DepthNet Nano: A Highly Compact Self-Normalizing Neural Network for Monocular Depth Estimation

Linda Wang, Mahmoud Famouri, and Alexander Wong
Department of Systems Design Engineering, University of Waterloo, Canada
Waterloo Artificial Intelligence Institute, Canada
DarwinAI Corp., Canada
{linda.wang, alexander.wong}@uwaterloo.ca

Abstract

Depth estimation is an active area of research in the field of computer vision, and has garnered significant interest due to its rising demand in autonomous driving applications. A particularly challenging problem in this area is monocular depth estimation, where the goal is to infer depth from a single image. An effective strategy that has shown considerable promise in recent years for tackling this problem is the utilization of deep convolutional neural networks. Despite these successes, the memory and computational requirements of such networks have made widespread deployment in embedded scenarios very challenging. In this study, we introduce DepthNet Nano, a highly compact self normalizing network for monocular depth estimation designed using a human machine collaborative design strategy, where principled network design prototyping based on encoder-decoder design principles are coupled with machine-driven design exploration. The result is a compact deep neural network with highly customized macroarchitecture and microarchitecture designs, as well as self-normalizing characteristics, that are highly tailored for the task of embedded depth estimation. The proposed DepthNet Nano possesses a highly efficient network architecture (e.g., $24\times$ smaller and $42\times$ fewer MAC operations than Alhashim et al.), while still achieving comparable performance with state-of-the-art networks on the KITTI dataset. Furthermore, experiments on inference speed and energy efficiency on a Jetson AGX Xavier embedded module further illustrate the efficacy of DepthNet Nano at different power budgets (e.g., $\sim$14 FPS and $>$0.46 images/sec/watt at 384×1280 at a 30W power budget on KITTI).

1 Introduction

The task of estimating depth from 2D images is crucial for 3D scene understanding in many autonomous driving applications. Recently, monocular depth estimation, where a dense depth map is obtained from a single image, as shown in Figure 1, has gained traction. Compared to depth estimation from stereo images or video sequence, monocular depth estimation is an ill-posed problem, which means there are more than one possible unique solution. To tackle this ill-posed problem, one method that has shown promise is deep convolutional neural networks (DCNNs).

These DCNNs learn deep features to estimate a depth for each pixel. Most of the recent developments have focused on an encoder-decoder architecture with very powerful deep neural network backbone macroarchitecture designs, such as VGG, ResNet and DenseNet [1, 2, 3], to learn deep features. However, these current deep neural networks for monocular depth estimation are large and difficult to deploy and run in edge scenarios such as autonomous vehicles. In addition to possessing very high network architecture complexity, these current deep neural networks have high computation time and

Machine Learning for Autonomous Driving Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

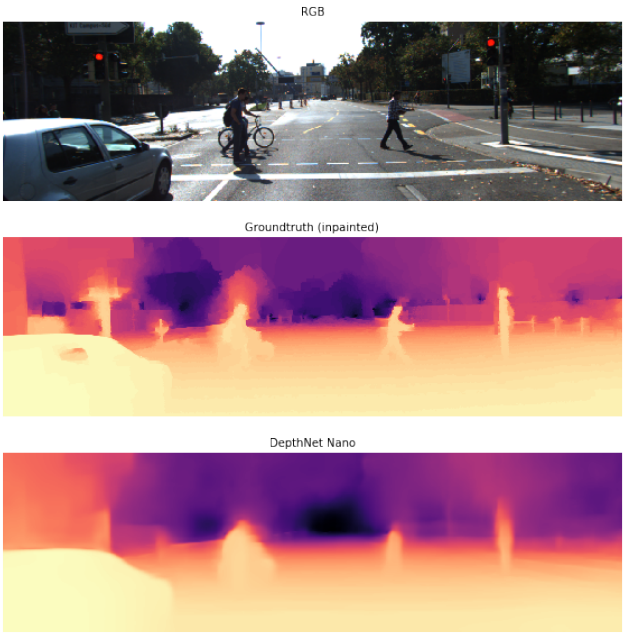

Figure 1: **Monocular depth estimation.** Top: RGB image (from KITTI dataset), Middle: Corresponding ground-truth dense depth map. Bottom: Estimated dense depth map by Depth-Net Nano (designed using the human-machine collaborative design strategy).

low energy efficiency, which makes them difficult to deploy for mission-critical scenarios such as self-driving cars, where reaction time is crucial for driver safety.

