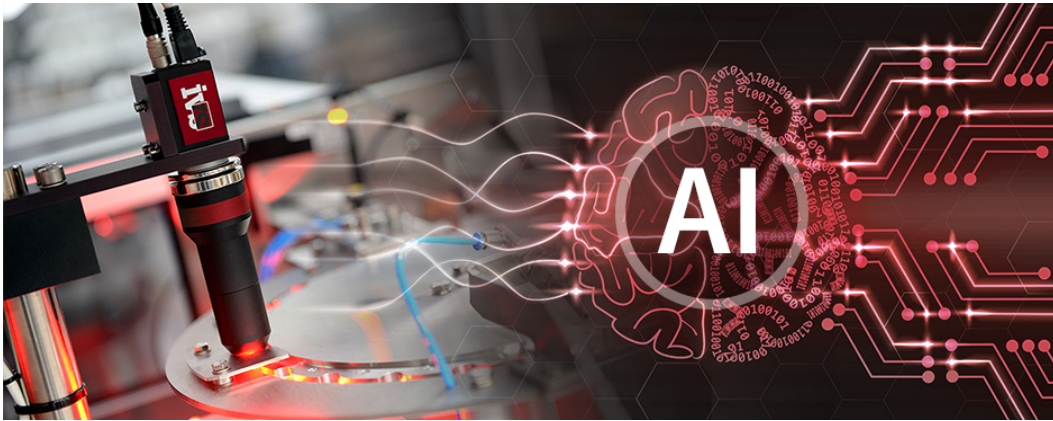

INSTALLATION / USER MANUAL



AUTHOR: SUNDY

DEYE: KEEP AN EYE ON DEFECTS INSPECTION

Abstract

Defect Eye (DEye) is a deep learning-based software for manufacturing surface defect inspection. It provides the basic function modules to facilitate the development of different defect inspection applications. The applications cover the full range of manufacturing environment, including incoming process tool qualification, wafer qualification, glass surface qualification, reticle qualification, research and development. Also, It can be used for medical image inspection, including Lung PET/CT, breast MRI, CT Colonography, Digital Chest X-ray images. This software library contains the basic function modules about data processing, model training and model inference. It is developed to reduce the burden of programmers who worked in this field. Based on this software, developers can design the added functions according to their requirements. This document describes how to build DEye software and how to use it.

1 Introduction

The DEye [4] software library consists of 7 modules with some inter-dependence. The function of each module is shown as below:

- **GUI**: a simple user interface for model training and inference (programmed with C#).
- **trainer**: it contains data processing and model training based on Tensorflow framework (programmed with C++).
- **classifier**: it contains model inference when feed the input images (programmed with C++).
- **dataGen**: a module to generate training data under the specified size (programmed with C++).
- **example**: some function test cases (programmed with C++).
- **SVM**: a simple demo to show how to use SVM classifier to classify images (programmed with C++).
- **txt2xml**: a tool to convert txt format to xml format (programmed with C++).

2 Development Environment

The Microsoft Visual Studio 2017 (VS2017) is used to develop the DEye software under Window (English version, 64 bits) operation system, the version of VS2017 is show in Figure 2.

Besides, the configuration of each module is shown in Figure 3, Figure 4. All of these modules should be configured in a right way.

3 3rd Party Library

DEye is depended on several third party libraries, i.e., Tensorflow [1], OpenCV [2], Glog [3], which all of them are open source. In this project, the C++ version of Tensorflow 1.4, OpenCV 3.4, and Glog are used to support DEye library development. The newest 3rd-party libraries should be built by developers who want to update the version. Fortunately, the default version is provided with the following links:

- BaiduYun Link: <https://pan.baidu.com/s/1o9tv1n8>, password: ekec
- Google Link: <https://drive.google.com/open?id=1kANDNErMNLU9wNR3rKhUTz-ltwPNPUv>

All the libraries should be downloaded into the computer, then configuring the environment with the following steps:

