PyNET 실행 가이드

1. 개요

1.1 실행 가이드 소개

본 문서는 PyTorch PyNET의 C++포팅 소스코드를 실행하는 방법을 설명함.

- 참조한 원본 PyTorch 소스코드

> https://github.com/aiff22/PyNET-PyTorch

1.2 사전 필요사항

문서에서 소개하고 있는 소스코드를 실행하기 위해서는 다음과 같은 사전 준비사항이 필요함.

1.2.1 실행환경

프로그램을 실행하기 위한 최소 사항은 다음과 같으며, 아래와 같은 환경에서 테스트되었음.

1. OS
    a. Ubuntu 20.04LTS (빌드환경), Android API 29 (Android 10) 이상 OS

b. Android는 Google 에서 제공하는 NDK (Native Developement Kit)내의 Clang++ 컴파일러로 크로스 컴파일함. 이 과정에서 cmake 툴 사용.

2. CPU
   a. CPU버전을 실행하시 위해서는 64bit CPU가 필요함.
   b. 싱글코어 CPU에서도 동작은 가능하나 많은 연산이 필요함으로 프로그램의 동작에 많은 시간이 소모됨.
3. Memory
   a. CPU코드를 실행하기 위해 필요한 메모리 : 6.4 GB 이상
   b. GPU코드를 실행하기 위해 필요한 메모리 : 4.5 GB 이상
      i. 임베디드 기기의 경우 CPU와 GPU가 메모리를 공유하는 통합 메모리 구조를 가지고 있는 경우가 많음.
      ii. 통합 메모리 구조의 경우 기기의 제조사의 정책에 따라서 GPU에서 사용가능한 가용메모리의 크기가 다를 수 있음에 주의.
      iii. 대부분의 경우 시스템에서는 전체 메모리의 절반까지 GPU에서 사용 가능하다 표기되지만 실제로는 70~80% 정도만 사용 가능한 경우가 있어서 확인이 필요함.
4. GPU
   a. OpenCL 및 OpenCL에서 half 데이터 타입을 지원하는 GPU가 필요.
      i. Nvidia의 모바일 GPU(Jetson 라인업)의 경우 OpenCL을 미지원.
   b. OpenCL에서 local work-group이 256이상의 크기를 지원해야 함.
   c. OpenCL에서 MAX_MEM_ALLOC_SIZE가 1,073,741,824 (1GB) 이상의 크기를 지원해야 함.
   d. OpenCL에서 Max size for 1D images from buffer가 134217728 이상의 pixel을 지원해야 함.

OS	CPU	GPU	기타
Android 11	Qualcomm Snapdragon 888	Adreno 660	* Qualcomm SD888 devkit

> https://developer.qualcomm.com/hardware/snapdragon-888-hdk

1.2.2 소스코드 구조

- CPU
  - androidBuild : android 빌드를 위한 스크립트
  - include : CPU 버전 소스코드의 헤더파일
  - src : CPU 버전 소스코드
- GPU
  - androidBuild : android 빌드를 위한 스크립트
  - include : GPU 버전 소스코드의 헤더파일
  - src : GPU 버전 소스코드
- lib : 외부 라이브러리(OpenCL, libpng, libzip) 폴더
  - ARM64 : ARM64 버전의 라이브러리
- raw_images : 테스트를 위한 raw 형태의 이미지 (.png 확장자)가 저장됨
  - huawei_full_resolution : raw 형태의 이미지, 일반적인 이미지 뷰어로는 검은색 이미지만 출력됨
- weights
  - weights.bin : pynet 모델의 가중치가 바이너리 형태로 저장되어 있음. fp16 데이터 타입
- utils
  - clinfo : 컴파일된 arm64 android 바이너리, android 기기에서 실행 시, 지원하는 OpenCL 스펙을 확인가능함
- push.sh : Android 기기에 필요한 파일을 전송하는 스크립트

2. 소스코드의 실행

2.1 프로그램의 빌드

- 본 소스코드는 cmake와 NDK의 clang++ 크로스 컴파일러를 사용하여 빌드해야 함.
- 빌드하려는 대상이 CPU인지 GPU인지에 따라서 1.2.2에서 소개된 소스코드 폴더 외 진행 방법은 동일.

