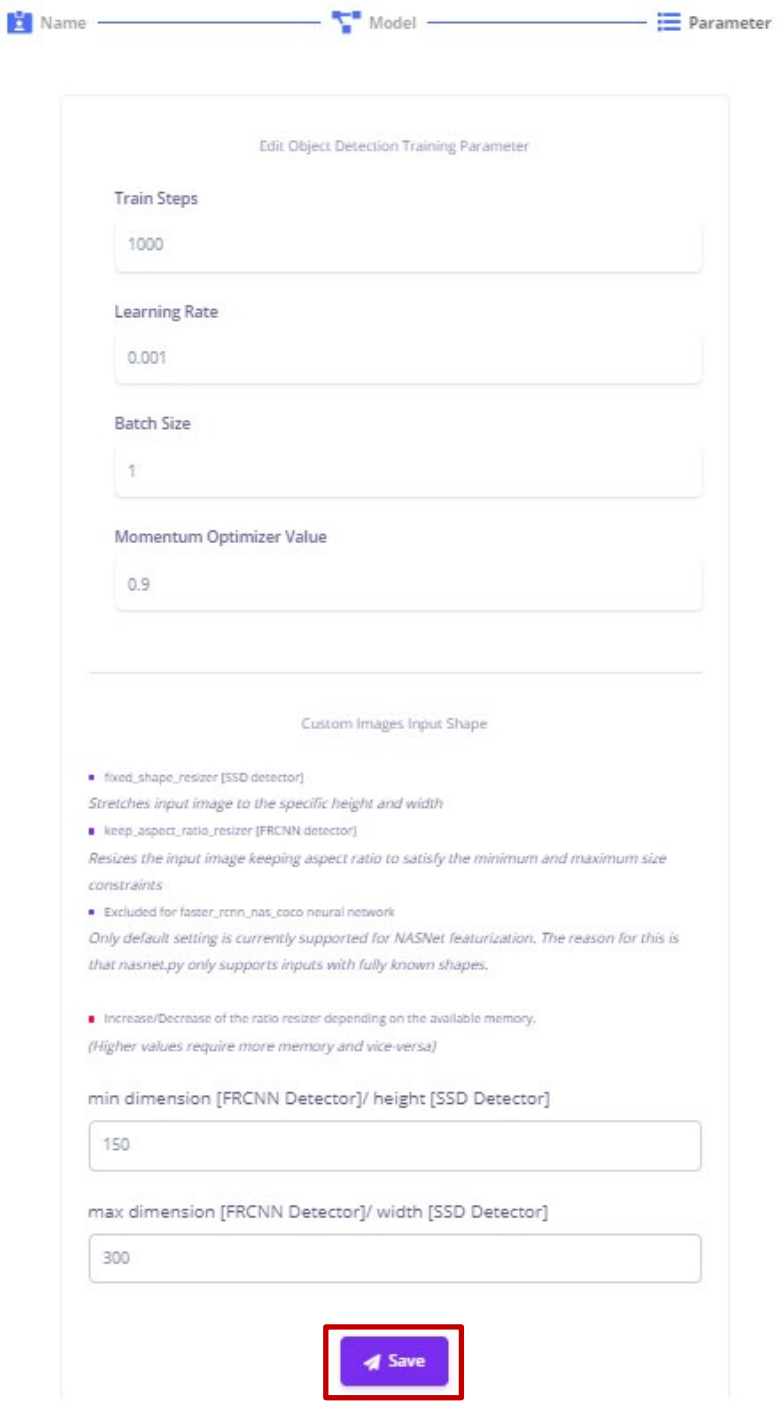
10. Click *"**Model**"* Tab to select deep learning models. This show example of Object Detection models list to be selected.