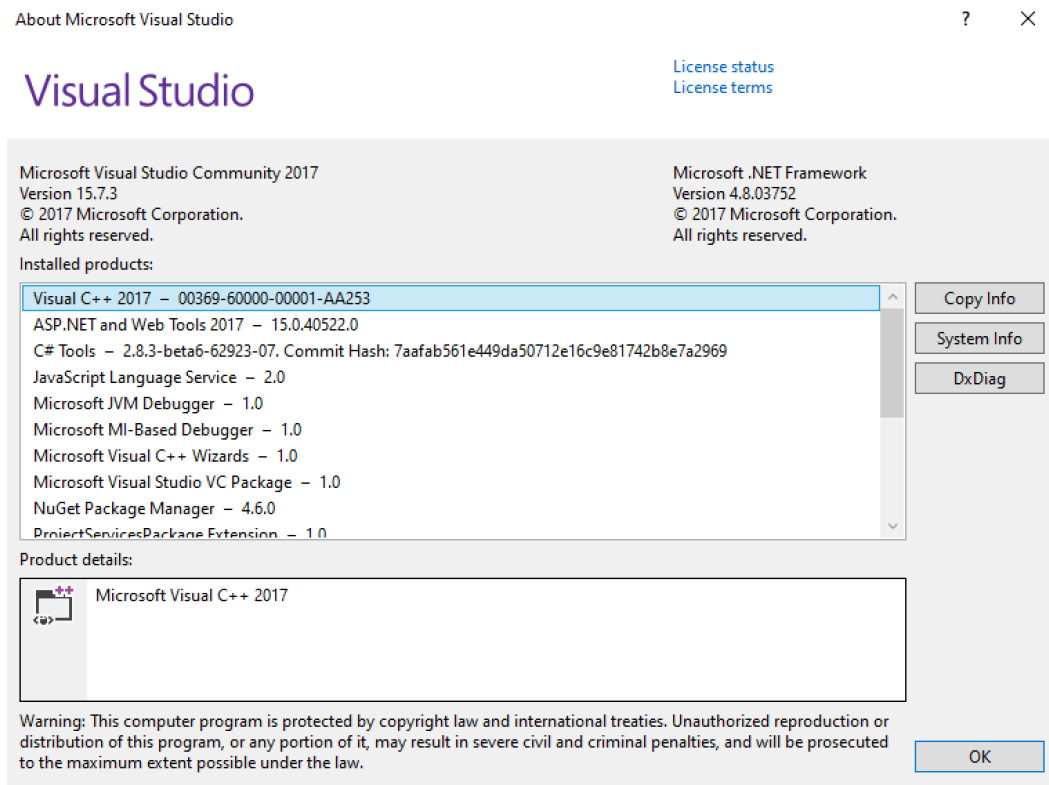


Figure 1: Visual Studio 2017 version

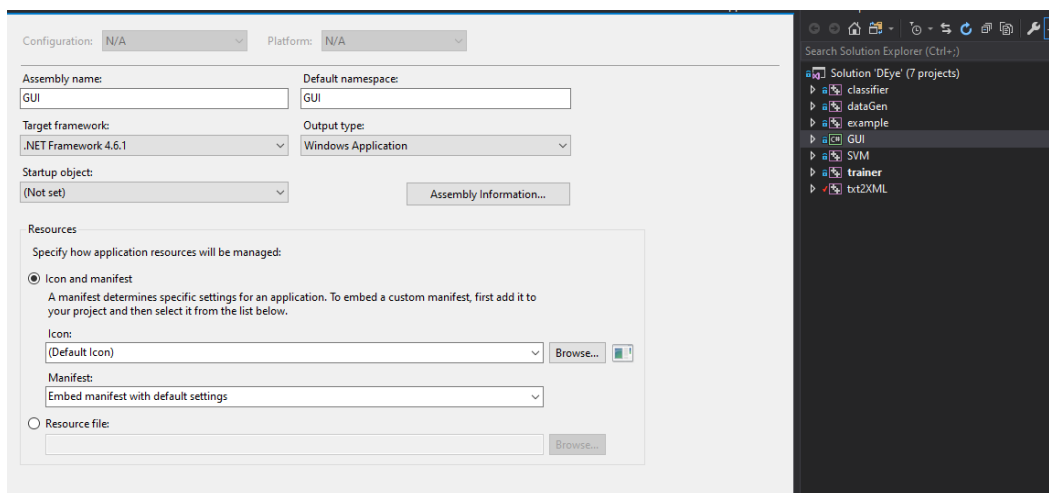
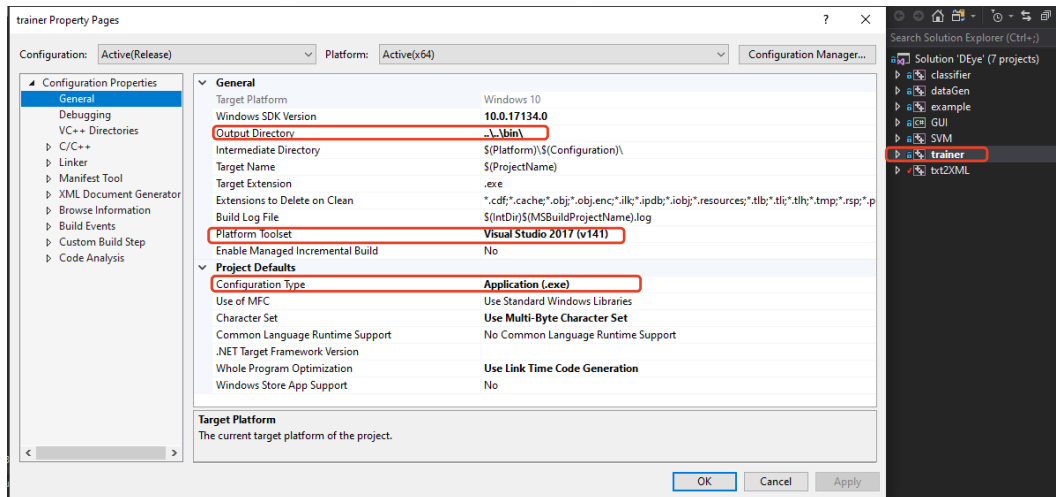
