2018 Third ACM/IEEE Symposium on Edge Computing

Embedded Deep Learning for Vehicular Edge Computing

Jacob Hochstetler*, Rahul Padidela*, Qi Chen*, Qing Yang† and Song Fu†
*{JacobHochstetler, RahulPadidela, QiChen}@my.unt.edu †{Qing.Yang, Song.Fu}@unt.edu
Department of Computer Science and Engineering
University of North Texas
Denton, TX, USA

***Abstract*—The accuracy of object recognition has been greatly improved due to the rapid development of deep learning, but the deep learning generally requires a lot of training data and the training process is very slow and complex. In this work, an Intel Movidius™ Neural Compute Stick along with Raspberry Pi 3 Model B is used to analyze the objects in the real time images and videos for vehicular edge computing. The results shown in this study tells how the stick performs in conjunction with different operating systems and processing power.**

***Index Terms*—Edge computing, Deep learning, Embedded computing, Object detection, Vehicular system.**

I. Introduction

The need for low-power but capable inference processors has produced a new class of computing called edge, or sometimes ”fog” computing. These stand-alone, specialized edge-computing devices has become more and more popular for three main reasons: lower network delay, energy efficiency, and better privacy protection. The promise of edge computing is that by processing data at the network edge would result in shorter response time, more efficient processing, and less congestion on the network [9].

Connected and Autonomous Vehicles (CAV) are a new class of vehicles that can both communicate with each other or infrastructure (CV) and employ a number of systems that enable automated driving functions (AV). The Society of Automotive Engineers (SAE) has created a six-level hierarchy to categorize AV's capabilities [1], ranging from no capability at all to full automation with no driver engagement.

The CV's communication capability is also categorized as Vehicle to Infrastructure (V2I), Vehicle to Vehicle (V2V), Vehicle to Cloud (V2C), Vehicle to Pedestrian (V2P) or the all-encompassing Vehicle to Everything (V2X).

Artificial neural networks (ANNs), or connectionist systems, are computing systems vaguely inspired by the biological neural networks (NN) that constitute animal brains. Such systems ”learn” (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cars by analyzing training images that have been manually labeled as ”car” or ”not car” and use those results to identify cars in future, unseen images. This was demonstrated in the TV show *Silicon Valley* by a character's SeeFood mobile app, which determined through an ANN if photos contained ”hotdog or not hotdog” [6].

This work presents benchmarks from deep learning networks running on an edge computing node assisted by a Vision Processing Unit (VPU). Our results show that the a mobile edge device assisted by a VPU is able to process video using a popular NN in real-time. This is important for a CAV, since it leaves the main CPU and memory free for V2X communication or other tasks.

II. Mobile Edge Computing Setup

A. Mobile Edge Device

The Raspberry Pi (RPi) is a small, single-board computer (SBC) [7]. Each RPi is based around a Broadcom system on a chip (SoC), which contains the ARM CPU, RAM, GPU and general purpose input/output controllers (GPIO). Originally intended to just be used in teaching computer science and basic embedded computing, but through three different models and several iterations, has become widely successful in home automation, industrial (edge) computing and packaged commercial products.

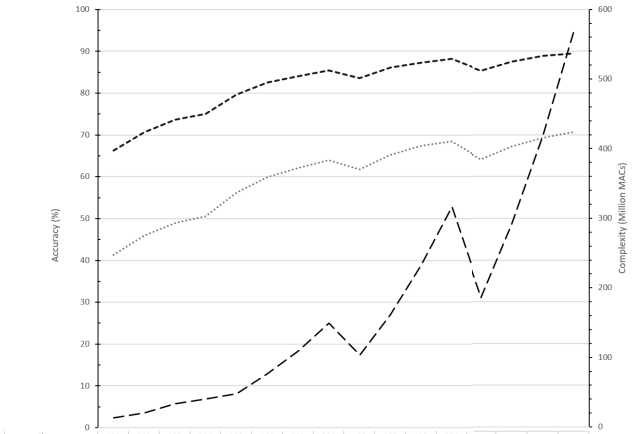
The RPi3B used in our experiments contains a 1.2 GHz 64-bit quad-core Cortex-A53 (ARMv8) CPU, 1 GB low-power DDR2 SDRAM and four USB 2.0 ports via on-board 5-port USB hub. It draws a maximum of 6.7 W at peak load.

