■ SeeDot で遊んでみる

STM32F407 Discovery Kit を Arduino で使う.

- Arduino IDE をインストール
- ボードマネージャの URL を追加 https://raw.githubusercontent.com/stm32duino/BoardManagerFiles/master/STM32/package_stm_index.json
- ツール → ボードマネージャ で，STM32 Core をインストール
- ボードで Discovery を，Board part number で STM32F407G-DISC1 を選択
- サンプルの 01.Basics → Blink が動作するのを確認する (LD4 がチカチカする)

SeeDot を使う

仮想環境用意して，https://github.com/microsoft/EdgeML/tree/master/Tools/SeeDot に書いてある手順で実行する.

- venv を用意

```
python3 -m venv EdgeML
source ./EdgeML/bin/activate
```

- GitHub から clone して環境準備

```
git clone https://github.com/Microsoft/EdgeML
cd EdgeML/tf/
pip install -r requirements-cpu.txt
pip install -e .
```

- usps10 で ProtoNN を学習

```
cd examples/ProtoNN
python fetch_usps.py
python process_usps.py
mkdir usps10/output
python protoNN_example.py --data-dir ./usps10 --projection-dim 25 --num-prototypes 55 --epochs 100 -sW 0.3 -o usps10/output
```

- Arudino 向けにビルド

```
cd ../../../Tools/SeeDot
mkdir arduino
 python SeeDot.py -a protonn --train ../../tf/examples/ProtoNN/usps10 /train.npy --test ../../tf/examples/ProtoNN/usps10/test.npy --model ../../tf/examples/ProtoNN/usps10/output -o arduino
```

    - こんな感じですすむ

```
(EdgeML) miyo@tama:% python SeeDot.py -a protonn --train ../../tf/examples/ProtoNN/usps10/train.npy --test ../../tf/examples/ProtoNN/usps10/test.npy --model ../../tf/examples/ProtoNN/usps10/output -o arduino

================================
Executing on protonn for Arduino
--------------------------------
Train file: ../../tf/examples/ProtoNN/usps10/train.npy
Test file: ../../tf/examples/ProtoNN/usps10/test.npy
Model directory: ../../tf/examples/ProtoNN/usps10/output
================================
```

```
----------------------
Collecting profile data
----------------------
Generating input files for float training dataset...done

Build...success
...
Accuracy is 89.985%

----------------------------
Generating code for arduino...
----------------------------

Generating input files for fixed testing dataset...done

Generating code...completed

Arduino sketch dumped in the folder arduino
```

- できたもの

```
(EdgeML) miyo@tama:% ls arduino
arduino.ino  config.h  input/  library.h  model.h  predict.cpp  predict.h
```

ボードとの接続

- UART2 を使う．PA2 が TX, PA3 が RX

```
HardwareSerial Serial1(USART2);
```

  - F407 上のシリアル通信について - ~/.arduino15/packages/STM32/hardware/stm32/1.5.0/variants/DISCO_F407VG/PeripheralPins.c
- なんかでた

```
4596: Predicted label: 9; True label: 9; Correct prediction
4597: Predicted label: 9; True label: 9; Correct prediction
4598: Predicted label: 9; True label: 9; Correct prediction
4599: Predicted label: 9; True label: 9; Correct prediction
4600: Predicted label: 9; True label: 9; Correct prediction

-----------------------
Average prediction time:
1380.68
-----------------------

4601: Predicted label: 9; True label: 9; Correct prediction
4602: Predicted label: 9; True label: 9; Correct prediction
4603: Predicted label: 9; True label: 9; Correct prediction
```

- ソースコードによると，micros() を使って測定した値をイテレーション回数で割ってるみたい
  - 入力データは flash 上に用意されているデータ
  - 1 イテレーションあたり 1380.68u 秒で推論できてるよ，ってことみたい.
- Accuracy Mode にすると，シリアルから入力を受けつけて推論できるみたい

メモ

- usps - https://www.kaggle.com/bistaumanga/usps-dataset

■ TVM/AWS-F1 で遊んでみた (失敗)

https://github.cfom/dmlc/tvm/blob/master/docs/deploy/aws_fpga.md をやってみる.