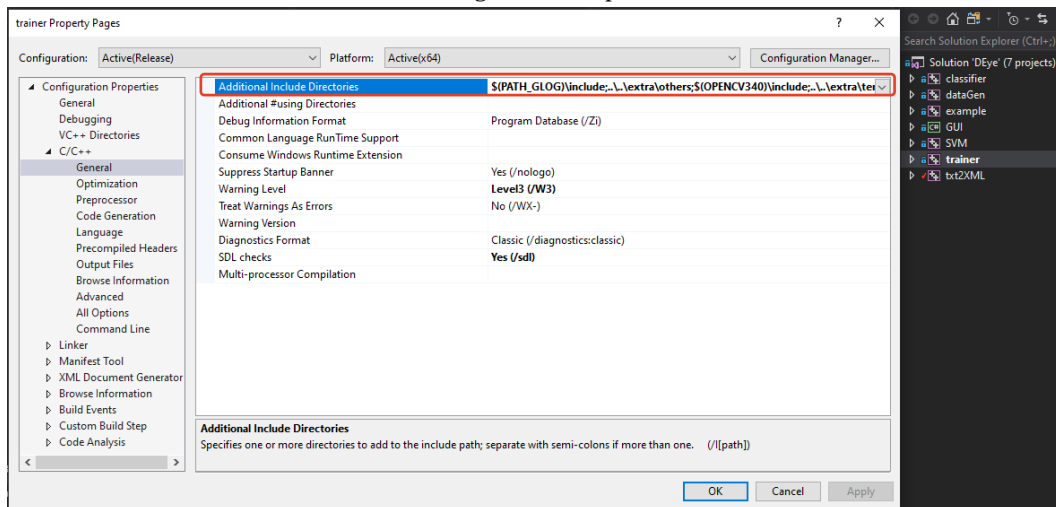


Figure 2: Configuration of GUI module.



(a) Configuration Properties.



(b) Configuration of Include Directories.

