ISSN 0002-306X. Proc. of the RA NAS and NPUA Ser. of tech. sc. 2022. V. LXXV, N1

UDC 004.832

COMPUTER SCIENCE AND INFORMATICS

DOI: 10.53297/0002306X-2022.v75.1-72

T.B. KHACHATRYAN, D.F. DAVTYAN

DEPTH ESTIMATION AI INFERENCING COMPARISON OF JETSON XAVIER NX AND CORAL DEV BOARD

AI inferencing, especially the real time processing of neural networks may require excessive calculation power in terms of speed and memory, thus opening a big area for research and design of new devices intended for AI acceleration. Examples of such devices are Nvidia's Jetson series, Google's Coral. Also, there are FPGA solutions such as Xilinx's AI applicable FPGA. Depending on application requirements it could be difficult to choose between these devices, as in most cases, speed and accuracy are the most important factors, while there are also applications which require low power and low cost. Thus, investigation and comparison of these inferencing devices in terms of speed, memory, power and cost for the chosen field of subject may be useful for choosing the right device for the given task. We have done such an analysis for the depth estimation task using Jetson Xavier NX and Coral Dev Board as inferencing devices.

Keywords: a artificial intelligence, neural network, depth estimation, Jetson Xavier NX, Coral Dev Board.

Introduction. In this work, we are analyzing the inference of depth estimation deep learning model on AI inferencing devices to find out their advantages and disadvantages for the specified task. We have chosen depth estimation from computer vision subtasks, as it is more challenging and comparably less investigated in literature. Also, the possible opportunity of replacing the traditional methods of depth estimation increases the interest of this analysis, because the Photogrammetry which is the main tool for depth estimation nowadays requires heavy 3D reconstruction processing, that is only suitable on stationary processing units. Meanwhile, there are applications that require instantaneous information on scene properties, such as UAVs obstacle avoidance, indoor navigation, self-driving vehicles, etc. [1]. We have used aerial footage for training and testing our models as it is the hardest in terms of picture properties due to long shooting distances, meaning low resolution on objects.

Many authors solved the depth estimation problem, using supervised neural networks [2-4]. Although, supervised methods provide high accuracy, they require large datasets with accurate ground truth depth maps, thus reducing their applicability. As an alternative, self-supervised methods can be applied, which do not require any

ground truth depths and provide enough accuracy. There are two main methods of self-supervised depth estimation: through stereo pairs [5, 6] and through monocular videos [7, 8]. We have used monocular videos in this work.

The inferencing devices were compared in terms of the output image depth quality, speed, required memory, power usage. We have chosen Jetson Xavier NX and Google Coral as inferencing devices, as they have one of the best price-performance ratios in the market and have not been sufficiently investigated in literature.

The chosen model. This section briefly describes the chosen model for depth estimation and also presents the inferencing hardware and the steps for preparing models to inference.

For depth estimation we have used a convolutional neural network which is similar to the network presented by [9] with some small architectural changes. One of such changes is the replacement of Exponential Linear Units (ELU) with Parametric Rectified Linear Units (PReLU) in the decoder part. This is necessary, because Edge TPU Compiler [10] does not support ELU operation.

Jetson Xavier NX specifications. Jetson Xavier NX is a power-efficient, compact module for AI edge devices. It accelerates the NVIDIA software stack in as little as 10 W with more than 10x the performance of its widely adopted predecessor Jetson TX2. The brief properties of this device are described below.

GPU	NVIDIA Volta architecture with 384 NVIDIA CUDA cores and 48 Tensor cores
Memory	8 *GB* 128-bit LPDDR4x 51.2 *GB/s*
Video Encode	2x 4Kp30 | 6x 1080p 60 | 14x 1080p30

Xavier NX is capable of **21 TOPS** (int8) or **6 TFLOPS** (fp16) of AI performance while consuming only 15 watts of power. When limited to 10 watts, it can still perform at **14 TOPS**. More detailed evaluation can be found in Nvidia's site.