Figure 53. Model Selection for Object Detection



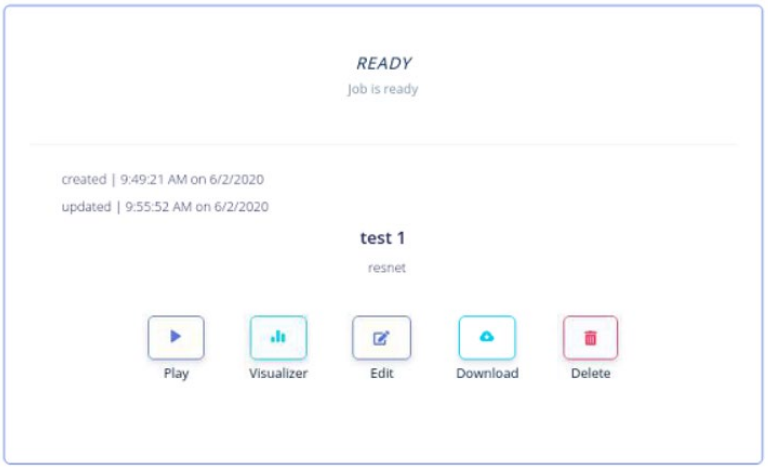
11. More Tuning Parameters settings to be selected. Click on **"Save"** button to send all the data to the task job. Note that each model has different parameter tuning.

Figure 54. Tuning Parameter for Object Detection



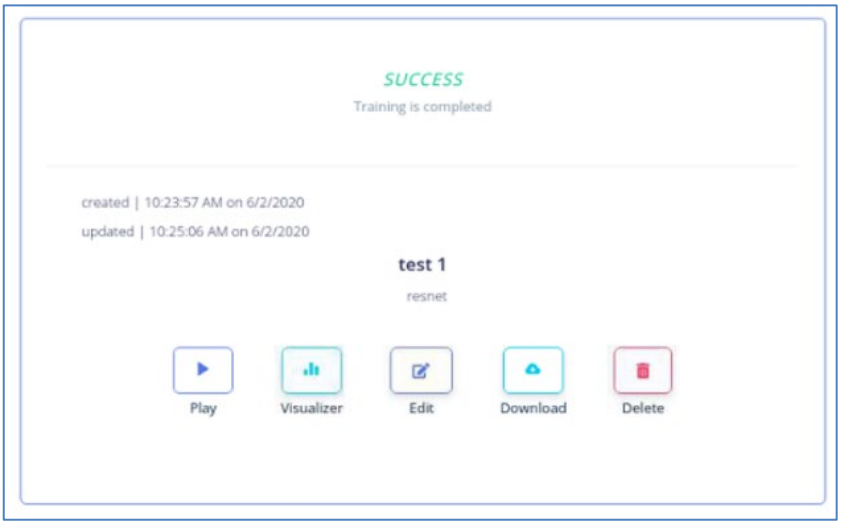
12. The job is in Ready to train mode after parameter is submitted. User now can start the training by clicking ***"Play"*** and wait for the training to finish.

Figure 55. Job Ready



13. Finally, training is done as shown below. User may download the trained model for deployment. User may also test the accuracy with OpenVINO™ through TLS2.0 Inference UI.

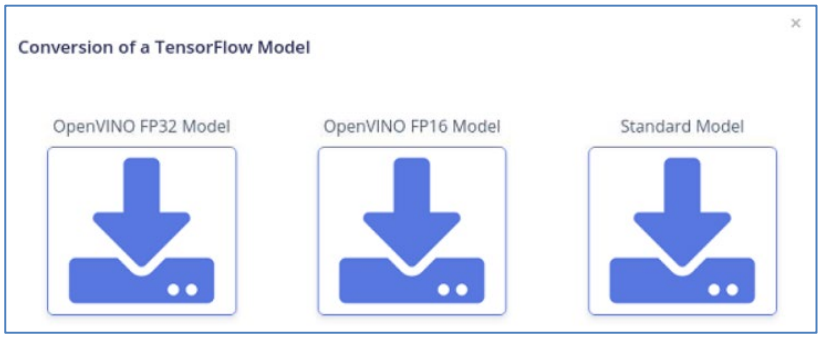
Figure 56. Successful Training



14. User can see the Confusion Matrix or Metrics Visualizer by clicking the Visualizer button on the project. User can also download the post training model in FP32, FP16, or standard model.