2.1.1 AMD64환경에서 Android ARM64 크로스 컴파일

- Ubuntu를 사용하는 경우 다음과 같은 명령어로 기본 패키지들을 설치할 수 있음.

```
sudo apt install build-essential cmake
```

- AMD64 환경에서 Android 소스코드를 빌드하기 위해서는 Google에서 제공하는 NDK (Native Developement Kit)내 크로스 컴파일러를 사용하여야 함.
- 아래 링크에서 빌드 환경에 맞는 NDK를 다운받을 수 있으며, 64비트 Linux 환경에서 r23c LTS 버전을 활용하여 동작을 확인함.

> https://developer.android.com/ndk/downloads?hl=ko

- 빌드 환경에서 android 기기로 실행 바이너리를 전송 시 Google에서 제공하는 ADB (Android Debug Bridge) 프로그램이 필요함.

> https://developer.android.com/studio/releases/platform-tools?hl=ko

- androidBuild 폴더에 존재하는 build.sh 스크립트 내에서 NDK를 다운받아 압축을 풀은 위치를 지정.

```
cd gpu/androidBuild/ # 또는 ./cpu/androidBuild
vim build.sh

# 아래 라인을 환경에 맞게 수정
# export NDK=해당_빌드_환경에서_NDK의_위치
```

- androidBuild폴더에서 비어있는 새로운 build 빈 폴더 생성 후 build.sh 스크립트를 새로 생성한 폴더로 복사.

```
# gpu/androidBuild 또는 cpu/androidBuild 폴더에서
mkdir build
mv build.sh ./build/
```

- 생성된 build 폴더에서 build.sh 스크립트를 실행.

```
cd build
chmod +x build.sh
```

```
./build.sh
```

- make 명령어로 프로그램 빌드.

```
# gpu/androidBuild/build 또는 cpu/androidBuild/build 폴더에서
make -j
```

2.2 프로그램의 실행

2.1의 과정을 마치고 프로그램 바이너리가 빌드완료된 상황을 가정함.

2.2.1 Android 환경에서 실행

- adb를 사용하여 빌드 결과물을 android 기기에 전송하는 과정이 필요함. (android 기기 쪽에서도 adb를 허용하는 과정이 필요함)

> https://developer.android.com/studio/command-line/adb?hl=ko

- 이 때 android의 경우 Google의 보안 정책으로 root 권한이 없는 경우 /data/local/tmp 의 위치를 사용하는 것 이 일반적임.
- 동봉된 push.sh 스크립트를 실행 시 자동으로 adb로 연결된 기기의 /data/local/tmp 폴더에 필요한 폴더를 생성 후 파일을 전송하는 과정까지 진행함.

```
# 소스파일 루트폴더에서
./push.sh
```

- android 기기에 shell로 접속하여 2.1.2과정에서 빌드한 바이너리 파일의 위치로 이동.

```
adb shell
# 이후 adb를 사용하여 android 기기 shell로 접속함
# android의 shell은 $로 표시

$ cd /data/local/tmp/pynet/gpu/build/Debug
```

- android 기기에서 라이브러리의 위치를 LD_LIBRARY_PATH로 지정.

```
# android 기기의 shell에서
$ export LD_LIBRARY_PATH=/data/local/tmp/pynet/lib
```

- 바이너리 실행.
    - ./pynet-0.1.out raw_이미지폴더_위치 가중치_위치 추론할_이미지_갯수

```
# /data/local/tmp/pynet/gpu/build/Debug 에서
$ ./pynet-0.1.out ../../../raw_images ../../../weights/weights.bin 1
```