Intel Neural Compute Stick 2

1. Opis

Intel Neural Compute Stick 2 to stosunkowo tanie urządzenie, którego przeznaczeniem jest akceleracja działania sieci neuronowych, głównie w porównaniu z procesorami CPU. W środku tego urządzenia znajdziemy procesor Intel Movidius Myriad X Vision Processing Unit (VPU). Wykonany w 16-nanometrowej litografii charakteryzuje się bazowym taktowaniem 700 MHz i składa się z 16 programowalnych tzw. “shave cores” i “neural compute engine” do sprzętowej akceleracji sieci neuronowej (deep neural network). Urządzenie może wykonywać obliczenia liczbach zmiennoprzecinkowych FP32 (IEEE754) i FP16 (IEEE754-2008 lub tzw. bfloat16).
Urządzenie jest przenośne i może być podłączone do prawie wszystkich systemów komputerowych, które mają port USB2.0/3.0. o ile wspierają instalację OpenVINO Toolkit (większość systemów Linux oraz Windows). Może być więc wykorzystywane w zwykłym komputerze stacjonarnym, laptopie, raspberry pi, itp.

2. Instalacja OpenVINO Toolkit

Żeby zainstalować oraz przetestować OpenVINO Toolkit dla systemu Windows posługiwaliśmy się instrukcją podaną na głównej stronie tego narzędzia Windows Installation Instruction. Instalacja Visual Studio jest potrzebna tylko w przypadku chęci uruchomienia przykładów z SDK.

3. Konwersja modeli Keras -> TensorFlow.

Żeby wykorzystać stworzony przez nas model do inferencji na urządzeniu Intel Neural Compute Stick 2 należy “zamrozić” model oraz przekonwertować go do odpowiedniego formatu. Dla tego celu przygotowaliśmy skrypt (**detector31.py)** który wczytuje odpowiedni plik z modelem (w tym wypadku person_detector.h5) i konwertuje do formatu .pb.

4. Optymalizacja modelu

Przed użyciem modelu należy wykonać jego optymalizację oraz przekształcić w format IR. Dla tego celu skorzystaliśmy z gotowego skryptu, który jest udostępniony przez OpenVINO jako *mo_tf.py*. Przydatne parametry tutaj to typ danych (--data_type) oraz rozmiar wejścia (--input_shape) w którym można też podać batch size.

5. Test inferencji

Przygotowałem skrypt pythonowy do inferencji *inference.py* oraz przetestowałem na kilku obrazkach czy sieć daje dobre wyniki.

6. Test wydajności urządzenia

Na początek przetestowaliśmy urządzenie korzystając z poprzedniego skryptu i zmierzyliśmy ile klatek na sekundę jest w stanie przetworzyć urządzenie.

	batch=1	**batch=2**	**batch=4**	**batch=8**
USB2.0	8.67	7.86	8.99	8.98
USB3.0	12.08	13.92	14.04	14.01

Tablica 1. Synchroniczne wywołanie z FP32

	batch=1	**batch=2**	**batch=4**	**batch=8**
USB2.0	8.62	8.72	8.85	8.87
USB3.0	11.64	12.26	12.33	14.08

Tablica 2. Synchroniczne wywołanie z FP16

W drugim kroku skorzystaliśmy z programu *benchmark_app.py,* który pozwala zmierzyć wydajność modeli wykorzystując asynchroniczne API. Pozwala to na równoległe zlecenie większej ilość żądań inferencji - parametr *ninfer*. Punkt odniesienia to processor Intel i5-7200U 2.5GHz, który osiąga wydajność około 15 fps (frames per second).

USB	**ninfer**	**batch=1**	**batch=2**	**batch=4**	**batch=8**
USB2.0	1	11.20	11.62	11.69	11.61
USB3.0	1	14.75	14.95	15.07	15.12
USB2.0	2	20.37	21.38	21.22	21.00
USB3.0	2	27.20	27.41	27.81	27.69
USB2.0	4	24.97	24.97	24.74	25.19
USB3.0	4	28.18	28.41	28.49	28.05
USB2.0	8	24.89	25.08	24.95	24.92
USB3.0	8	27.86	28.11	27.66	24.05

Tablica 3. Asynchroniczne wywołanie z FP32

USB	ninfer	batch=1	batch=2	batch=4	batch=8
USB2.0	1	11.10	11.39	11.56	11.63
USB3.0	1	14.54	14.25	14.37	14.41
USB2.0	2	20.20	21.00	21.20	21.07
USB3.0	2	25.64	26.00	26.63	26.63
USB2.0	4	24.93	24.92	25.13	25.22
USB3.0	4	25.81	24.83	24.73	24.45
USB2.0	8	24.63	24.97	25.00	24.92
USB3.0	8	23.58	22.50	22.56	23.17

Tablica 4. Asynchroniczne wywołanie z FP16

7. Wnioski

Z powyższych tabel wynika, że wykorzystując urządzenie Intel Neural Compute Stick 2 możemy oczekiwać wydajności na poziomie około 25 klatek na sekundę.