B. Embedded Deep Learning Device

The Intel® Movidius™ [4] Neural Compute Stick (NCS) is a tiny fanless, USB 3.0 Type-A deep learning device that can be used to learn AI programming at the edge. NCS is powered by the same low-power, high-performance Myriad 2 Vision Processing VPU) that can be found in smart security cameras, gesture-controlled drones, industrial machine vision equipment, and other embedded systems. Ubuntu 16.04 is supported installed on a physical x86_64 system, or Debian Stretch running on a Raspberry Pi 3 Model B. The Neural Compute SDK comes with a C++ and Python (2.7/3.5) API [5].

The Myriad 2 VPU within the NCS produces almost 100 GFLOPS using only 1 W of power and between 10 to 15 inferences per second. The VPU includes 4 Gb of low-power DDR3 DRAM, imaging and vision accelerators, and an array of 12 very long instruction word (VLIW) vector processors called SHAVE processors. The entire NCS draws 2.5 W of peak power through its USB 3.0 port. A trained Caffe-based or TensorFlow™ CNN is compiled into an embedded neural network that is optimized to run on the VPU inside the NCS.

978-1-5386-9445-9/18/$31.00 ©2018 IEEE
DOI 10.1109/SEC.2018.00038

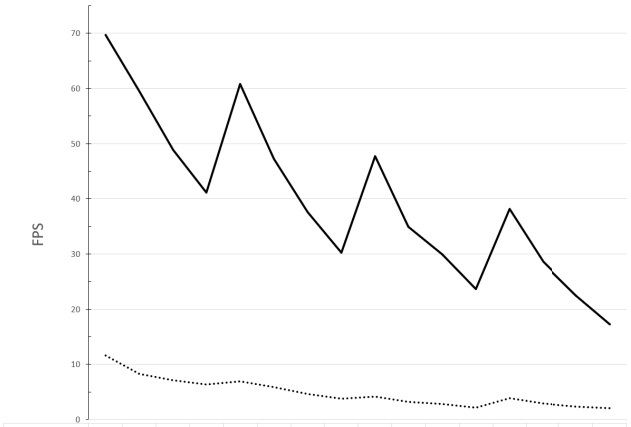


Size	128	160	192	224	128	160	192	224	128	160	192	224	128	160	192	224
Depth	0.25	0.25	0.25	0.25	0.50	0.50	0.50	0.50	0.75	0.75	0.75	0.75	1.00	1.00	1.00	1.00
Top-1 Accuracy	41.3	46	49	50.6	56.2	59.9	62.1	64	61.8	65.2	67.4	68.4	64.1	67.2	69.3	70.7
Top-5 Accuracy	66.2	70.7	73.6	75	79.6	82.5	84	85.4	83.6	86.1	87.3	88.2	85.3	87.5	88.9	89.5
Million MACs	14	21	34	41	49	77	110	150	104	162	233	317	186	291	418	569

Fig. 1. Google's MobileNets Accuracy and Complexity

III. Experimental Results

Google's MobileNets [3] is a Convolutional Neural Network (CNN) that uses depth-wise separable convolutions to build light weight deep neural networks. MobileNets is tunable by two hyper-parameters (i.e., size and depth), so that a less powerful embedded system can trade accuracy for model latency (i.e., time to result). The Top-1 and Top-5 accuracy in Fig. 1 is measured against the Large Scale Visual Recognition Challenge [8]. The results from benchmarking MobileNets on a bare RPi3B and assisted with an NCS are shown in Fig 2.