Coral Dev Board specifications. The Coral Dev Board is a single-board computer that's ideal for performing fast machine learning (ML) inferencing in a small form factor. The on-board Edge TPU coprocessor is capable of performing **4 trillion operations (tera-operations) per second (TOPS)**, using 0.5 watts for each TOPS (2 TOPS per watt).The brief properties of this device are described below.

CPU	NXP i.MX 8M SoC (quad Cortex-A53, Cortex-M4F)
GPU	Integrated GC7000 Lite Graphics
ML accelerator	Google Edge TPU coprocessor
Memory	4 *GB* LPDDR4

Inference. We have done inference on Jetson Xavier NX using Nvidia TensorRT Python API, CUDA Python API and Jetson-inference library. For generating the TensorRT engine which is required for the inference, first we have converted our PyTorch model to Open Neural Network Exchange (ONNX) format, and then we have generated TensorRT engine from ONNX model [11].

The PyCoral Python API and TFLite models have been used during the inference on Coral Dev Board [12]. For generating TFLite models, first we have converted our PyTorch model to Open Neural Network Exchange (ONNX) format, then we have converted ONNX format to OpenVINO, and finally the TFLite models have been created by converting OpenVINO model to Tensorflow and using TFLite converter. The ONNX to OpenVINO step was necessary, because PyTorch and Tensorflow use different data storing formats and it is difficult to directly convert between them.

We have created various engines with different input resolution and precision parameters and have performed extensive experiments which are described in the following section.

Experiments. In this section we introduce the dataset and the comparison of TensorRT and TFLite models' results.

Dataset. We have used China video sequences from the UAVid dataset [13] for creating training, validation and test sets. There are 34 video sequences with a resolution of 3840x2160 in China sub-dataset.

Like [14], for testing the model performance, we compared the depth maps generated by the model with reference depths. As reference depths the point clouds generated through Pix4D photogrammetric tool were used. A total of 412 images were obtained as reference depths through this process.

Evaluation metrics. To assess the performance, various pixel-wise metrics are calculated between the predicted and reference depths. The evaluation of the accuracy is done based on calculating several metrics between the single image depths (d') generated from the model and the reference depths (d) produced using Pix4D. The evaluation metrics are: Absolute Relative difference (Abs Rel) given in equation (1), Squared Relative difference (Sq Rel) given in equation (2), Root Mean Square Error (RMSE) given in equation (3), Root Mean Square Logarithmic Error (RMSE log) given in equation (4). The accuracy given in equation (5) is described in [9] and [14]:

$$Abs\ Rel = \frac{1}{N}\sum_{i=1}^{N}\frac{|d(x_i)-d'(x_i)|}{d(x_i)}, \quad (1)$$

$$SqRel = \frac{1}{N}\sum_{i=1}^{N}\frac{|d(x_i)-d'(x_i)|^2}{d(x_i)}, \quad (2)$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(d(x_i)-d'(x_i)\right)^2}, \quad (3)$$

$$RMSElog = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(log\left(d(x_i)\right)-log\left(d'(x_i)\right)\right)^2}, \quad (4)$$

$$Accuracy(\delta_\theta) = \frac{1}{N}\sum_{i=1}^{N} max\left(\frac{di}{d_i'},\frac{d_i'}{d_i}\right) < \theta. \quad (5)$$

Here the accuracy represents the fraction of pixels that are within a certain threshold θ to the corresponding pixel wise value in the reference depth map. The thresholds chosen are 1.05, 1.15, 1.25, $(1.25)^2$, $(1.25)^3$.

Comparison of results. We have performed the comparison of TensorRT and TFLite models in terms of accuracy, inference statistics, memory usage, power usage, model complexities. The evaluation of our models' accuracies was carried out by comparing the results with reference depths generated using the PIX4D software. We also present the comparison with the result published by [14] which is one of the best works that uses the UAVid dataset.

In Table 1 and Table 2 are shown the comparisons of model accuracies with input resolutions of 256x160 and 320x192 respectively.

From Table 1, we can see that decreasing precision from fp32 to fp16 does not significantly change the results. The results are slightly decreased when using int8 data type, but it gives more performance boost. The results of the model with int8 data type were measured after performing the calibration process. For that we have used calibration dataset, which includes more than 400 images from the original training dataset.