まだうまくいってない.

やってみたこと

ビルド用にセットアップした c4.4xlarge マシンにログインして，AWS-F1 用の環境変数を

```
% source src/project_data/aws-fpga/sdaccel_setup.sh
% source ${XILINX_SDX}/settings64.sh
```

でセット.
TVM を，https://docs.tvm.ai/install/from_source.html を参考にビルドする.
LLVM がいるみたいなので LLVM 4.0.1 をビルド．CMake が古いので CMake から ...

```
% wget https://cmake.org/files/v3.8/cmake-3.8.2.tar.gz
% wget http://releases.llvm.org/4.0.1/llvm-4.0.1.src.tar.xz
% wget http://releases.llvm.org/4.0.1/cfe-4.0.1.src.tar.xz
% tar xvf cfe-4.0.1.src.tar.xz
% tar xvf llvm-4.0.1.src.tar.xz
% mv cfe-4.0.1.src llvm-4.0.1.src/tools/clang
% cd llvm-4.0.1.src
% mkdir build; cd build
% cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_INSTALL_PREFIX=$HOME/llvm-4.0.1 ../
% make -j8 && make install
% export PATH=$HOME/llvm-4.0.1/bin:$PATH
% sudo yum install python36 python36-devel python36-pip
% sudo pip3 install numpy decorator
```

で準備してから

```
% git clone --recursive https://github.com/dmlc/tvm
% cd tvm
% git submodule init
% git submodule update
% mkdir build
% cp cmake/config.cmake build
% cd build
```

config.cmake の

```
set(USE_LLVM OFF)
set(USE_SDACCEL OFF)
set(USE_OPENCL OFF)
```

を

```
set(USE_LLVM ON)
set(USE_SDACCEL ON)
set(USE_OPENCL ON)
```

に変更して,

```
% cmake ..
% make -j8
```

おわったら,

```
% export TVM_HOME=$HOME/tvm
% export PYTHONPATH=$TVM_HOME/python:$TVM_HOME/topi/python:$TVM_HOME/nnvm/python:${PYTHONPATH}
```

で，利用の準備が完了.
エミュレーション環境の設定を

```
% emconfigutil --platform ${AWS_PLATFORM} --nd 1
% sudo cp emconfig.json $(dirname $(which python))
```

build.py と run.py を用意して,

```
% export XCL_EMULATION_MODE=1
% export XCL_TARGET=sw_emu
% python3 build.py
```

と実行すると

```
TypeError: string argument without an encoding
```

とエラーが.
$TVM_HOME/python/tvm/contrib/sdaccel.py の

```
out_file.write(bytes(code))
```

を

```
out_file.write(bytes(code, 'UTF-8'))
```

に変更して,

```
python3 build.py
```

myadd.so とかができるので,

```
export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
python3 run.py
```

で実行.

```
[centos@ip-172-31-23-90 tvm-test]$ python3 run.py
ERROR: xclProbe-scan failed at fpga_pci_get_all_slot_specs
xclProbe found 0 FPGA slots with xocl driver running
ERROR: [SDx-EM 08] Please set XCL_EMULATION_MODE to "hw_emu" to run hardware emulation.
ERROR: [SDx-EM 09] Please set XCL_EMULATION_MODE to "sw_emu" to run software emulation.
ERROR: No devices found
[03:43:37] /home/centos/tvm/src/runtime/opencl/opencl_device_api.cc:263: No OpenCL platform matched
given existing options ...
[03:43:37] /home/centos/tvm/src/runtime/opencl/opencl_device_api.cc:263: No OpenCL platform matched
given existing options ...
Traceback (most recent call last):
  File "run.py", line 17, in <module>
    a = tvm.nd.array(np.random.uniform(size=n).astype("float32"), ctx)
  File "/home/centos/tvm/python/tvm/ndarray.py", line 214, in array
    return empty(arr.shape, arr.dtype, ctx).copyfrom(arr)
  File "/home/centos/tvm/python/tvm/_ffi/ndarray.py", line 132, in empty
```

```
    ctypes.byref(handle)))
  File "/home/centos/tvm/python/tvm/_ffi/base.py", line 314, in check_call
    raise get_last_ffi_error()
tvm._ffi.base.TVMError: Traceback (most recent call last):
  [bt] (2) /home/centos/tvm/build/libtvm.so(TVMArrayAlloc+0x9c) [0x7f743f29588c]
      [bt]    (1 )    /home/centos/tvm/build/libtvm.so(tvm::runtime::NDArray::Empty(std::vector<long,
std::allocator<long> >, DLDataType, DLContext)+0x1b8) [0x7f743f295758]
  [bt] (0) /home/centos/tvm/build/libtvm.so(+0xec74c6) [0x7f743f2e74c6]
  File "/home/centos/tvm/src/runtime/opencl/opencl_device_api.cc", line 123
TVMError: Check failed: context != nullptr: No OpenCL device
```

