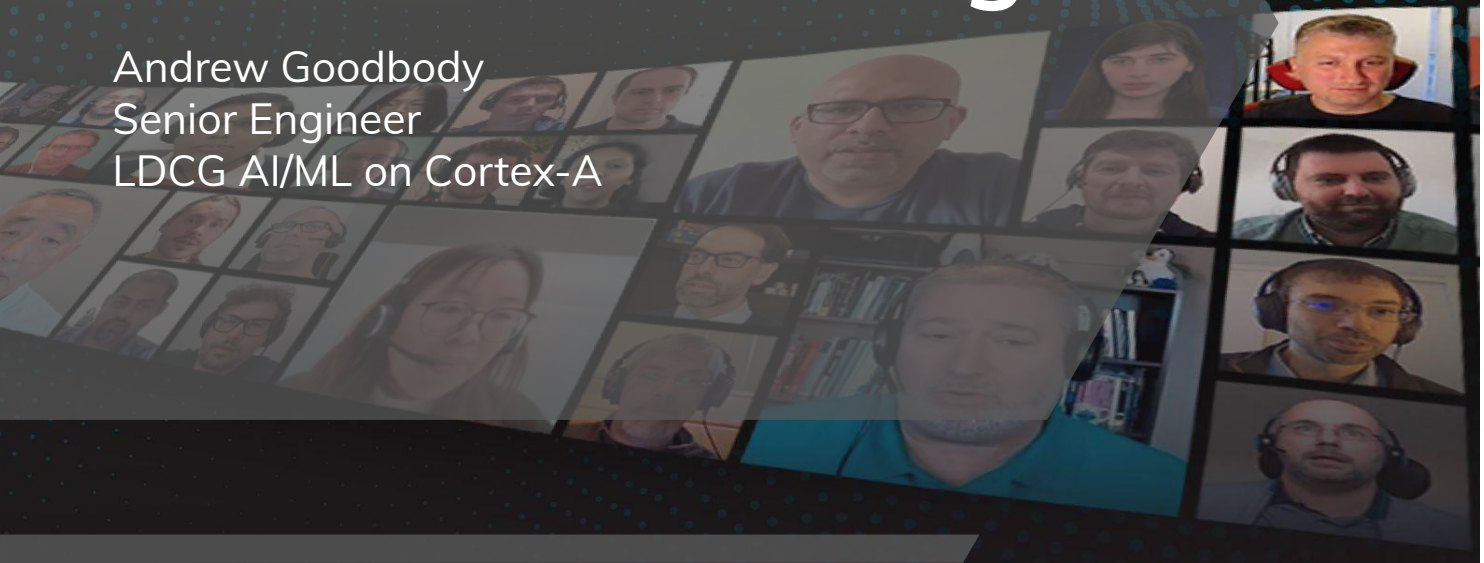
Testing TensorFlow AARCH64 Packages

Andrew Goodbody
Senior Engineer
LDCG AI/ML on Cortex-A



Linaro Connect
VIRTUAL • FALL 2021

TensorFlow - what is it?

TensorFlow is an extremely popular framework for developers to build their Machine Learning powered applications.

https://www.tensorflow.org/

TensorFlow is written in Python but also makes use of native code and libraries

TensorFlow

Why do we need to test TensorFlow packages?

- The use of native code and libraries is the major issue for the AARCH64 world.
- Obviously packages built for other CPU architectures will not work on AARCH64
- The first part of that work is done. There is support for AARCH64 in sufficient places that TensorFlow can be built for AARCH64.
- This work is of course continuing. Now that AARCH64 is supported of course we want to improve its performance.
- But we now come to the crux of it. Although TensorFlow can be built for AARCH64 there are currently no officially distributed packages for AARCH64.

So what will be tested?

- For a while now Linaro has been using its Jenkins CI resources to build both releases of TensorFlow as they are published, but also the head of the source tree on a nightly basis.
- We wish to test both the Linaro builds of the releases but also the nightly builds.
- Testing the releases will ensure that Linaro has built the packages correctly and this will be an assurance to all who use the Linaro built packages that they will perform correctly.
- As well as ensuring that no code has been added to TensorFLow that will not run on AARCH64 testing the nightly builds will allow Linaro to assess the performance of TensorFlow over time and so pick up any significant changes, either positive or negative, and ensure that this feedback can be given to the TensorFlow team in case remedial action is needed.

How will TensorFlow be tested

Initially there will be two main parts to the testing that will be performed. Firstly there are the in tree unit tests.

Unit Tests

- The unit tests already exist in the source tree and were developed alongside the main source code by the developers.
- The unit tests will give us the confidence that TensorFlow has been built correctly and also that no CPU specific code that does not handle AARCH64 has been added to the execution path.