Note: Downloaded file will be in this format **<job-id_job-title_date_size-format>**

Figure 57. Download Model



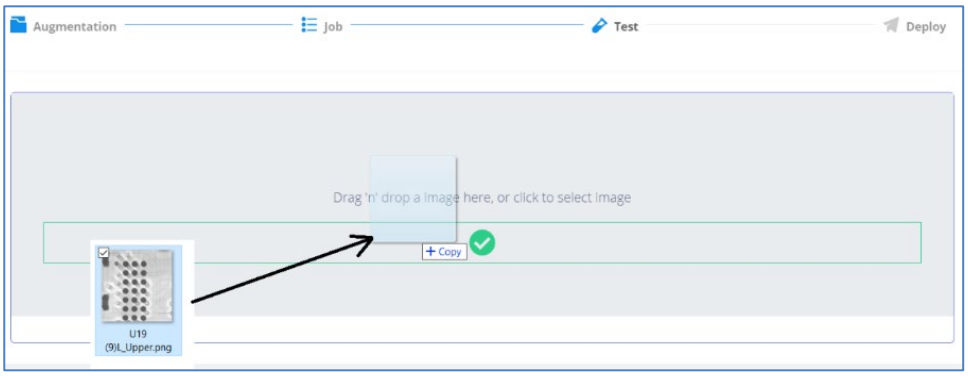
4.5 Steps to do the Inferencing on Image

After training the model, user can do a local inferencing by

1. Click the **Test** button on the UI dashboard.

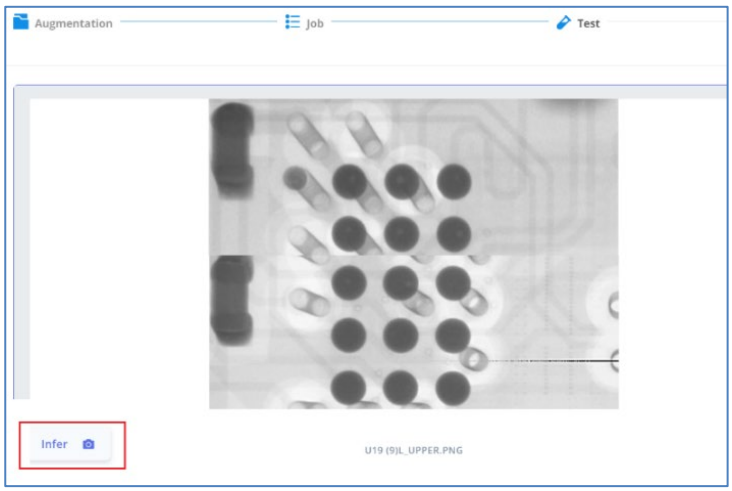
1. User needs to drop, drag, or click on the box to select image.

Figure 58. Image Selection



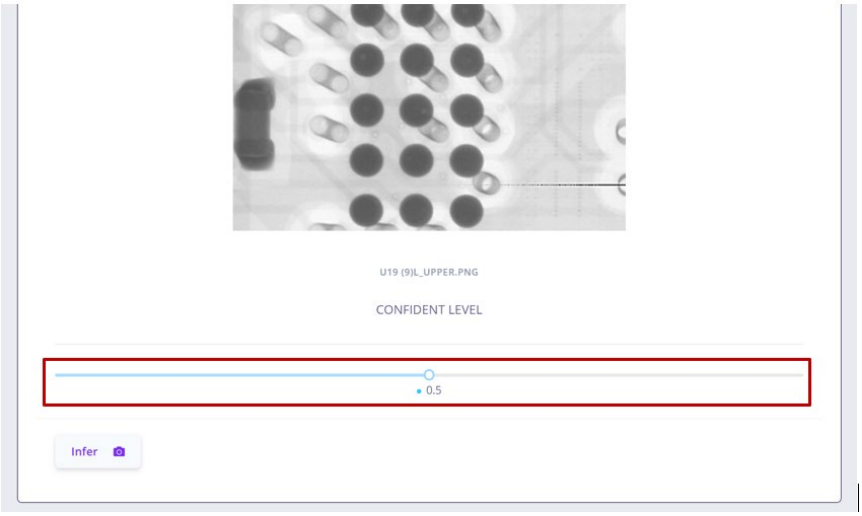
2. Once the image is uploaded, user can click on the infer button. The output results of inferencing will come out later.