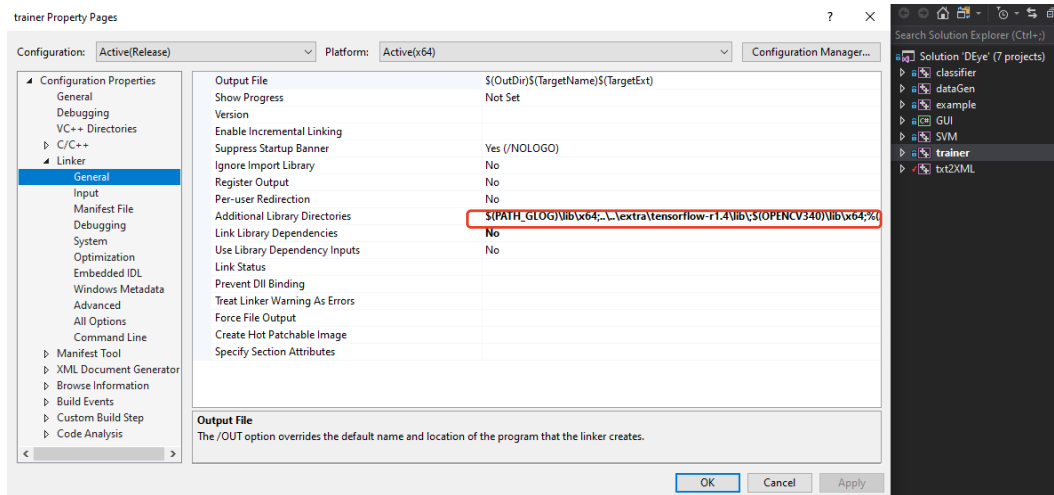
Figure 3: General Configuration.

- Step 1: put the libprotobuf.lib and tensorflow.lib into DEye->extra->tensorflow-r1.4->lib
- Step 2: put all the needed dll files into DEye->bin
- Step 3: configure the system parameters. Open the environment setting windows: Advanced system settings -> Environment variables -> System variables. The example is shown in Figure 5. **Note that the value of variables should be changed to your own.**

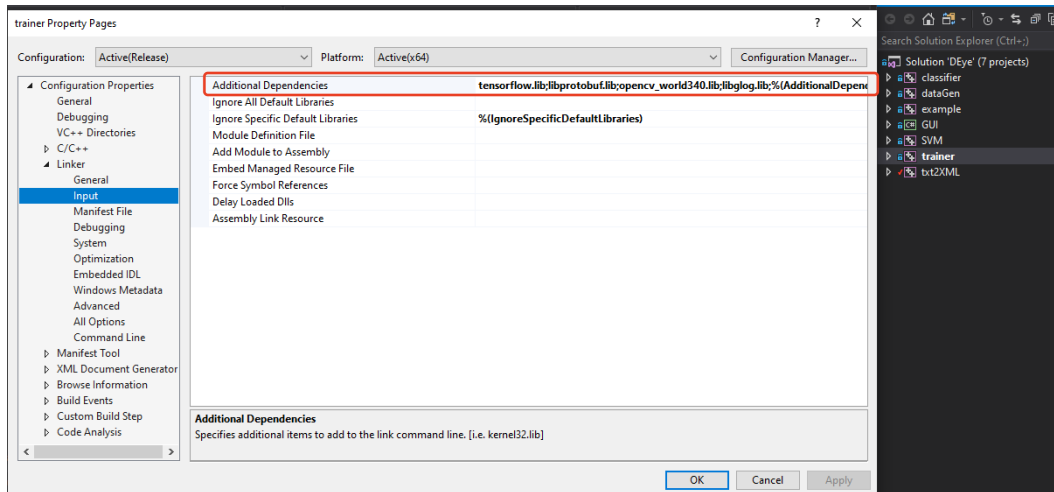
4 DEye Compilation

The 7 modules can be compiled when finished the above configurations. Please follow the the steps show in Figure 6.

If all the modules are build successfully, you will find the executable files under DEye->bin, which is shown in Figure 7.



(a) Configuration of Linker->General.



(b) Configuration of Linker->Input.

Figure 4: Configuration of Linker.

5 DEye Training

5.1 Use the GUI

Please follow the steps as shown in Figure 8.

If you want to change the the event of "Start Training" button, please modify the `bt_train_Click()` function as shown in Figure 9.