How will TensorFlow be tested

Secondly we actually test the installed packages themselves using resnet50

resnet50

- resnet50 is a well known model that is widely used in ML as a simple verification test.
- The reduced imagenet collection of images is publicly downloadable.
- resnet50 can be run using MLCommons Inference giving a well controlled and repeatable environment. It was formerly known as MLPerf.
- resnet50 will give us a simple measure for gauging the performance of TensorFlow

Unit Tests

- The unit tests are included in the TensorFlow source tree.
- The unit tests themselves need to built in the same source tree as you wish to test.
- This does ensure that the unit tests are run on the same code that you are going to package and using the same CPU architecture.

Unit Tests

The command line used to build and invoke the tests needs to have the same options as used to build TensorFlow. So if your build line looks like this:

```
bazel build $extra_args --config=v2 --copt="-mcpu=${CPU}" --copt="-march=${ARCH}" --copt="-O3"
--copt="-fopenmp" --cxxopt="-mcpu=${CPU}" --cxxopt="-march=${ARCH}" --cxxopt="-O3"
--cxxopt="-fopenmp" --linkopt="-lgomp  -lm" //tensorflow/tools/pip_package:build_pip_package
//tensorflow:libtensorflow_cc.so //tensorflow:install_headers
```

Your test line might look something like this

```
bazel test $extra_args --flaky_test_attempts=3 --test_output=all --cache_test_results=no --config=v2
--copt="-mcpu=${CPU}" --copt="-march=${ARCH}" --copt="-O3" --copt="-fopenmp"
--cxxopt="-mcpu=${CPU}" --cxxopt="-march=${ARCH}" --cxxopt="-O3" --cxxopt="-fopenmp"
--linkopt="-lgomp -lm" -- //tensorflow/core/kernels/... -//tensorflow/core/kernels/mlir_generated/...
```

Unit test issues

We have discovered the following issues with running some of the unit tests on AARCH64

- Some tests that also fail on x86 and are skipped in the results. We will not worry about these for the moment.
- tanh and sigmoid function numerical errors.

```
Failed example:
    tf.math.tanh(x)
Expected:
    <tf.Tensor: shape=(8,), dtype=float32, numpy=
    array([-1.        , -0.99990916, -0.46211717,  0.7615942 ,  0.8336547 ,
            0.9640276 ,  0.9950547 ,  1.        ], dtype=float32)>
Got:
    <tf.Tensor: shape=(8,), dtype=float32, numpy=
    array([-0.99999976, -0.99990916, -0.46211717,  0.7615942 ,  0.8336546 ,
            0.9640276 ,  0.9950547 ,  0.99999976], dtype=float32)>
```

Unit test issues

We have discovered the following issues with building some of the unit tests

- Build issue when mlir_generated tests are included, possibly due to non-installation of llvm
- Build issue with mlir_generated tests when using non-system gcc and glibc
  - https://github.com/tensorflow/tensorflow/issues/50873 - has just been fixed

resnet50

Running resnet50 gives a simple measure of performance. Of course it does not cover all use cases of TensorFlow and so will not give a comprehensive view of all possible performance changes. We intend to extend the range of tests performed in the future to give better coverage by adding the use of different models. At the moment we wish to get the process started first of all.

The use of MLCommons Inference is a convenience to give us a simple way to run the test. We do not use the whole of the MLCommons Inference suite of code, far from it. There are multiple parts to the MLCommons Inference suite and some of them can take a few days to run. This is clearly not an option for nightly builds and would unnecessarily delay release builds as well.

Obtaining resnet50 model and data

1. python -m pip install ck
2. ck pull repo:ck-env
3. ck install package --tags=image-classification,dataset,imagenet,aux
4. ck install package --tags=image-classification,dataset,imagenet,val
5. cp ~/CK-TOOLS/dataset-imagenet-ilsvrc2012-aux-from.dividiti/val.txt ~/CK-TOOLS/dataset-imagenet-ilsvrc2012-val-min/val_map.txt
6. mkdir ~/tf_test
7. cd ~/tf_test
8. wget https://zenodo.org/record/2535873/files/resnet50_v1.pb

Prepare your test environment

- If using a Python virtual environment this should be initialised and activated in the usual way
- Obtain and install the version of TensorFlow that you wish to test.
  - python3 -m pip install --extra-index-url https://snapshots.linaro.org/ldcg/python/tensorflow-manylinux/11/ tensorflow-cpu

MLCommons Inference building

1. git clone https://github.com/mlcommons/inference.git
2. git checkout r0.7
3. cd inference/loadgen
4. CFLAGS="-std=c++14 -Wp,-U_GLIBCXX_ASSERTIONS" python setup.py develop
5. cd ../vision/classification_and_detection
6. python setup.py develop
7. export MODEL_DIR=${HOME}/tf_test
8. export DATA_DIR=${HOME}/CK-TOOLS/dataset-imagenet-ilsvrc2012-val-min
9. ./run_local.sh tf resnet50

You need to disable C++ assertions when building loadgen due to https://github.com/mlcommons/inference/issues/864