The difference between TFLite and TensorRT Int8 models' accuracies may be caused by different calibration processes of the TensorRT and TFLite converter.

From Table 2, we can see that using higher input resolution improves the results for both TensorRT and TFLite; however, it increases the model complexity.

Table 1

Results achieved over models with input resolution of 256x160

Model name	TensorRT model (256x160)			TFLite model (256x160)	Results from Madhuanand et al (2021)
Data type	Fp32	Fp16	Int8	Int8	Fp32
Absolute relative	0.269	0.269	0.27	0.29	0.109
Square relative	17.552	17.549	17.541	23.692	7.742
RMSE	52.463	52.479	52.352	63.314	48.303
RMSE log	0.293	0.293	0.293	0.316	-
a1 (<1.25)	0.649	0.649	0.649	0.642	0.878
a2 ($<1.25^2$)	0.859	0.859	0.86	0.836	-
a3 ($<1.25^3$)	0.936	0.937	0.936	0.92	-
a2* (<1.15)	0.487	0.487	0.485	0.466	0.761
a3* (<1.05)	0.192	0.191	0.193	0.175	0.327

Table 2

Results achieved over models with input resolution of 320x192

Model name	TensorRT model (320x192)			TFLite model (320x192)	Results from Madhuanand et al (2021)
Data type	Fp32	Fp16	Int8	Int8	Fp32
Absolute relative	0.255	0.255	0.259	0.234	0.109
Square relative	15.523	15.528	15.954	16.095	7.742
RMSE	51.523	51.552	52.132	55.275	48.303
RMSE log	0.279	0.279	0.282	0.262	-
a1 (<1.25)	0.65	0.65	0.643	0.709	0.878
a2 ($<1.25^2$)	0.874	0.874	0.869	0.902	-
a3 ($<1.25^3$)	0.945	0.945	0.944	0.956	-
a2* (<1.15)	0.489	0.489	0.484	0.517	0.761
a3* (<1.05)	0.2	0.2	0.196	0.204	0.327

Table 3 and Table 4 show the model complexities expressed in floating point operations (FLOPs) and memory usages of the models. As we can see, TFLite models on Coral are simpler and significantly superior TensorRT models in terms of the memory used. Table 3 shows that TensorRT models require less memory in case of fp16 and int8 data types.

Table 3

TensorRT model complexities and memory usages

Input resolution		256x160	320x192
Number of parameters		14.8 M	14.8 M
Model complexity	Fp32	0.85 GFLOPs	1.6 GFLOPs
	Fp16	0.6 GFLOPs	0.65 GFLOPs
	Int8	NA	NA
Memory usage	Fp32	4 *Gb*	4 *Gb*
	Fp16	3.3 *Gb*	3.3 *Gb*
	Int8	3.3 *Gb*	3.3 *Gb*

Table 4

TFLite model complexities and memory usages

Input resolution		256x160	320x192
Number of parameters		14.8 M	14.8 M
Model complexity	Int8	NA	NA
Memory usage	Int8	175 *Mb*	180 *Mb*

Comparisons of models' performances measured by the number of frames processed in a second (FPS) are given in Table 5 and Table 6. From these tables, we can see that decreasing the precision significantly improves FPS numbers of TensorRT models giving about 4-5 times higher results compared to TFLite models.

Table 5

Inference statistics (FPS) of various models with input resolution of 256x160

Video resolution	Input resolution	TensorRT model			TFLite model
		Fp32	Fp16	Int8	Int8
640x192	256x160	65 FPS	95 FPS	106 FPS	20 FPS
720x480	256x160	48 FPS	62 FPS	68 FPS	16 FPS
1280x720	256x160	31 FPS	38 FPS	39 FPS	11 FPS

Table 6

Inference statistics (FPS) of various models with input resolution of 320x192

Video resolution	Input resolution	TensorRT model			TFLite model
		Fp32	Fp16	Int8	Int8
640x192	320x192	44 FPS	75 FPS	85 FPS	16 FPS
720x480	320x192	30 FPS	55 FPS	60 FPS	14 FPS
1280x720	320x192	25 FPS	33 FPS	36 FPS	10 FPS

In Table 7 and Table 8 are shown the power usages of TensorRT models with input resolutions of 256x160 and 320x192 respectively. TFLite models consume less than 2000 *mW* on Coral Dev Board. We are not presenting detailed power consumption results for Coral Dev Board, as Google has not developed yet an accurate utility for power estimation. As we can see, Coral Dev Board is much more power-efficient than Jetson Xavier NX.

