Deep Learning Face Recognition on ARTIK and Raspberry Pi

|  |
| --- |
| **INTRODUCTION**  Training a deep neural network requires moderate computing resources, such as GPU to accelerate  the process. However, deep learning inference requires less computing power. It is still a challenging  task to build deep learning applications on IoT devices.  With the advance of IoT hardware, such as Samsung ARTIK and Rapberry Pi’s advanced instruction  set, more deep learning inference tasks can be made possible. This work focuses on leveraging the  deep learning models to build computer vision applications on these platforms.  This instruction describes the step-by-step procedure for setting up a deep learning application  environment for computer vision applications, such as facial recognition using Samsung ARTIK  Module or Raspberry Pi. It explains the following topics:   * Install a pre-built Tensorflow library on ARTIK or Raspberry Pi * Use a subset of MS-Celeb-1M dataset to train a simple deep neural network for facial   recognition   * Using Keras with TensorFlow backend and run it on ARTIK or Raspberry Pi   By following this procedure, you can create a sample facial recognition application on the Samsung  ARTIK or Raspberry Pi Module. |
| **INSTALLING TENSORFLOW FROM BINARY**  There are two ways of installing TensorFlow, one is to build directly from source code and the other  is to use a .whl binary file to install. Building from the source will take a huge amount of time and  it will not benefit you unless your configuration is very different from this instruction. Therefore,  we recommend you use the pre-compiled version provided by this app note.   1. First, you need a few dependencies for the installation: 2. Download tensorflow-1.3.0rc0-cp27-cp27mu-linux\_aarch64.whl from Github page 3. Install TensorFlow. Power up your ARTIK/Pi and go to the folder where the TensorFlow library   .whl file is located.   1. Test the TensorFlow installation, it should print "1.3.0-rc0" if the installation is successful. 2. Keras and other dependencies will also be needed for running inference on ARTIK.     apt-get install python-dev python-pip python-setuptools liblapack-dev  apt-get install gfortran  python-numpy python-scipy  pip install ~/tensorflow-1.3.0rc0-cp27-c27mu-linux\_aarch64.whl    python -c "import tensorflow as tf; print tf.\_\_version\_\_"    pip install keras  apt-get install libhdf5-10  apt-get install python-h5py |
| **PREPARING HARDWARE FOR RUNNING INFERENCE ON**  **ARTIK or Raspberry Pi**  Once you have successfully installed TensorFlow and the dependencies, you have all the software  ready for inference. Before starting the inference, you need to install a memory drive (USB key or  Micro-SD card) as a swap in order to successfully load large data and model files.  You need at least 2GB memory for the swap space. It is highly recommended you use large (8GB  and above) and fast memory drives for the swap space.   1. First, insert one memory drive to one of the USB ports or Micro-SD card slot (if you use Micro-SD   as the swap media) on ARTIK/Pi, then find the device path /dev/XXX by running the  command below:  sudo blkid  The device path should look like /dev/sda1 or /dev/mmcblk1. Note that  /dev/mmcblk0 is your internal flash (eMMC) disk, do not unmount it when executing the  flowing unmount command:  sudo umount /dev/XXXXXX  We use /dev/sda1 for swap as an exmaple. The aforementioned  command becomes:  sudo umount /dev/sda1   1. Format your device for the swap space:   sudo mkswap /dev/sda1   1. Execute the command again to copy the UUID associated with the /dev/sda1.   sudo blkid   1. Edit your /etc/fstab file to register the device as swap space with your favorite text editor:   sudo vi /etc/fstab   1. On a new line, write the following information to the file and save. Replace the X pattern with the   actual UUID of /dev/sda1:  UUID=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXX none swap sw, pri=5 0 0   1. Run the following command to enable the swap space that we just added:   sudo swapon -a |
