Final exercise

NXSOL-OJT-2024

Exported on 2024-01-25 17:38:36

Table of Contents

1 Set up 4

1.1 Download tensorflow and its dependencies 4

1.2 Root directory after cloning 4

1.3 Local.conf 4

1.4 Add the layers 5

2 Build image 6

2.1 Set the build environment 6

2.2 Start build 6

3 Verify 7

3.1 Run the built image with slrip + kvm + 5GB memory: 7

3.2 Verify the install 7

3.3 Run tutorial case 7

3.4 TensorFlow/TensorFlow Lite C++ Image Recognition Demo 8

* This tensorflow image is based on the hello world image.
* Reference: https://git.yoctoproject.org/meta-tensorflow/ (meta-tensorflow/BUILD.md)

# Set up

## Download tensorflow and its dependencies

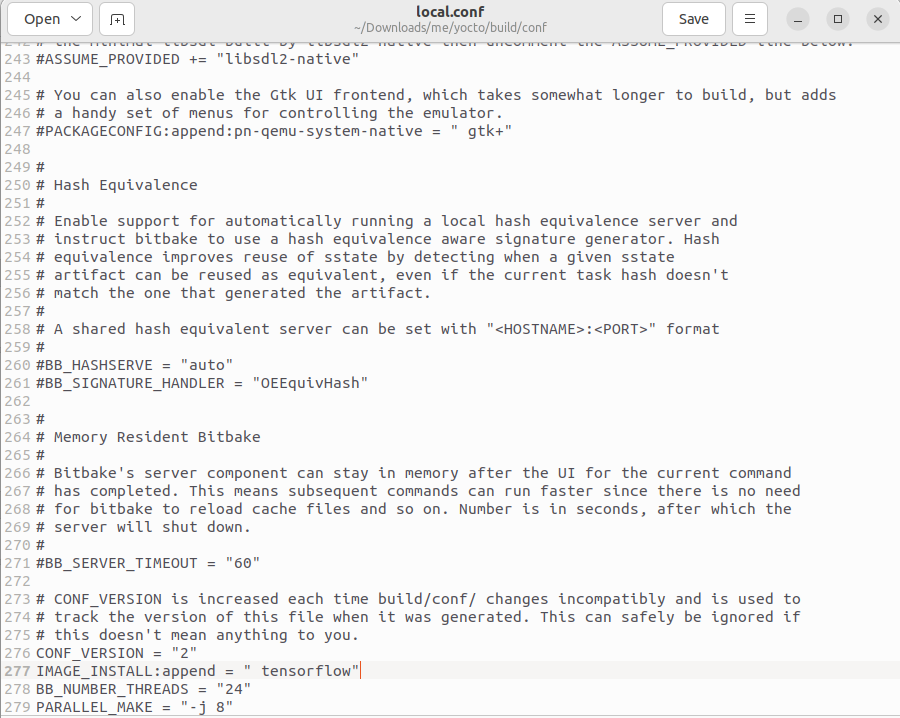
|  |
| --- |
| git clone -b kirkstone git://git.yoctoproject.org/meta-tensorflow  git clone -b kirkstone git://git.openembedded.org/meta-openembedded |

## Root directory after cloning

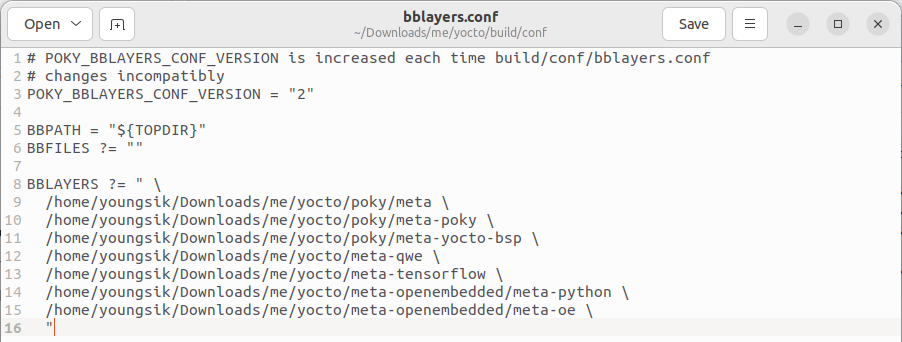


## Local.conf

* Add the tensorflow package to the final image
* Speed up the build by setting multi-threads