Figure 59. Image Added



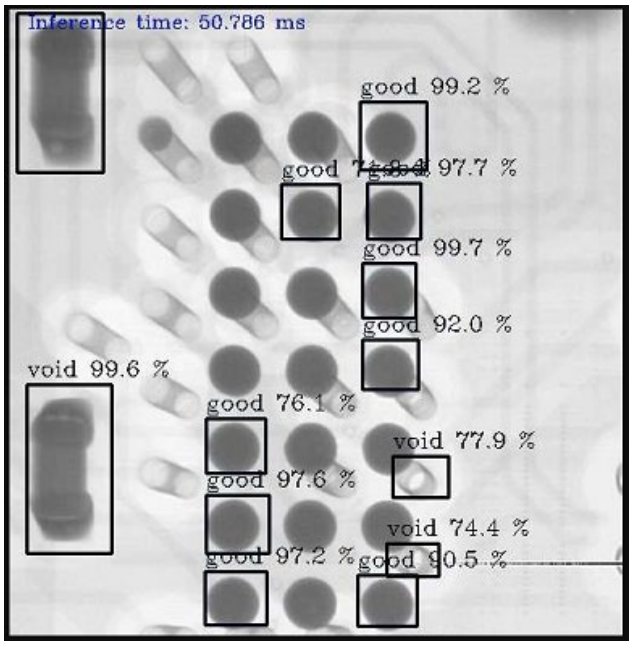
3. User can adjust the confident level before starting inferencing.

Figure 60. Threshold Limit Adjustment Bar



4. Below is the example of local inferencing result

Figure 61. Local Inferencing Result



4.6 Steps to Perform Remote Inferencing on Image

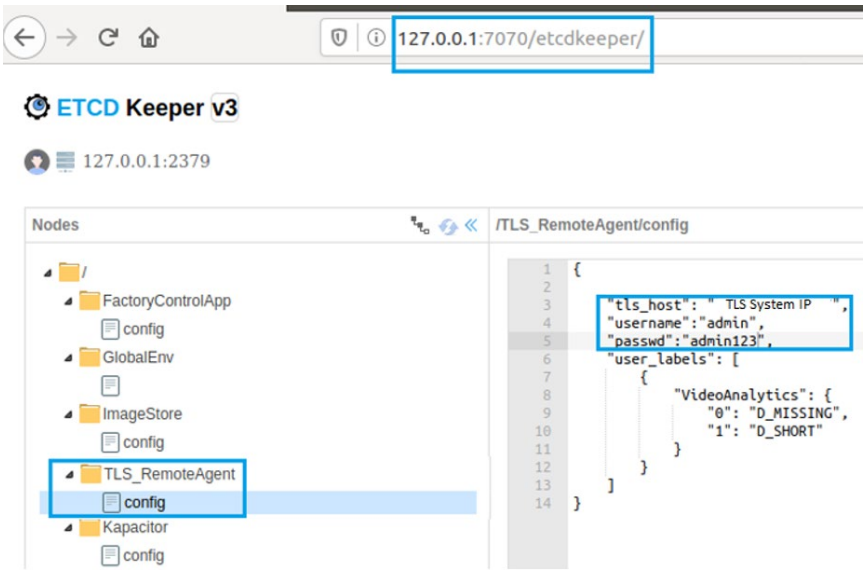
User can do the inferencing by using the remote inferencing with EIS (Edge Insight Software). Before using the EIS installed system remote for inferencing at the edge device, it is assumed that the EIS has been setup according to the EIS documentation and successfully running inferencing on the edge device.

In addition, it is also required the ***ia_tls_remoteagent*** service being setup and running in the EIS, as described in the TLSRemoteAgent readme.md file in EIS repository.

In order to demonstrate EIS running inferencing at the edge and connecting with TLS system, a default demo sample app in EIS has been included for this purpose.

User needs to change the IP address to the targeted host machine, username, and password from RabbitMQ* account that been set up during TLS 2.0 installation in ECTD Keeper page from **EIS installed system** (http://127.0.0.1:7070/etcdkeeper).

Figure 62. ETCD Web UI



1. In TLS remote inference web UI page, click “Add New Agent” and filled up the following information to create a remote connection to each camera in EIS.

    a) Create a name for the camera connection.

    b) The UUID can be obtained from the EIS system display screen once EIS running with ***ia_tls_remoteagent*** service.

    c) Choose the trained model that is deployed into the EIS system.

