ECP5 Face Identification Quick Start Guide

Application Note

FPGA-AN-02010-1.0

October 2019
Disclaimers
Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.
Contents

Acronyms in This Document ........................................................................................................................................... 4
1. Introduction ........................................................................................................................................................................... 5
   1.1. Design Process Overview ........................................................................................................................................... 5
   2.1. Caffe Installation ......................................................................................................................................................... 7
   2.2. Preparing the Dataset ................................................................................................................................................... 7
   2.3. Training the Machine with Caffe ............................................................................................................................... 8
   2.4. Training the Machine with TensorFlow ..................................................................................................................... 8
   2.5. Generating the Frozen (*.pb) File ............................................................................................................................... 11
3. Generating the Binary File .................................................................................................................................................. 12
   3.1. Programming the Bitstream and Binary files to VIP Board and SD Card .................................................................. 12
Technical Support Assistance .................................................................................................................................................. 13
Revision History ........................................................................................................................................................................ 14

Figures

Figure 1.1. Lattice EVDK with MicroSD Card Adapter Board ................................................................................................. 5
Figure 1.2. Lattice Machine Learning Design Flow .................................................................................................................. 6
Figure 2.1. Default Content of the Train Folder ................................................................................................................... 8
Figure 2.2. Machine Training Contents ................................................................................................................................ 8
Figure 2.3. Update Proto File for Dataset Label File Name .................................................................................................... 8
Figure 2.4. Last Fully-connected Layer Output Number Setting ........................................................................................... 9
Figure 2.5. train.sh Parameter Setting Example ................................................................................................................... 9
Figure 2.6. Executing train.sh Script Command .................................................................................................................. 9
Figure 2.7. Training Status Example ................................................................................................................................... 10
Figure 2.8. Machine Training Output File .......................................................................................................................... 10
Figure 2.9. Machine Training Output File .......................................................................................................................... 11
Figure 2.10. TensorFlow – Launch ..................................................................................................................................... 11
Figure 2.11. *.pb File Generation from Checkpoint ........................................................................................................... 11
## Acronyms in This Document

A list of acronyms used in this document.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKPT</td>
<td>Checkpoint</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field-Programmable Gate Array</td>
</tr>
</tbody>
</table>
1. Introduction

This document provides a quick guide on how to train a machine and create a frozen file for the Lattice Machine Learning development using the Lattice’s Embedded Vision Development Kit. It assumes that the reader is familiar with the basic Lattice FPGA design flow and mainly focuses on the Machine Learning part of the overall development process. For detailed instructions of the design flow described in this document, refer to the ECP5-Face-Identification-Reference-Design (FPGA-RD-02062).

![Figure 1.1. Lattice EVDK with MicroSD Card Adapter Board](image-url)
1.1. Design Process Overview

The design process involves the following steps:
- Setting up the basic environment
- Preparing the dataset
- Training the machine
- Creating the frozen file (*.pb)
- Creating the binary file with Lattice sensAI™ 2.1 program
- Programming the binary and bitstream files to VIP board and SD card

![Design Flow Diagram](image)

Figure 1.2. Lattice Machine Learning Design Flow

2.1. Caffe Installation

For more information on how to install Caffe, go to https://caffe.berkeleyvision.org/.

Note: Caffe has some dependencies of environment and library packages, Lattice cannot help on resolving all installation issues caused by system dependencies.

2.2. Preparing the Dataset

Use the links below to download the training and testing VGGFace2 dataset:

- Training dataset – http://zeus.robots.ox.ac.uk/vgg_face2/get_file?fname=vggface2_train.tar.gz
- Testing dataset – http://zeus.robots.ox.ac.uk/vgg_face2/get_file?fname=vggface2_test.tar.gz

For the detailed procedure in preparing the dataset, refer to the Preparing the Dataset section in ECP5-Face-Identification-Reference-Design (FPGA-RD-02062).
2.3. Training the Machine with Caffe

1. Check three script files in the train folder as shown in Figure 2.1.

![Figure 2.1. Default Content of the Train Folder](image1)

2. Copy the label files train_new.txt and test_new.txt to the train folder.

![Figure 2.2. Machine Training Contents](image2)

3. Update the proto file faceid.proto for Image dataset source.

```protobuf
name: "FaceID_vgpu"
layer {
  name: "train data"
  type: "ImageData"
  top: "data"
  top: "label"
  transform_param {
    mirror: true
    #crop_size: 99
    mean_value: 128
    mean_value: 128
    mean_value: 128
    max_rotation_angle: 20
  }
  param {
    transform_param {
      #mirror: true
      #crop_size: 120
      #batch_size: 64
      #batch_size: 32
      #batch_size: 16
      new_height: 90
      new_width: 90
      shuffle: true
    }
    include { phase: TRAIN }
  }
  layer {
    name: "test_data"
    type: "ImageData"
    top: "data"
    top: "label"
    transform_param {
      mirror: false
      mean_value: 128
      mean_value: 128
      mean_value: 128
    }
    param {
      transform_param {
        #mirror: true
        #crop_size: 120
        #batch_size: 64
        #batch_size: 32
        #batch_size: 16
        new_height: 90
        new_width: 90
        shuffle: true
      }
      include { phase: TEST }
    }
  }
}
```

![Figure 2.3. Update Proto File for Dataset Label File Name](image3)
4. Set the output number of FC6 layer.
   Set ‘num_output’ to the number of dataset class.
   The training code of face identification is for the object classification as MNIST application. Therefore, the final output number of the fully-connected model is the number of object class. The default value of FC6 output is 9295.

5. Update the following variables in train.sh as per your environment setup.
   - CAFFE_ROOT – with compiled caffe location.
   - OPT – GPU number or comment out to use CPU only.

6. Run the train.sh script.
7. Create the inference network model proto file.
   The inference network mode is different from the training proto file. The inference model does not use Drop5 and FC6 layer.
   Remove train_data, test_data, drop5, loss, accuracy, fc6, and center_loss.
   Lattice provides the reference proto file of Inference model file faceid_demo.proto.
   The inference proto file is used in the part of generating binary file.
   Refer to the Preparing the Dataset section in ECP5-Face-Identification-Reference-Design (FPGA-RD-02062).
2.4. Training the Machine with TensorFlow

To train the machine with TensorFlow:

1. Run the command in the code root directory:

```bash
$ python face-identification.py --data_dir=/path of data directory
```

2. Check the training status on TensorBoard by running the command below.

```bash
$ tensorboard --logdir=/path of log directory
```

2.5. Generating the Frozen (*pb) File

Use the command below to generate the frozen protobuf(*pb) file:

```bash
$ python face-identification.py --log_dir=/path of log(checkpoint) directory --freeze_model
```

---

© 2019 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal.
All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.
3. Generating the Binary File

For the detailed procedure in creating the binary file, refer to the Creating Binary File with sensAI section ECP5-Face-Identification-Reference-Design (FPGA-RD-02062).

3.1. Programming the Bitstream and Binary files to VIP Board and SD Card

For the detailed procedure in flashing the bitstream file to the VIP board and flashing the binary file to the SD card, refer to refer to the Preparing the Dataset section in ECP5-Face-Identification-Reference-Design (FPGA-RD-02062).
Technical Support Assistance
Submit a technical support case through www.latticesemi.com/techsupport.
## Revision History

**Revision 1.0, October 2019**

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>