Taking inspiration from recent work in efficient object detection, where large backbone architectures are replaced with more efficient backbone network architectures, such as MobileNetV2 [4], Wofk et al. [5] used smaller backbone architectures (e.g., ResNet-16 and MobileNet) in order to decrease the number of parameters and run-time necessary to operate on embedded devices. To further reduce the network size and inference runtime, network pruning was applied [6]. While the resulting depth estimation networks achieved significant improvements in terms of inference speed, the depth estimation performance was significantly lower and not comparable with current state of the art.

Taking a different direction than manual architecture selection strategies and network pruning strategies, human-machine collaborative design strategies [7] has shown recent success in designing highly compact DCNNs by coupling principled network design prototyping and machine-driven design exploration based on human-specified design requirements and constraints. In particular, such strategies have been demonstrated to be quite effective at designing efficient deep neural networks well suited for various perception tasks, such as object detection, image classification, and semantic segmentation [8, 7].

In this study, we explore a human-machine collaborative design strategy to design highly compact deep convolutional neural networks for the task of monocular depth estimation on the edge. More specifically, we leverage encoder-decoder design principles that were found to be effective in current state of the art monocular depth estimation to create DepthNet Nano, a highly compact network with highly customized module-level macroarchitecture and microarchitecture designs tailored specifically for embedded depth estimation. In Figure 2, we show that DepthNet Nano is significantly smaller and faster than current state of the art depth estimation networks while maintaining a comparable accuracy on the KITTI benchmark dataset.

The paper is organized as follows. Section 2 provides a detailed description of the human-machine collaborative design strategy leveraged in this study to create DepthNet Nano. Section 3 provides a detailed description and discussion of interesting characteristics of the resulting architecture design of DepthNet Nano. Section 4 presents and discusses the results from the quantitative and qualitative experiments conducted to study the efficacy of DepthNet Nano when compared to state-of-the-art depth estimation networks. Finally, Section 5 draws conclusions and discusses future directions.

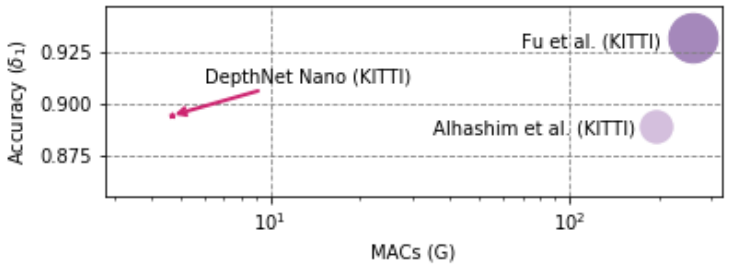

Figure 2: **Accuracy vs. MACs and number of parameters** comparison of various depth estimation networks. The size of the point represents the number of parameters in the depth estimation network. Top left represents high accuracy and high efficiency.

2 Methods

In this study, we leverage a human-machine collaborative design strategy to design DepthNet Nano, a highly compact DCNN tailored for monocular depth estimation under edge scenarios. The human-machine collaborative design strategy comprises of two main design stages: i) principled network design prototyping, and ii) machine-driven design exploration.

2.1 Principled Network Design Prototyping

The network design prototyping stage is the initial design stage, where we create an initial network design prototype (denoted as φ) based on human-driven design principles to guide the machine-driven design exploration stage. In this study, an initial network design prototype was constructed based on densely-connected encoder-decoder architecture design principles [2], which has been demonstrated to be quite successful in achieving high-resolution monocular depth estimation.

A standout characteristic of the densely-connected encoder-decoder architecture is the leveraging of a large number of direct connections between not only encoder layers, but also between encoder and decoder layers. Below is a detailed description of the design principles leveraged in constructing the initial network design prototype. It is very important to note that the actual macroarchitecture and microarchitecture designs of the individual modules and layers in the final DepthNet Nano network architecture, as well as the number of network modules, are left for the machine-driven design exploration stage to decide in an automatic manner based on both human-specified design requirements and constraints catered to edge device scenarios with limited computational and memory capabilities.

2.1.1 Encoder-Decoder Architecture

In this study, the initial network design prototypes possess an encoder-decoder architecture that is designed for dense depth map generation. The encoder layers in such an architecture are designed to learn a multitude of low to mid level features for characterizing an input scene. Next, the decoder layers are designed to merge and upsample the features learned from the encoder layers to recover a dense depth map. The decoder layers consist of upsampling blocks followed by a concatenation and two convolutional operations. Finally, the encoder-decoder architecture leveraged by the initial network design prototypes follows the decoder layers with a 3×3 convolutional layer at the end, which is designed to produce the final dense depth map.