5.2 Use the command

The trainer module have 5 input arguments, please find the descriptions in following:

- `–method`: 1: input model is Tensorflow pb file, 2: input model is Tensorflow checkpoint file.
- `–graph`: the path of input model.
- `–config`: training configuration file.
- `–neg`: the path of negative samples (i.e.: defectiveness images)

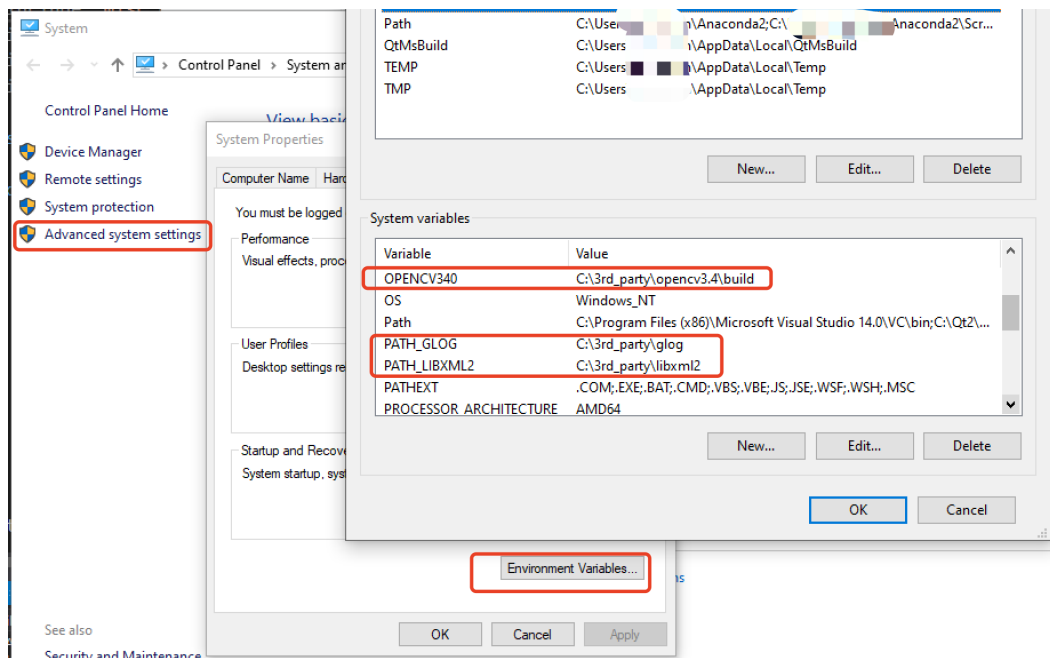


Figure 5: Environment variables configuration of OPENCV340, PATH_GLOG, and PATH_LIBXML2.

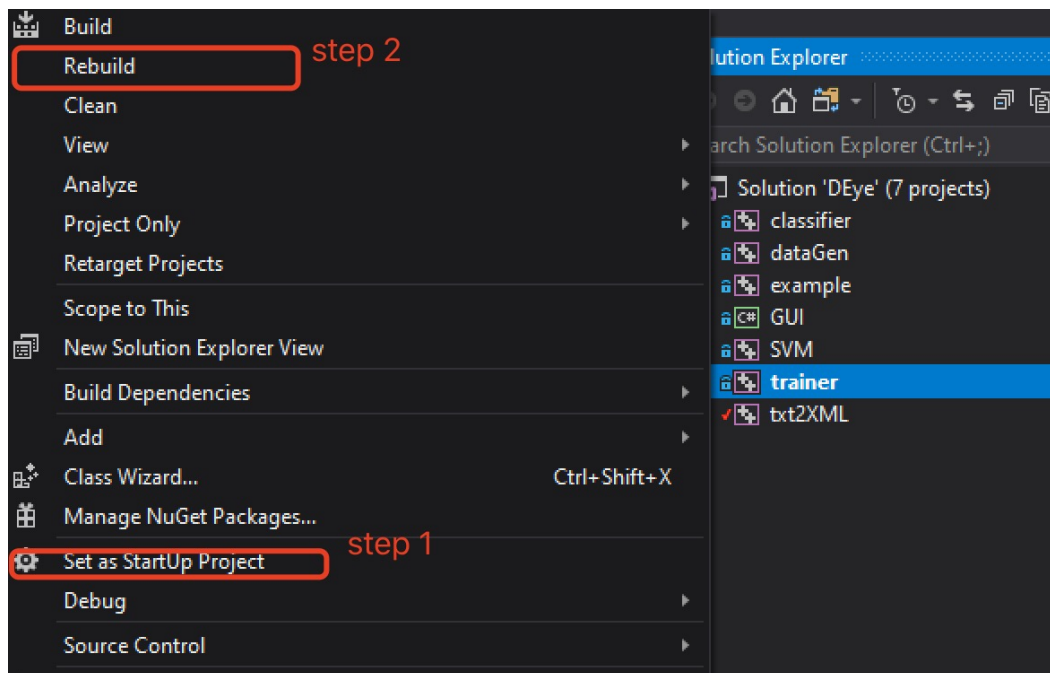


Figure 6: Build the module.

