



RUNNING FACTORY AI ON WSL2

GPU Acceleration Enabled

Tommy Wu

1. WSL2 Environment Setup :
 - A. Surface Book2 with GTX 1060
 - B. Windows Insider Preview Build 20150 or higher : [register for the Windows Insider Program](#).
 - C. Download Nvidia Driver : <https://developer.nvidia.com/45541-gameready-win10-dch-64bit-international>
 - D. [Enable WSL 2](#) and [install a glibc-based distribution](#) (like Ubuntu or Debian). I choose Ubuntu 18.04
 - E. Check this [guide](#) to install Nvidia container. (Should install this before IoT Edge installation) .
2. Install IoT Edge and Configure your device on IoT Hub by the [doc](#).

[Home](#) > [a9iothub | IoT Edge](#) > [wsl2edge](#) >

Set modules on device: wsl2edge

a9iothub

[Modules](#) [Routes](#) [Review + create](#)

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to pull module images. If you don't specify credentials, IoT Edge will use the default registry. If you specify a registry and it returns an error code 500, it can't find a container registry setting for a module.

NAME	ADDRESS
<input type="text" value="Name"/>	<input type="text" value="Address"/>

IoT Edge Modules

An IoT Edge module is a Docker container you can deploy to IoT Edge devices. It communicates with the IoT Hub. You can import Azure Service IoT Edge modules or specify the settings for an IoT Edge module. Settings are throttled based on the IoT Hub tier and units. For example, for S1 tier, modules can be set 10 times per hour. [Learn more](#)

[+ Add](#) [Runtime Settings](#)

NAME	DESIRED STATUS
WebModule	running
InferenceModule	running

3. Steps to run AI workload for Web Module & Inference Module –
 - A. Start the previous distro environment

```
PS C:\Users\towu> wsl.exe --list --all --verbose
NAME                STATE              VERSION
* Ubuntu-18.04      Stopped            2
ubuntu-iotedge      Stopped            2
PS C:\Users\towu> wsl -d ubuntu-iotedge
root@DESKTOP-5RK65D0:/mnt/c/Users/towu# cd /`
```

- B. Clone the [start.sh](#) on the root folder.
- C. `cd ~ & ./start.sh`

```
root@DESKTOP-5RK65D0:~# ./start.sh
root@DESKTOP-5RK65D0:~# iotedge list
```

- D. Check your module status :

```
root@DESKTOP-5RK65D0:~# iotedge list
NAME                STATUS      DESCRIPTION      CONFIG
InferenceModule     running    Up 5 minutes     intelligentedge/inferencemodule:0.1.10-gpuamd64
WebModule           running    Up 6 minutes     intelligentedge/visionwebmodule:0.1.10-amd64
edgeAgent           running    Up 6 minutes     mcr.microsoft.com/azureiotedge-agent:1.0
edgeHub             running    Up 5 minutes     mcr.microsoft.com/azureiotedge-hub:1.0
root@DESKTOP-5RK65D0:~# |
```

- E. nano the `/etc/docker/daemon.json` as

```
{
  "default-runtime": "nvidia",
  "runtimes": {
    "nvidia": {
      "path": "nvidia-container-runtime",
      "runtimeArgs": []
    }
  },
  "dns": ["8.8.8.8", "8.8.4.4"]
}
```

- F. `systemctl restart docker & iotedge restart`

4. Testing Process :

- A. Check WebModule is running :

```
iotedge logs -f WebModule --tail 500
```

- B. Go to <http://localhost:8080/> and add a new location.

localhost:8080/partIdentification

Vision on Edge

Home > Job Configuration

Part Identification

Select camera: Demo Video [Add Camera](#)

Select parts: aeroplane × bicycle × bird × boat × bottle × bus × car × cat × chair × cow × diningtable × dog × horse × [Add Part](#)

Select location: Demo Location [Add Location](#)

☒ Set up retraining ☐ Send message to cloud

Capture Image

Minimum: 30 %

Maximum: 80 %

The Part contains images , recommend to set the range to Min 0% and Max 0%

Maximum Images to Store: 20

[Configure](#) [Back](#)

C. Choose “demo Pretrained Detection “ and Configure

D. Check the Right-hand side Configuration about GPU(accelerated) : around **10-15 ms** , compared with CPU around 300-500 ms.

Configuration

Threshold to see current model detection result on objects: 10 % [Update Threshold level](#)

Success Rate: **100%** Running on GPU (accelerated): **14.47/ms**

Successful Inferences: 89

Unidentified Items: 0 [Identify Manually](#)

Capture image and model successful result:

E. Verify the logs on inference module : some onnxruntime warning but initializer scalepreprocessor scale.

```

Started Inference...
Press Ctrl+C to exit...
* Serving Flask app "main" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
INFO:werkzeug: * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
INFO:werkzeug:172.18.0.3 - - [17/Jul/2020 18:37:41] "GET /update_retrain_parameters?confidence_min=30&confidence_max=80&
max_images=20 HTTP/1.1" 200 -
INFO:werkzeug:172.18.0.3 - - [17/Jul/2020 18:37:41] "GET /update_iotHub_parameters?is_send=False&threshold=50&fpm=6 HTTP
/1.1" 200 -
INFO:werkzeug:172.18.0.3 - - [17/Jul/2020 18:37:41] "GET /metrics HTTP/1.1" 200 -
INFO:werkzeug:172.18.0.3 - - [17/Jul/2020 18:37:41] "GET /update_cam?cam_type=rtsp&cam_source=sample_video%2Fvideo.mp4 H
TTP/1.1" 200 -
2020-07-17 18:37:41.702261522 [W:onnxruntime:, graph.cc:814 Graph] Initializer scalerPreprocessor_scale appears in graph
inputs and will not be treated as constant value/weight. This may fail some of the graph optimizations, like const fold
ing. Move it out of graph inputs if there is no need to override it, by either re-generating the model with latest expor
ter/converter or with the tool onnxruntime/tools/python/remove_initializer_from_input.py.
2020-07-17 18:37:41.702323123 [W:onnxruntime:, graph.cc:814 Graph] Initializer scalerPreprocessor_bias appears in graph
inputs and will not be treated as constant value/weight. This may fail some of the graph optimizations, like const foldi
ng. Move it out of graph inputs if there is no need to override it, by either re-generating the model with latest export

```

F. Done and Enjoy your WSL2 !