| **DATASET**  MS-Celeb-1M is a well-known benchmark dataset for testing Face Recognition algorithm  performance. We select 6 celebrities: Meryl Streep, Harrison Ford, Charlize Theron, Matthew  McConaughey, Mackenzie Foy, and Isaac Hempstead Wright from MS-Celeb-1M dataset to  construct a small dataset to train and evaluate our neural network. The small dataset contains over  100 images for each celebrity and 678 images in total, we name this small dataset as FamilyFaces  dataset. For testing purpose, this app note only selects 6 celebrities from the dataset and trains our  deep neural networks.  First, download the entity list from http://www.msceleb.org/download/list. Once it is  downloaded you will have a file named: Top1M\_MidList.Name.tsv  Second, download the aligned face images from <http://www.msceleb.org/download/aligned>,  unzip the downloaded file: FaceImageCroppedWithAlignment.zip (61GB), you will have a tsv file as  below:  FaceImageCroppedWithAlignment.tsv (85GB) Each line of the tsv file contains one image record  with 7 columns, delimited by TAB:    *Figure 3. Selected Celebrities*  Column1: Freebase MID  Column2: ImageSearchRank  Column3: ImageURL  Column4: PageURL  Column5: FaceID  Column6: FaceRectangle\_Base64Encoded (four floats, relative coordinates of  UpperLeft and BottomRight corner)  Column7: FaceData\_Base64Encoded |
| In order to find corresponding records for persons of interest from the unzipped tsv file, we provide a  simple Pythor script to process both the entity list file and the face image file to prepare the  sub-dataset. Copy process\_family\_names.py from the app note package. Then put the Python script  in the same folder as Top1M\_MidList.Name.tsv, and FaceImageCroppedWithAlignment.tsv.  Make sure the Python script and both .tsv files are under the same folder on your development  machine, not on ARTIK/Pi. To get a subset of 6 celebrities face dataset, you need to run the  following command under your Linux console:  The script, if executed successfully, will generate a file named "FamilyFaces.pkl", which is about  100MB. This file contains the face images in a byte stream fashion which is more convenient for  us to train and test our neural network models.  python process\_family\_names.py |
| **FACE RECOGNITION WITH SIMPLE CONVOLUTIONAL**  **NEURAL NETWORK**  In order to show the capability of ARTIK/Pi for running deep learning applications, we design a very  simple neural network with only two convolutional layers (32 filters and 64 filters), each followed by  a max pooling layer and a dropout layer (0.7 rate) for regularization. The output of the last dropout  layer is flattened and fed into a fully connected layer (128 hidden units) followed by a dropout  layer (0.7 rate) and the final softmax output (6 classes). The script “familyfaces\_cnn.py” is the  detailed implementation of this simple ConvNet. This script is in the downloadable from Github pag.  You must run the following script on your own development  computer, not ARTIK boards. It is recommended you use Nvidia GPU, not CPU, to run this training  process for time-saving purposes.  You are required to install Keras and its dependencies before running the code on your GPU  development machine for training.  Now, put FamilyFaces.pkl and this CNN code under the same directory on your GPU development  machine, and then execute the code;  There are about 24 million parameters in total. *Figure 4* shows the neural network specification:    sudo apt-get install python-scipy  sudo pip install protobuf  sudo pip install backports.weakref  pip install keras  pip install scikit-learn  python familyfaces\_cnn.py    *Figure 4. Face Recognition with Simple CNN* |
| *Figure 5* shows the TensorBoard output of the accuracy curve during the training process on an  Nvidia Titan X Pascal GPU. The orange line indicates the actual accuracy value. The accuracy  is 91.17%, and the actual inference results may vary a little due to the stochastic nature of the  training and testing with the neural networks.  After the training process, the script will save the mode file "familyfaces\_cnn.h5" (274MB)  automatically and you are ready to test the model on the ARTIK/Pi board by copying the model  file to ARTIK/Pi.    *Figure 5. Simple CNN Result* |
| \ |
| **INFERENCE ON ARTIK and Raspberry Pi**  Once the model is trained, you can easily perform the inference by transferring the cPickle data file,  FamilyFaces.pkl and the trained model, familyfaces\_cnn.h5 file to ARTIK.  Execute the following command to make the inference.  *Figure 6* shows the outcome of the inference.  python familyfaces\_inference.py |

*Figure 6. Inference Outcome Screen*