といわれる．ERROR の通り，

```
export XCL_EMULATION_MODE=sw_emu
```

として実行.

```
[centos@ip-172-31-23-90 tvm-test]$ python3 run.py
ERROR: xclProbe-scan failed at fpga_pci_get_all_slot_specs
xclProbe found 0 FPGA slots with xocl driver running
ERROR: device::load_binary binary target=Bin, no Hw HAL handle
Traceback (most recent call last):
  File "run.py", line 21, in <module>
    fadd(a, b, c)
  File "/home/centos/tvm/python/tvm/_ffi/function.py", line 153, in __call__
    return f(*args)
  File "/home/centos/tvm/python/tvm/_ffi/_ctypes/function.py", line 209, in __call__
    raise get_last_ffi_error()
tvm._ffi.base.TVMError: Traceback (most recent call last):
  [bt] (4) /home/centos/tvm/build/libtvm.so(TVMFuncCall+0x46) [0x7f8d3998ac76]
  [bt] (3) /home/centos/tvm/build/libtvm.so(+0xed1998) [0x7f8d399fb998]
  [bt] (2) /home/centos/tvm/build/libtvm.so(+0xed159a) [0x7f8d399fb59a]
  [bt] (1) /home/centos/tvm/build/libtvm.so(+0xecd99f) [0x7f8d399f799f]
  [bt] (0) /home/centos/tvm/build/libtvm.so(+0x722392) [0x7f8d3924c392]
  File "/home/centos/tvm/src/runtime/opencl/opencl_module.cc", line 219
  File "/home/centos/tvm/src/runtime/module_util.cc", line 73
TVMError: Check failed: ret == 0 (-1 vs. 0) : Check failed: err == CL_SUCCESS: OpenCL Error,
code=-44: CL_INVALID_PROGRAM
[centos@ip-172-31-23-90 tvm-test]$
```

FPGA ささってるマシンじゃないとだめな想定のようにみえる．とりあえず，合成だけでもしておく．

```
% unset XCL_EMULATION_MODE
% export XCL_TARGET=hw
% python3 build.py
```

とすると，最後に同様のエラーはでるけど，xclbin の合成はできた．おわったら

```
% $SDACCEL_DIR/tools/create_sdaccel_afi.sh \
  -xclbin=myadd.xclbin \
  -o=myadd \
  -s3_bucket=[ バケット名 ] \
  -s3_dcp_key=[DCP 保存フォルダ名 ] \
  -s3_logs_key=[ ログ保存フォルダ名 ]
```

で，AWS-F1 用のイメージ作成処理をキック

```
% cat *_afi_id.txt
```

で FpgaImageId を確認して，

```
% aws ec2 describe-fpga-images --fpga-image-ids [FpgaImageId]
```

で，State が pending から available になったら完了.