2.1.2 Encoder-Encoder and Encoder-Decoder Skip Connections

In general, as the number of layers increases, the network accuracy improves because each layer is learning deeper features. However, He et al. [9] found that a 56-layer deep convolutional neural network has higher training and test error than a 26-layer CNN deep convolutional neural network introduced a fundamental building block, the residual block, to alleviate training of deep neural networks. The residual block adds a previous layer to the current layer. By adding information from previous layers, the network can learn residuals or errors between a previous layer and the current one. Extending upon this idea of skip connections, densely-connected network architectures

consists of a large number of skip connections between different layers. More specifically, instead of adding the previous layer as an identity function, densely-connected architectures concatenate outputs from previous layers to the current layer. This is found to alleviate the vanishing gradient problem, strengthen feature propagation and feature reuse [10]. A such, the introduction of encoder-encoder skip connections into a deep encoder-decoder architecture can improve the training process and improve network performance.

Furthermore, in the case of encoder-decoder architectures, as the layers in the encoder get deeper, higher level features are learned; however, the resolution of the feature maps get progressively lower. As such, the input to the decoder is of low resolution. Since the purpose of the decoder network is to upsample the features learned from the encoder, the resulting depth map image would also have of low resolution.

To overcome the low resolution problem, an effective strategy is the leveraging of skip connections between encoder and decoder layers within an encoder-decoder architecture. Such encoder-decoder skip connections merge high resolution feature maps from the encoder layers to the features in the decoder layers, resulting in a more detailed decoder output.

2.2 Machine-driven Design Exploration

The machine-driven design exploration stage takes in the given data, initial network design prototype φ, and human-specified design requirements and constraints, which are designed specifically around edge scenarios with limited computational and memory capabilities.

Using the initial network design prototype φ described in the previous section, as well as human specified design requirements, a machine-driven design exploration is leveraged in the form of generative synthesis [7] to determine macroarchitecture and microarchitecture designs for depth estimation on edge devices. The process of generative synthesis is capable of determining the optimal network macroarchitecture and microarchitecture design that satisfy the human-specified constraints. This is achieved by learning generative machines that can generate deep neural networks that meet the specified constraints. To learn the optimal generator, generative synthesis is formulated as a constrained optimization problem, defined in Equation 1, where given a set of seeds $\mathcal{S}$, a generator $\mathcal{G}$ can generate networks $\{\mathcal{N}_s | s \in \mathcal{S}\}$ that maximize a universal performance function $\mathcal{U}$, while also satisfying constraints defined in an indicator function $1_r(\cdot)$.

$$\mathcal{G} = \max_{\mathcal{G}} \mathcal{U}(\mathcal{G}(s)) \text{ subject to } 1_r(\mathcal{G}(s)) = 1, \forall s \in \mathcal{S} \tag{1}$$

The generative synthesis process is guided by both the initial prototype φ and human-specified constraints. To guide the process towards learning generative machines that generate highly efficient and compact depth estimation networks for edge devices, an indicator function $1_r(\cdot)$ is configured so that the generated networks are within the human-specified constraints. In this study, for generating a highly efficient and compact depth estimation network tailored for KITTI, the indicator function $1_r(\cdot)$ was set up for this case such that: i) δ_1 accuracy $\geq$ 0.89 on KITTI, and ii) architectural complexity $\leq$ 2M parameters. The δ_1 accuracy and network architecture complexity conditions in the indicator function $1_r(\cdot)$ are set such that the δ_1 accuracy of the resulting DepthNet Nano network exceeds to that of Alhashim et al. [2], a popular, high-performance deep convolutional neural network for monocular depth estimation for KITTI, while having more than 20$\times$ fewer parameters.

Finally, in this study, the universal performance function $\mathcal{U}$ leveraged in Eq. 1 is NetScore [11], a quantitative performance metric designed for assessing the balance between accuracy, computational complexity, network architecture complexity of a deep neural network. The NetScore metric is defined as

$$\Omega(\mathcal{N}) = 20 \log \Big(\frac{a(\mathcal{N})^{\kappa}}{p(\mathcal{N})^{\beta} r(\mathcal{N})^{\gamma}} \Big) \tag{2}$$

where for this study, $a(\mathcal{N})$ is the combination of δ_1 accuracy and absolute relative error, $p(\mathcal{N})$ is the number of parameters in the network, $m(\mathcal{N})$ is the number of multiply-accumulate (MAC) operations, and κ, β, γ control the the influence of accuracy, architectural complexity and computational complexity, respectively. For this study, κ is set to 0.7, β and γ are both set to 0.15 to put an emphasis on accuracy while maintaining balance with architectural complexity and computational complexity.