Table 7

Power usage of TensorRT models with input resolution of 256x160

Video resolution	Input resolution	TensorRT model			Idle power usage
		Fp32	Fp16	Int8	
640x192	256x160	10100 *mW*	7700 *mW*	6800 *mW*	3000 *mW*
720x480	256x160	8800 *mW*	6800 *mW*	6200 *mW*	3000 *mW*
1280x720	256x160	7800 *mW*	6400 *mW*	5900 *mW*	3000 *mW*

Table 8

Power usage of TensorRT models with input resolution of 320x192

Video resolution	Input resolution	TensorRT model			Idle power usage
		Fp32	Fp16	Int8	
640x192	320x192	11000 *mW*	8200 *mW*	7100 *mW*	3000 *mW*
720x480	320x192	9600 *mW*	7400 *mW*	6400 *mW*	3000 *mW*
1280x720	320x192	8400 *mW*	6600 *mW*	6000 *mW*	3000 *mW*

Conclusion. For the comparison of metrics evaluated in this work, it is obvious that Jetson Xavier NX outperforms Coral Edge TPU in terms of inference speed and computational power. Meanwhile, Coral Edge TPU is more energy and memory efficient. Adding to this Coral's low price, we can argue that it can be a good choice in applications where low power and low cost are essential. Although, both devices perform well, they still have to work on improving data transmission, as we can see both of them have difficulties with high resolution inputs.

For future work, we would like to explore the depth estimation inference on high resolution videos, as the experiments show, that increasing resolution can significantly improve the accuracy.

REFERENCES

1. **Nex F., Remondino F.** UAV for 3D mapping applications: A review // Appl. Geomatics. - 2014. - Vol. 6. - P. 1-15.
2. **Eigen D., Puhrsch C., Fergus R.** Depth Map Prediction from a Single Image using a Multi-Scale Deep Network // Advances in Neural Information Processing Systems. - 2014. - P. 2366-2375.
3. Deep ordinal regression network for monocular depth estimation / **H. Fu, M. Gong, C. Wang, et al //** CVPR. - 2018. - P. 2002-2011.
4. Deeper depth prediction with fully convolutional residual networks / **I. Laina, C. Rupprecht, V. Belagiannis, et al //** 3DV. - 2016. - P. 239-248.
5. **Garg R., Kumar BG V., Carneiro G., Reid I.** Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue // ECCV. - 2016. - P. 740-756.
6. **Godard C., Mac Aodha O., Brostow G. J.** Unsupervised Monocular Depth Estimation with Left-Right Consistency // Computer Vision and Pattern Recognition. - 2017. - P. 6602-6611.

7. **Zhou T., Brown M., Snavely N., Lowe D.** Unsupervised learning of depth and ego-motion from video // Computer Vision and Pattern Recognition. - 2017. - P. 6612-6619.
8. **Casser V., Pirk S., Mahjourian R., Angelova A.** Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos // AAAI. – 2019.
9. **Godard C., Aodha O.M., Firman M., Brostow G.** Digging into self-supervised monocular depth estimation // IEEE International Conference on Computer Vision. - 2019. - P. 3827-3837.
10. https://coral.ai/docs/edgetpu/compiler/.
11. https://docs.nvidia.com/deeplearning/tensorrt/developer-guide/index.html.
12. https://coral.ai/docs/reference/py/.
13. UAVid: A semantic segmentation dataset for UAV imagery / **Y. Lyu, G. Vosselman, G. Xia, et al** // ISPRS Journal of Photogrammetry and Remote Sensing. - 2020. - Vol. 165. - P. 108-119.
14. **Madhuanand L., Nex F., Yang M.Y.** Self-supervised monocular depth estimation from oblique UAV videos // ISPRS Journal of Photogrammetry and Remote Sensing. - 2021. - Vol. 176. - P. 1-14.