AWS-F1 インスタンスでトライ

AWS-F1 インスタンスを起動して，tvm，llvm などの一切合切を c4 インスタンスからコピーして

```
% sudo -s
% source $AWS_FPGA_REPO_DIR/sdaccel_setup.sh
% export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
% export TVM_HOME=/home/centos/tvm
% export PYTHONPATH=$TVM_HOME/python:$TVM_HOME/topi/python:$TVM_HOME/nnvm/python:${PYTHONPATH}
```

と準備.

```
% export XCL_EMULATION_MODE=sw_emu
% export XCL_TARGET=sw_emu
% python3 build.py
% python3 run.py
```

とすると,

```
[root@ip-172-31-62-53 tvm-test]# python3 run.py
xclProbe found 1 FPGA slots with xocl driver running
[0.5068266  0.1325183  0.9167701  ... 0.46502367 0.02036605 0.5523464 ]
[0.6834595  0.8389502  0.16160455 ... 0.9921764  0.5801108  0.86317337]
[0. 0. 0. ... 0. 0. 0.]
[1.1902862  0.9714685  1.0783746  ... 1.4572     0.60047686 1.4155197 ]
[root@ip-172-31-62-53 tvm-test]#
```

と計算できた (出力用に print を適当に追加した)

```
% export XCL_EMULATION_MODE=hw_emu
% export XCL_TARGET=hw_emu
% python3 build.py
% python3 run.py
```

では,

```
[root@ip-172-31-62-53 tvm-test]# python3 run.py
xclProbe found 1 FPGA slots with xocl driver running
[0.64612037 0.24518912 0.6705971  ... 0.75197536 0.02399846 0.12009709]
[0.8558938  0.9514521  0.5152762  ... 0.3747665  0.5249482  0.58834535]
[0. 0. 0. ... 0. 0. 0.]
INFO: [SDx-EM 01] Hardware emulation runs simulation underneath. Using a large data set will result in long simulation times. It is recommended that a small dataset is used for faster execution. This flow does not use cycle accurate models and hence the performance data generated is approximate.
[1.5020142  1.1966412  1.1858733  ... 1.1267419  0.5489466  0.70844245]
INFO: [SDx-EM 22] [Wall clock time: 04:39, Emulation time: 0.0579385 ms] Data transfer between kernel(s) and global memory(s)
myadd_kernel0_1:m_axi_gmem-DDR          RD = 8.000 KB               WR = 4.000 KB
```

と計算できたみたい.
FPGA では，と,

```
% unset XCL_EMULATION_MODE
% export XCL_TARGET=hw
% python3 run.py
```

とやってみた.

```
[root@ip-172-31-62-53 tvm-test]# python3 run.py
xclProbe found 1 FPGA slots with xocl driver running
xclAllocBO ERROR: AllocBO IOCTL failed
ERROR: std::bad_alloc
ERROR: Operation failed due to earlier error 'std::bad_alloc'
Traceback (most recent call last):
  File "run.py", line 18, in <module>
    b = tvm.nd.array(np.random.uniform(size=n).astype("float32"), ctx)
  File "/home/centos/tvm/python/tvm/ndarray.py", line 214, in array
    return empty(arr.shape, arr.dtype, ctx).copyfrom(arr)
  File "/home/centos/tvm/python/tvm/_ffi/ndarray.py", line 254, in copyfrom
    check_call(_LIB.TVMArrayCopyFromBytes(self.handle, data, nbytes))
  File "/home/centos/tvm/python/tvm/_ffi/base.py", line 314, in check_call
    raise get_last_ffi_error()
tvm._ffi.base.TVMError: Traceback (most recent call last):
  [bt] (2) /home/centos/tvm/build/libtvm.so(TVMArrayCopyFromBytes+0x768) [0x7fe99b3b9368]
  [bt] (1) /home/centos/tvm/build/libtvm.so(+0xecb6f2) [0x7fe99b4106f2]
  [bt] (0) /home/centos/tvm/build/libtvm.so(+0x722392) [0x7fe99ac67392]
  File "/home/centos/tvm/src/runtime/opencl/opencl_device_api.cc", line 171
TVMError: Check failed: e == CL_SUCCESS: OpenCL Error, code=-5: CL_OUT_OF_RESOURCES
```

で，固まってしまった.
ためしに，F1 インスタンスでも

```
% python3 build.py
% python3 run.py
```

としてみたが，かわらず．残念.