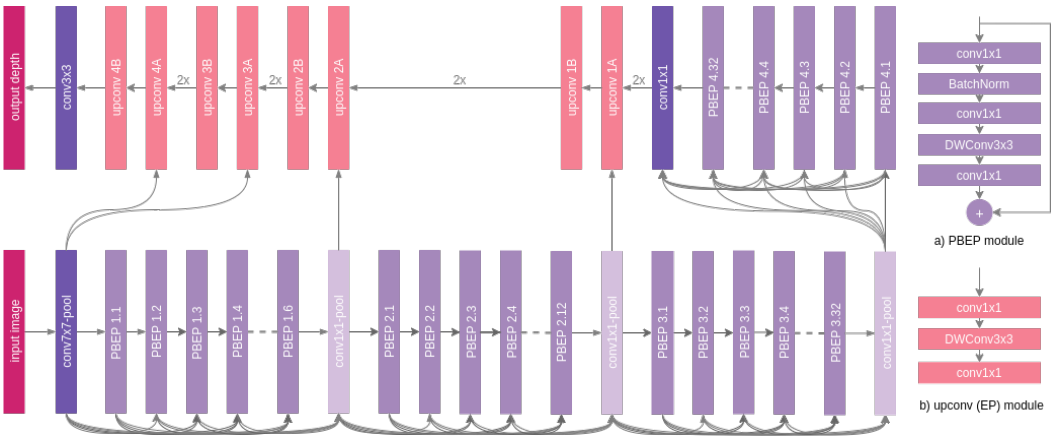

Figure 3: **DepthNet Nano Architecture.** The network architecture exhibits high macroarchitecture and microarchitecture heterogeneity with a mix of PBEP and EP modules, as well as individual 7×7, 3×3, and pointwise convolution layers. Finally, the network architecture possesses a very deep densely-connected self-normalization macroarchitecture, which has not been previously explored. Microarchitecture details for each individual layer can be found in Appendix.

3 DepthNet Nano Architectural Design

The network architecture of the proposed DepthNet Nano, which is illustrated in Fig. 3 and has several interesting characteristics that are discussed in detail below.

3.1 Self-Normalization Macroarchitecture

The first interesting characteristic of the DepthNet Nano architecture is its self-normalizing property within a very deep densely-connected network architecture, which has not been previously explored. More specifically, rather than leveraging popular activation functions such as Rectifier Linear Units (ReLU) that are more commonly found in depth estimation networks, the proposed DepthNet Nano architecture heavily leverages Scaled Exponential Linear Units (SELU) [12] as the only form of activation within the network architecture, which can be defined as

$$\text{selu}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha \exp(x) - \alpha & \text{if } x \leq 0 \end{cases} \tag{3}$$

One of the key advantages of SELUs is their self-normalizing properties that makes learning more robust for deep neural networks. More specifically, since the activations in each layer of the network are close to zero mean and unit variance, as the data is propagated through, it will converge towards zero mean and unit variance. The self-normalizing property is achieved with SELUs by decreasing the variance for negative inputs and increasing the variance for positive inputs. To achieve zero mean and unit variance, the amount of decrease for very negative inputs and the amount of increase for near zero values are greater than other inputs. The result is a self-normalizing neural network that is able to achieve high depth estimation accuracy while being extremely efficient and possessing very low network architecture complexity despite high network inter-connectivity.

3.2 Densely Connected Projection BatchNorm Expansion Projection Macroarchitecture

Another interesting characteristic of the DepthNet Nano architecture is the densely connected projection batchnorm expansion projection (PBEP) module, which are leveraged heavily in the encoding layers of the network architecture (see Fig. 3). Compared to the expansion projection (EP) modules leveraged extensively in the decoding layers of the network architecture (see Fig. 3), which have been seen in other literature on efficient network architectures [4, 13, 14], PBEP has an additional projection layer that decreases the number of channels of the previous layer before expanding the layer for depth-wise convolution. PBEP macroarchitecture consists of:

Table 1: **Performance on KITTI.** All networks are evaluated using a pre-defined center cropping [15]. Best results in **bold**.