D:\workspace\DEye\bin					Search bin
Name	^	Date modified	Type	Size	
model		10/26/2020 11:26 PM	File folder		
1@trainer.bat		10/26/2020 11:31 PM	Windows Batch File	1 KB	
2@blobClassifier-ok.bat		10/26/2020 11:31 PM	Windows Batch File	1 KB	
3@blobClassifier-ng.bat		10/26/2020 11:31 PM	Windows Batch File	1 KB	
classifier.exe		10/30/2020 10:04 PM	Application	99 KB	
dataGen.exe		10/30/2020 10:06 PM	Application	69 KB	
GUI.exe		10/26/2020 11:31 PM	Application	13 KB	
SVM.exe		10/30/2020 10:16 PM	Application	47 KB	
tf_train.ini		10/26/2020 11:18 PM	Configuration sett...	1 KB	
trainer.exe		10/31/2020 12:45 AM	Application	147 KB	

Figure 7: All the executable files after compilation.

DEEP LEARNING FOR DEFECT INSPECTION

DEye Training

DEye Testing

step 1

step 2

step 3

Figure 8: Illustration of model training.

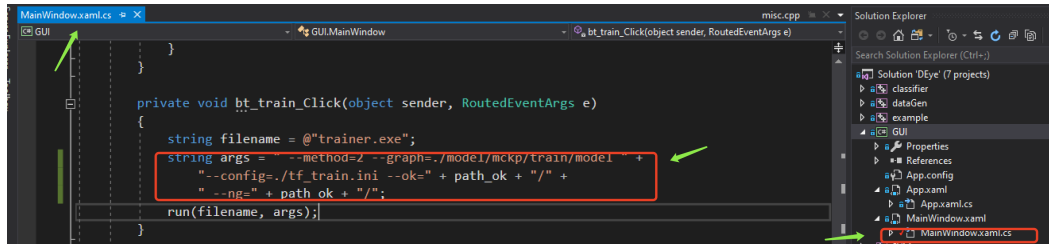


Figure 9: The backend of bt_train_Click().

- --pos: the path of positive samples(i.e.: defective images)

Example:

trainer.exe --method=2 --graph=../model/mckp/train/model --config=../tf_train.ini --neg=../data/train/OK/ --pos=../data/train/NG/

6 DEye Inference

6.1 Use the GUI

Please follow the steps as shown in Figure 10.