Figure 63. Remote Agent Page

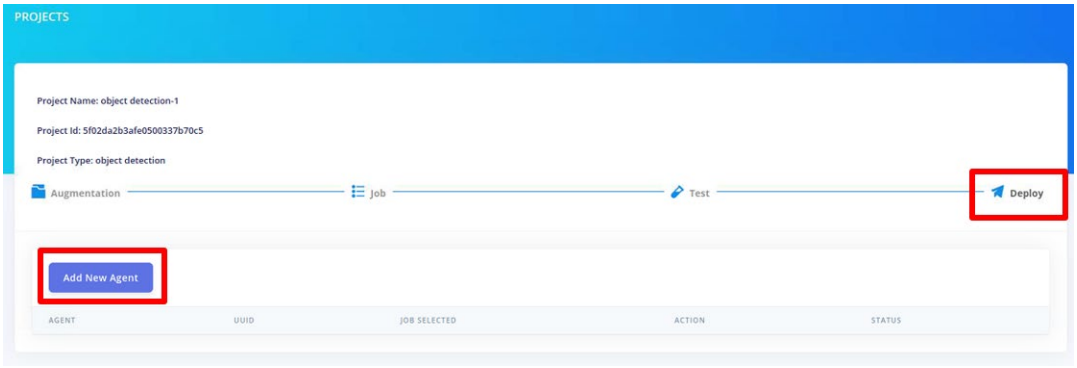
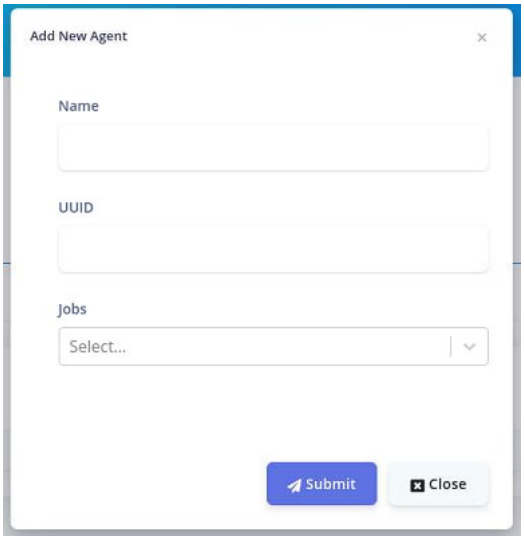
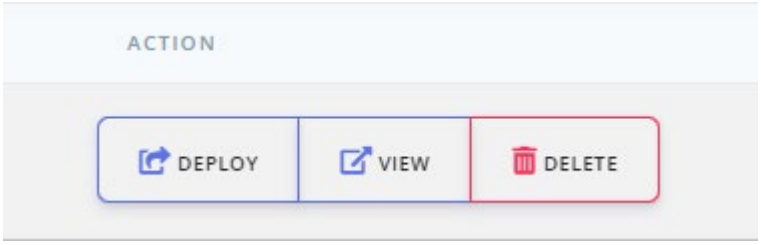


Figure 64. Add New Agent



2. Once filled up, click **"Submit"**, a new connection will be setup with EIS. Each connection represents one camera stream topic from EIS system.

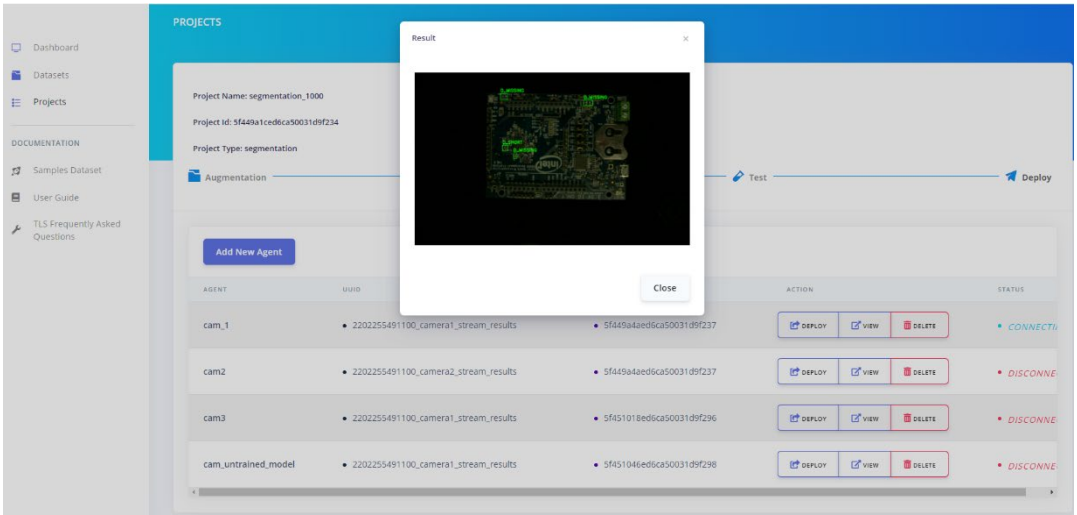
Figure 65. Action Button



3. There will be 3 action buttons. Click **"Deploy"** button to deploy the trained model into EIS edge device. The model is deployed into EIS's **/opt/intel/eis/model_repo** host system directory.