National Polytechnic University of Armenia. The material is received on 03.03.2022.

Տ.Բ. ԽԱՉԱՏՐՅԱՆ, Դ.Ֆ. ԴԱՎԹՅԱՆ

ԱՐՀԵՍՏԱԿԱՆ ԲԱՆԱԿԱՆՈՒԹԱՆ ՄԻՋՈՑՈՎ ՊԱՏԿԵՐԻ ԽՈՐՈՒԹՅԱՆ ԳՆԱՀԱՏՄԱՆ ՀԱՄԱՐ JETSON XAVIER NX-Ի ԵՎ CORAL DEV BOARD ՏՐԱՄԱԲԱՆԱԿԱՆ ԵԶՐԱԿԱՑՈՒԹՅԱՆ ՍԱՐՔԵՐԻ ՀԱՄԵՄԱՏՈՒԹՅՈՒՆԸ

Արհեստական բանականության (ԱԲ) տրամաբանական եզրակացությունը, մասնավորապես՝ նեյրոնային ցանցերի իրական ժամանակում մշակումը, կարող են պահանջել չափազանց մեծ հաշվարկային հզորություններ, այդպիսով բացելով մեծ դաշտ ԱԲ արագացման համար նախատեսված նոր սարքերի հետազոտման և նախագծման համար: Այդպիսի սարքերի օրինակներ են Nvidia-ի Jetson շարքը, Google-ի Coral-ը: Կան նաև FPGA լուծումներ, ինչպիսին է Xilinx-ի ԱԲ կիրառելի FPGA-ը: Կախված հավելվածի պահանջներից՝ կարող է դժվար լինել ընտրության կատարումը այս սարքերի միջև, քանի որ հիմնականում ամենակարևոր գործոնները արագությունն ու ճշգրտությունն են, մինչդեռ կան նաև հավելվածներ, որոնք պահանջում են ցածր հզորություն և գին: Այս եզրակացության սարքերի հետազոտումն ու համեմատումը, ընտրված ոլորտի համար արագության, հիշողության, ծախսած հզորության և գնի տեսանկյունից, կարող են օգտակար լինել տրված խնդրի համար ճիշտ սարք ընտրելու գործում: Կատարվել է այսպիսի վերլուծություն պատկերի խորության գնահատման խնդրի համար՝ որպես եզրակացության սարքեր օգտագործելով Jetson Xavier NX-ը և Coral Dev Board-ը:

Առանցքային բառեր. արհեստական բանականություն, նեյրոնային ցանց, խորության գնահատում, Jetson Xavier NX, Coral Dev Board:

Т.Б. ХАЧАТРЯН, Д.Ф. ДАВТЯН

СРАВНЕНИЕ УСТРОЙСТВ ЛОГИЧЕСКОГО ВЫВОДА JETSON XAVIER NX И CORAL DEV BOARD ДЛЯ ОЦЕНКИ ГЛУБИНЫ ИЗОБРАЖЕНИЯ С ПОМОЩЬЮ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Логический вывод искусственного интеллекта (ИИ), в частности - обработка нейронных сетей в реальном времени, требует чрезмерно больших вычислительных мощностей, тем самым открывая большую область для исследования и проектирования новых устройств ускорения ИИ. Примерами таких устройств являются Nvidia Jetson, Google Coral. Имеются также решения FPGA, такие как FPGA Xilinx, для применения ИИ. В зависимости от требований приложения, выбор этих устройств может быть трудным, так как самыми главными факторами в основном являются скорость и точность, но имеются приложения, которые требуют низких мощностей и цен. Таким образом, исследование и сравнение этих устройств применения, с точки зрения скорости, памяти, потраченной мощности и цены для выбранной сферы, позволят выбрать правильное устройство для поставленной задачи. Проведен анализ для задачи оценки глубины изображения, используя в качестве устройств применения Jetson Xavier NX и Coral Dev Board.

Ключевые слова: искусственный интеллект, нейронная сеть, оценка глубины, Jetson Xavier NX, Coral Dev Board.