Sample resnet50 results

$ ./run_local.sh tf resnet50

INFO:main:Namespace(accuracy=False, backend='tensorflow', cache=0, count=None, data_format=None, dataset='imagenet', dataset_list=None, dataset_path='/home/builder/CK-TOOLS/dataset-imagenet-ilsvrc2012-val-min', find_peak_performance=False, inputs=['input_tensor:0'], max_batchsize=32, max_latency=None, mlperf_conf='../../mlperf.conf', model='/home/builder/mlperf/resnet50_v1.pb', model_name='resnet50', output='/home/builder/1/mlperf_build/mlperf/r0.7/mlperf/vision/classification_and_detection/output/tf-cpu/resnet50', outputs=['ArgMax:0'], profile='resnet50-tf', qps=None, samples_per_query=None, scenario='SingleStream', threads=4, time=None, user_conf='user.conf')

INFO:imagenet:reduced image list, 49500 images not found

INFO:imagenet:loaded 500 images, cache=0, took=0.4sec

WARNING:tensorflow:From /home/builder/1/mlperf_build/mlperf/r0.7/mlperf/vision/classification_and_detection/python/backend_tf.py:48: FastGFile.__init__ (from tensorflow.python.platform.gfile) is deprecated and will be removed in a future version.

Instructions for updating:

Use tf.gfile.GFile.

WARNING:tensorflow:From /home/builder/tf_test/venv/lib/python3.6/site-packages/tensorflow/python/tools/strip_unused_lib.py:88: extract_sub_graph (from tensorflow.python.framework.graph_util_impl) is deprecated and will be removed in a future version.

Instructions for updating:

Use `tf.compat.v1.graph_util.extract_sub_graph`

WARNING:tensorflow:From /home/builder/tf_test/venv/lib/python3.6/site-packages/tensorflow/python/tools/optimize_for_inference_lib.py:117: remove_training_nodes (from tensorflow.python.framework.graph_util_impl) is deprecated and will be removed in a future version.

Instructions for updating:

Use `tf.compat.v1.graph_util.remove_training_nodes`

INFO:main:starting TestScenario.SingleStream

TestScenario.SingleStream qps=9.97, mean=0.1002, time=102.699, queries=1024, tiles=50.0:0.1001,80.0:0.1017,90.0:0.1026,95.0:0.1033,99.0:0.1055,99.9:0.1067

Testing platforms

Unit tests

As these must be run in the build tree, the unit tests will be added to the Jenkins build jobs and will report from there.

resnet50

We are adding machines to our in house instance of LAVA that will be used for running the resnet50 tests. LAVA gives us the ability to deploy tests on different platforms with varying operating systems installed.

Our current plans are to use an N1SDP from ARM, a Huawei Taishan server with Kunpeng 916 CPU and a node of our Fujitsu FX700 with its A64FX CPU.

LAVA

- LAVA or Linaro Automated Validation Architecture is a continuous integration system for deploying operating systems onto hardware and for running tests.
- We will use the capabilities of LAVA to be able to run multiple tests with varying operating systems on the same hardware.
- You can discover more about LAVA at https://www.lavasoftware.org/

Neoverse N1 SDP

The ARM Neoverse N1 SDP is a software development platform from ARM and is based around the N1 CPU.

https://community.arm.com/developer/tools-software/oss-platforms/w/docs/458/neoverse-n1-sdp

Unfortunately due to licensing restrictions the default build of the EDK2 UEFI firmware for the N1SDP does not include support for its network controller.

This meant some development time had to be dedicated to adding this support to UEFI firmware build. There is a UEFI driver binary available for download from the manufacturer of the network controller. This then needed to be added to the build along with the other drivers etc that are needed to enable network boot support.

Other machines

Huawei Taishan 2280

This is a 2U 19” rack mount server with a Kunpeng 916 CPU. While not the latest machine, it has reasonable performance.

Fujitsu FX700

This machine is housed in a 2U 19” rack chassis and is made up of 4 blades each with 2 nodes. This gives a total of 8 nodes per chassis. Each node has a A64FX CPU with 32GB of High Bandwidth Memory.

What is the point of this testing?

As mentioned earlier in this talk, there are no officially built and released versions of TensorFlow for AARCH64. We would like to change this fact and make AARCH64 packages of TensorFlow freely available to the community. But of course we do not wish to make available packages that do not meet the needs of the community by failing to work correctly. The testing that we will be doing is to ensure that not only does the code work today but that it will continue to work tomorrow. But also we hope to be able to see and measure improvements in performance as well.

What next

Very shortly we hope to have Linaro built packages of TensorFlow for AARCH64 available for download from PyPi. Initially these will have been tested manually but we will be automating the testing process as described in this talk so that future builds will benefit from this.

Of course we would welcome your help and involvement in ensuring that the unit tests are suitable for use on AARCH64. Also as we look to extend the range of tests that are used any contributions in that area would be appreciated.

Thank you

Accelerating deployment in the Arm Ecosystem

LVC21F-318 Testing TensorFlow AARCH64 Packages

Andrew Goodbody

Linaro Connect
VIRTUAL • FALL 2021