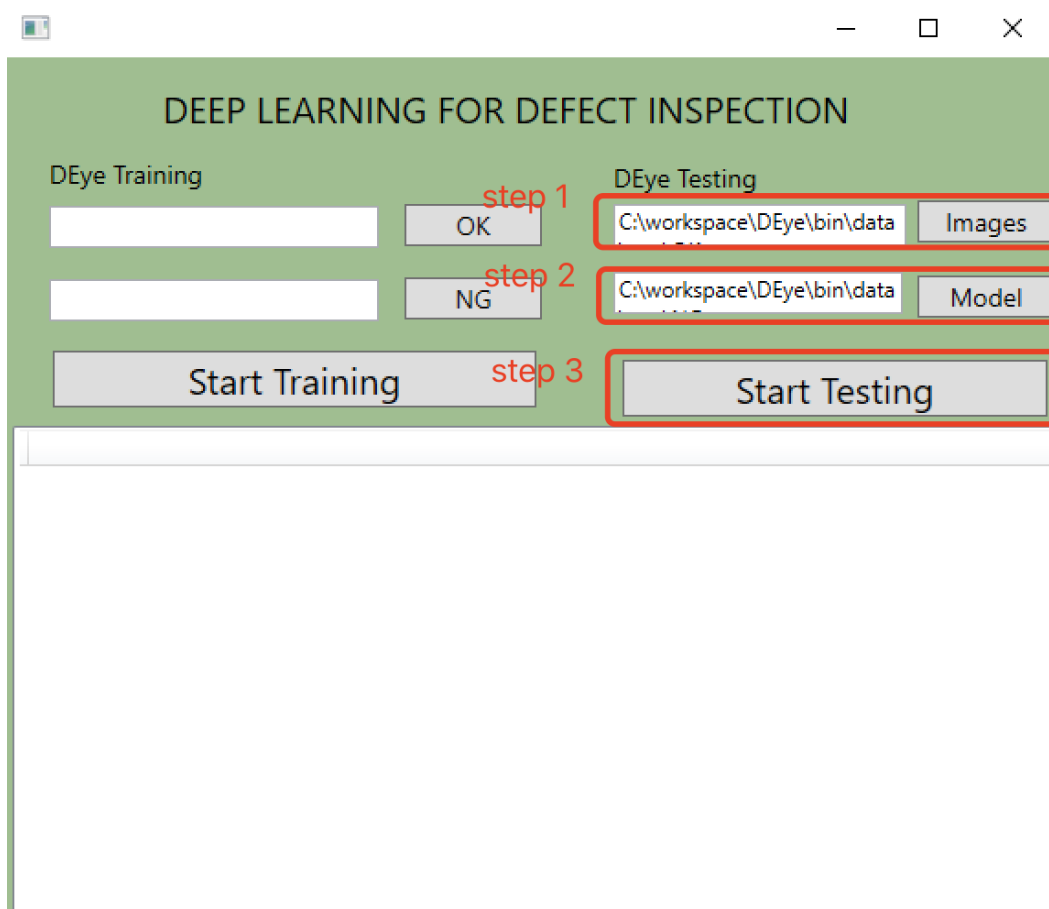


Figure 10: Illustration of model inference.

If you want to change the event of "Start Testing" button, please modify the `bt_test_Click()` function as shown in Figure 11.

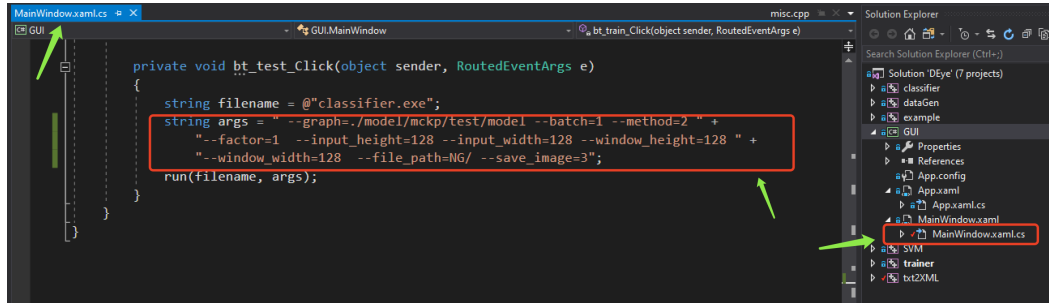


Figure 11: The backend of `bt_test_Click()`.

6.2 Use the command

The classifier module have 5 input arguments, please find the descriptions in following:

- `--method`: 1: input model is Tensorflow pb file, 2: input model is Tensorflow checkpoint file.
- `--graph`: the path of input model.
- `--batch`: batch size of input images.
- `--load_channel`: the number of channels of input image, 1 for grey image, 3 for color images.
- `--input_height`: the height of input images.
- `--input_width`: the width of input images.
- `--window_height`: the height of sliding windows.
- `--window_width`: the width of sliding windows.
- `--file_path`: the path of input images.
- `--save_image`: save output images or not.

Example:

```
classifier.exe --graph=./model/mckp/test/model --batch=1 --method=2 --factor=1 --load_channel=1
--input_height=128 --input_width=128 --window_height=128 --window_width=128 --
file_path=test_data_ng --save_image=3
```

7 Acknowledgement

Welcome to contact with me if you want to be a collaborator of this open source project.

References

- [1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.
- [2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
- [3] Google Inc. Google logging library, 2020. [Online; accessed 30-Oct-2020].
- [4] Sundy. A deep learning-based software for manufacturing defect inspection. <https://github.com/sundyCoder/DEye>, 2017.