## Add the layers



# Build image

## Set the build environment

|  |
| --- |
| source poky/oe-init-build-env ; cd .. |

## Start build

|  |
| --- |
| bitbake -k qwe-image |

# Verify

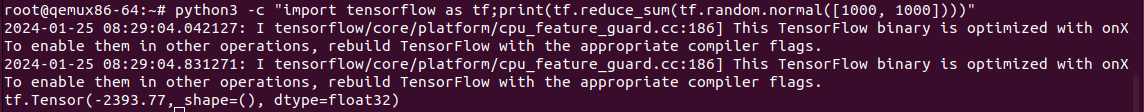
## Run the built image with slrip + kvm + 5GB memory:

|  |
| --- |
| runqemu qemux86-64 qwe-image slirp kvm qemuparams="-m 5120" nographic |

## Verify the install

|  |
| --- |
| python3 -c "import tensorflow as tf;print(tf.reduce\_sum(tf.random.normal([1000, 1000])))" |

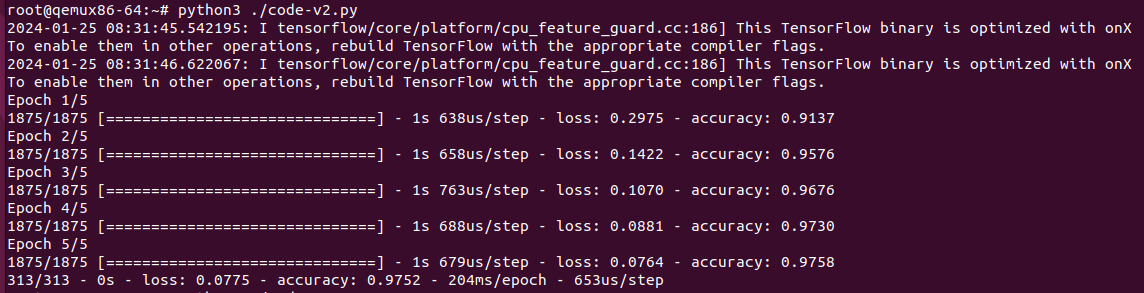
Result



## Run tutorial case

|  |
| --- |
| cat >code-v2.py <<ENDOF  **import** tensorflow as tf  mnist = tf.keras.datasets.mnist    (x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()  x\_train, x\_test = x\_train / 255.0, x\_test / 255.0    model = tf.keras.models.Sequential([  tf.keras.layers.Flatten(input\_shape=(28, 28)),  tf.keras.layers.Dense(128, activation='relu'),  tf.keras.layers.Dropout(0.2),  tf.keras.layers.Dense(10)  ])    predictions = model(x\_train[:1]).numpy()  tf.nn.softmax(predictions).numpy()  loss\_fn = tf.keras.losses.SparseCategoricalCrossentropy(from\_logits=True)  loss\_fn(y\_train[:1], predictions).numpy()    model.compile(optimizer='adam',  loss=loss\_fn,  metrics=['accuracy'])  model.fit(x\_train, y\_train, epochs=5)  model.evaluate(x\_test, y\_test, verbose=2)    probability\_model = tf.keras.Sequential([  model,  tf.keras.layers.Softmax()  ])  probability\_model(x\_test[:5])      ENDOF |

Result



## TensorFlow/TensorFlow Lite C++ Image Recognition Demo

|  |
| --- |
| time label\_image.lite |

Result