4. If the inferencing has started in EIS system, click the **"View"** button to connect the stream topic and initiate the video frame streaming from EIS system to the web UI.

Figure 66. Example of Inference Result from EIS



5. Click the **"Delete"** button to delete the remote agent if needed.

§

5.0 Frequently Asked Questions

5.1 Docker Documentation

User can use this Docker command to check their Docker status.

To check if the Docker is running, open up the terminal and run the following commands:

1. List containers in TLS2.0:

```
$ docker ps
```

```
tls_core
tls_apiui
tls_rabbitmq
tls_mongo
tls_proxy
cvat_proxy
cvat_ui
cvat
cvat_db
cvat_redis
```

2. To fetch the logs of a container for debug purpose

```
$ docker logs [CONTAINER]
```

example: **$ docker logs tls_core**

3. To build, (re)create, start, and attach to containers for a service.

```
$ docker-compose up -d
```

4. To stop running containers without removing them. They can be started again with docker-compose start.

```
$ docker-compose stop
```

5. To stop containers and removes containers, networks, volumes, and images created by up.

```
$ docker-compose down
```

6. To remove all unused images, not just dangling ones:

```
$ docker system prune -a
```

5.2 Information on Frequently Asked Questions

- **How to check if TLS2.0 is setup successfully?**

  Please refer to this User Guide, Section 3.2, Figure 14, screen capture of the TLS Setup Completion.

- **What does the message "WorkerLostError: Worker Exited Prematurely: signal 11 (SIGSEGV)" mean?**

  Hardware RAM is insufficient to support the training process. Increase the available RAM of the hardware by referring to the recommended training batch size for each model on the 32GB and 64GB system table in Subchapter 2.1 on this User Guide.

- **What does the message "Allocation of 3425697792 exceeds of 10% of system memory" mean?**

  Hardware RAM is insufficient to support the training process. Increase the available RAM of the hardware by referring to the recommended training batch size for each model on the 32GB and 64GB system table in Subchapter 2.1 on this User Guide.

- **What does the message "ValueError: zero-size array to reduction operation maximum, which has no identity" mean?**

  Dataset is not selected. Please select training dataset before starting the training.

- **What does the message "TypeError: Signature mismatch. Keys must be dtype dtype: dtype: 'string'." mean?**

  Dataset is not selected. Please select training dataset before starting the training.

- **What does the message "ValueError: With n_samples=1, test_size=0.1 and train_size=None, the resulting train set will be empty. Adjust any of the aforementioned parameters." mean?**

  The input dataset is not enough split into train dataset, test dataset and validate dataset. Please increase dataset.

- **Why does the label not appear after the annotation is done?**

  Please click on "Open Menu" and "Save Work" button after annotated the dataset. Please refer to Step 6 in Subchapter 4.2 on this guide.

- **Can the user customize the neural network?**

  Currently TLS2.0 only supports pretrained models that are also supported in OpenVINO™ toolkit.

- **How do I cross check if all TLS2.0 Dockers are up and running?**

  i. `$ docker ps`

  ii. *Please refer to the Use the Docker command line tab to check if any docker names are missing*

- **Why does the uploaded image not have a label?**

  Please use the “Assign Label” button to assign them. The correct sequence to assign labels is shown here:

  i. *Add label from “Add Label” button*

  ii. *Click “Upload Image” button. At pop-up dialog, select labels from drop-down list*

  iii. *Select images from folder, then drag-drop them*

  iv. *Click “Upload # Files” button to upload images into TLS2.0*

- **Training Error?**

  i. `$ docker logs -f tls_core`

ii. Please refer to FREQUENTLY ASKED QUESTION for the solution.

- **Why can't the user create a new remote agent after creating one on the deploy page?**

  Refresh the page and try again.

5.3 Summary

Most features and design flows of TLS2.0 are based on TLS1.0, with new categories enabled, such as Auto-encoder, Segmentation, and new models like NasNet, DenseNet, and YOLOv3.

Note: Please take note of the danger/warning alerts appearing while using TLS2.0.

§