Size	128	160	192	224	128	160	192	224	128	160	192	224	128	160	192	224
Depth	0.25	0.25	0.25	0.25	0.50	0.50	0.50	0.50	0.75	0.75	0.75	0.75	1.00	1.00	1.00	1.00
Pi3	11.60	8.30	7.10	6.40	6.90	5.90	4.60	3.80	4.20	3.20	2.80	2.20	3.90	2.90	2.40	2.10
Pi3 w/NCS	69.71	59.52	48.89	41.15	60.77	47.22	37.61	30.27	47.77	34.94	29.95	23.61	38.20	28.61	22.53	17.21
NCS Speedup	601%	717%	689%	643%	881%	800%	818%	797%	1137%	1092%	1070%	1073%	979%	987%	939%	820%

Fig. 2. MobileNets RPi3B (FPS): CPU only vs. 1 x NCS

The next two experiments, depicted in Fig 3, compare a RPi3B to a standalone Ubuntu Desktop. The Ubuntu 16.04 LTS system used for comparison is powered by a 3.06 GHz dual-core Intel® E7600 CPU, 4 GB of DDR3/1066 MHz RAM. The Linux kernel used during testing was 4.15.0-23.

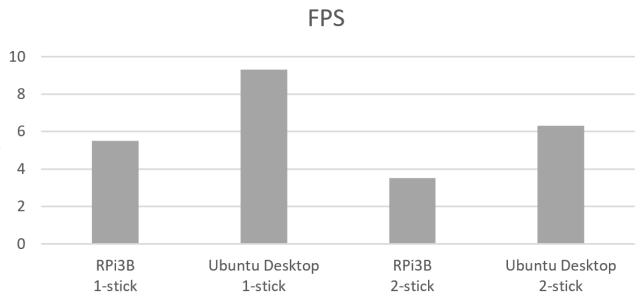


Fig. 3. Frames per second (FPS) platform comparison with 1 or 2-sticks

The first experiment used `video_objects.py` from the Movidius™ `ncappzoo/apps`, along with the six example videos used for testing and training. The desktop achieved 9.3 frames-per-second (FPS) with a single NCS attached while the RPi3B produced 5.7 FPS. The second experiment used the GoogLeNet [10] CNN classifier `street_cam.py` with example videos. With two sticks the desktop produced 6.6 FPS, while the RPi3B produced 3.5 FPS.

IV. Discussion

In addition to a lower clock speed, the RPi3B only has a 60 MBps USB 2.0 bus, which is ten times slower than the theoretical maximum of a USB 3.0 bus. This greatly limits the speed that data can flow into the NCS from the host RPi3B.

KITTI is one of the main open resources for training CAV models and all the examples are synchronized at 10 Hz (cameras, lidar, and GPS) [2]. The bare RPi3B is not able to keep up with real-time processing when the depth and size increases in MobileNets. In contrast, a single NCS can process a single 10 Hz feed in real-time at the largest size and depth, producing the most accurate results. Since each NCS is uniquely indexed, different feeds can be sent to different sticks for independent processing, or independent networks can be loaded on each stick for processing. Lab tests at Movidius™ show linear performance increases up to 4 sticks, with validation pending for 6 to 8-stick configurations.

Neither the desktop, nor the RPi3B were able to process `video_objects.py` or `street_cam.py` in real-time, which means that the models used could be further optimized to produce better results on either platform.

V. Conclusions

The results show that the RPi3B is capable of processing real time video and recognizing objects using the embedded deep learning device. Multiple sensor feeds, which are prevalent in CAVs, can be processed independently by different sticks. This frees the edge computing device's CPU and memory for other tasks, like real-time diagnostics, V2X communication or Advanced Driver-Assistant Systems.

REFERENCES

[1] James M Anderson, Kalra Nidhi, Karlyn D Stanley, Paul Sorensen, Constantine Samaras, and Oluwatobi A Oluwatola. *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[2] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[3] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[4] Intel® Corporation. Movidius™ neural compute stick product page. https://developer.movidius.com/. Accessed: 2018-06-30.

[5] Intel® Corporation. Movidius™ Neural Compute SDK (ncsdk). https://movidius.github.io/ncsdk/, 7 2018. version V2.05.00.

[6] Meghan Pleticha. Silicon Valley. *Teambuilding Exercise*, Season 4 (Episode 4), May 2017.

[7] Raspberry Pi Foundation™. Raspberry pi 3 model b product page. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. Accessed: 2018-06-30.

[8] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[9] W. Shi and S. Dustdar. The Promise of Edge Computing. *Computer*, 49(5):78–81, May 2016.

[10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. *CoRR*, abs/1409.4842, 2014.