Model	Input Size	MACs [G]	Params [M]	higher is better			lower is better			
				$\delta_1 < 1.25$	$\delta_2 < 1.25^2$	$\delta_3 < 1.25^3$	Abs Rel	Sq Rel	RMSE	RMSE *log*
Fu et al. [1]	385×513	258	99.8	**0.932**	**0.984**	**0.994**	**0.072**	**0.307**	**2.727**	**0.120**
Alhashim et al. [2]	384×1280	196	42.8	0.886	0.965	0.986	0.093	0.589	4.170	0.171
DepthNet Nano	384×1280	**4.66**	**1.75**	0.894	0.978	**0.994**	0.103	0.511	3.916	0.150

1. A projection layer, where the output channels of the previous layer are projected to a lower dimensionality in this layer using 1×1 convolutions.
2. A batch normalization layer that normalizes the output of a previous layer to improve the stability of the network.
3. An expansion layer, where 1×1 convolutions are leveraged to expand the output of the batch normalization layer to a higher dimensionality.
4. A depth-wise convolution layer, where spatial convolutions with a different filter are applied to each of the individual output channels of the expansion layer.
5. A projection layer with 1×1 convolutions that projects the output channels from the depth-wise convolution layer to a lower dimensionality.

The use of densely connected PBEP macroarchitectures reduces the architectural and computational complexity of the DepthNet Nano architecture while maintaining high model expressiveness and producing high quality depth estimations.

3.3 Macroarchitecture and Microarchitecture Heterogeneity

Unlike hand-crafted architectures, the generated macroarchitecture and microarchitecture within the network can differ greatly from layer to layer. There are a mix of different type of modules, such as PBEP and EP modules, as well as individual 7×7, 3×3 and 1×1 convolution layers. In addition, the same module type has vastly different microarchitectures since each module is catered specifically for the needs of the task (see Appendix for microarchitecture details for each layer). For instance, each PBEP and EP module have different numbers of channels to represent the learned features and a different multiplicity for the channel expansion layer.

The benefit of high macroarchitecture and microarchitecture heterogeneity in DepthNet Nano network architecture is that it enables each component of the network architecture to be uniquely tailored to achieve a very strong balance between architectural and computation complexity while maintaining model expressiveness. The architectural diversity in DepthNet Nano demonstrates the advantage of leveraging a human-collaborative design strategy as it would be difficult for a human designer, or other design exploration methods to customize a network architecture to the same level of architectural granularity.

4 Experimentation Results

To demonstrate the efficacy of the proposed DepthNet Nano network designed using the human-machine collaborative design strategy, we examine its network architecture complexity, depth estimation performance, and computational cost on the KITTI benchmark dataset.

4.1 Dataset and Implementation Details

In this study, we leverage the KITTI dataset, which contains 2D outdoor scenes and corresponding ground-truth lidar points captured using a Velodyne HDL-64E [16]. Since this study requires dense input depth maps, 23158 lidar scans were inpainted using Levin et al's colorization method [17], as consistent with other studies. For this study, using the same procedure as previous literature, the network was trained on 23158 color images and their corresponding dense depth maps, and tested on 697 images from 29 scenes split by Eigen et al. [15]. For testing, the depth predictions for are evaluated on a pre-defined center cropping by Eigen [15].

The proposed DepthNet Nano was implemented using the TensorFlow open source platform for machine learning. The training scheme leveraged in this study was that outlined in [2]. The ADAM

Table 2: **KITTI Inference.** All networks are evaluated on a Jetson AGX Xavier embedded module. Best results in **bold**.

Model	MACs [G]	δ_1	30W [FPS]	15W [FPS]	30W [$\frac{\text{images/s}}{\text{watt}}$]	15W [$\frac{\text{images/s}}{\text{watt}}$]
Alhashim et al. [2]	196	0.886	3.00	1.65	0.100	0.110
DepthNet Nano	**4.66**	**0.894**	**13.92**	**7.70**	**0.464**	**0.513**

optimizer [18] was leveraged with a learning rate of 0.00005 and parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

4.2 Performance Evaluation Metrics

Each tested network in this study is evaluated using several error and accuracy metrics that were used in prior works [2, 15, 1, 3]. More specifically, the following performance metrics were leveraged in this study:

- relative absolute error (Abs Rel): $\frac{1}{N}\sum_{i=1}^{N}\frac{|y_i-\hat{y}_i|}{\hat{y}_i}$
- relative squared error (Sq Rel): $\frac{1}{N}\sum_{i=1}^{N}\frac{(y_i-\hat{y}_i)^2}{\hat{y}_i}$
- root mean squared error (rmse): $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2}$
- log rmse (rmse log): $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(log(y_i)-log(\hat{y}_i))^2}$
- average log error (Log10): $\frac{1}{N}\sum_{i=1}^{N}|log(y_i)-log(\hat{y}_i)|$
- δ_i accuracy: % of y_i s.t. $max(\frac{y_i}{\hat{y}_i}, \frac{\hat{y}_i}{y_i}) = \delta < thr$ for $thr = 1.25, 1.25^2, 1.25^3$

where y_i is a pixel in predicted depth image, $\hat{y}_i$ is a pixel in the ground-truth depth image and N is the total number of pixels in the depth image.

Finally, evaluating the real-world performance of the proposed DepthNet Nano in a realistic embedded scenario, we performed inference speed and power efficiency evaluations on a Jetson AGX embedded module at two different power budgets: 1) 30W and 2) 15W. For inference speed, we computed the framerate (FPS), while for power efficiency we computed the number of images processes per sec per watt (i.e., images/sec/watt).

4.3 Quantitative Analysis

To study the efficacy of human-machine collaborative design, we evaluate DepthNet Nano alongside state-of-the-art depth estimation networks on performance metrics defined in Section 4.2, network architecture complexity, and computational complexity on the KITTI datasets, as shown in Table 1, respectively. Since this study targets high quality depth estimation, only state-of-the-art networks in literature with δ_1 accuracies greater than 0.88 are considered.

Performance and complexity. It was observed that DepthNet Nano had significantly lower architecture complexity and computational complexity compared to the tested state-of-the-art networks. For example, DepthNet Nano has $>24\times$ fewer parameters and requires $>42\times$ fewer multiply-accumulate operations (MACs) for inference than [2]. Furtherore, despite the much lower architectural and computatinal complexity, DepthNet Nano was able to achieve higher δ_1, δ_2 and δ_3 accuracies than [2].

Speed and energy efficiency. The inference speed of DepthNet Nano and [2] are compared on a Jetson AGX Xavier embedded module in Table 2. For KITTI, DepthNet Nano was more than $4.6\times$ faster and more energy efficient than Alhashim et al. [2] at both 30W and 15W power budgets. These quantitative results demonstrate that the proposed DepthNet Nano networks, created using a human-machine collaborative design strategy, can achieve a strong balance between accuracy, network architecture complexity, and computational complexity that makes it very well suited for embedded depth estimation for edge scenarios such as autonomous driving.

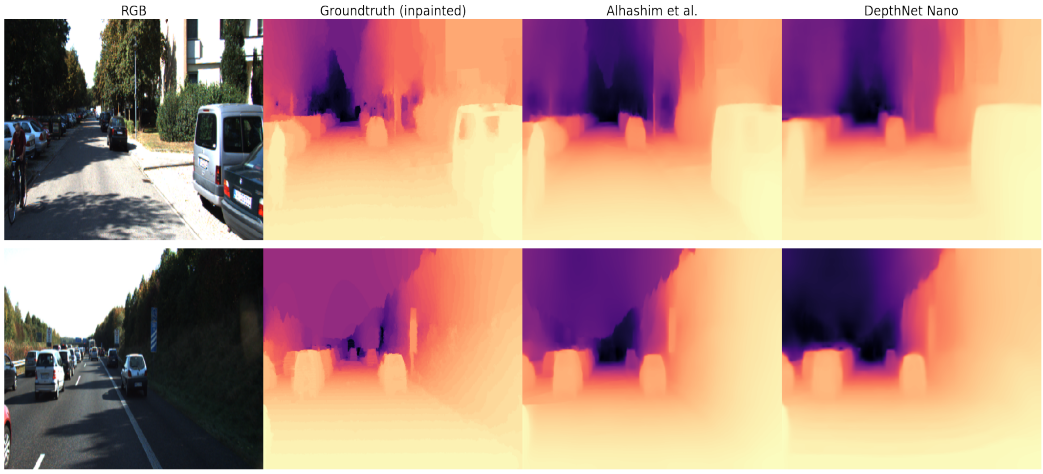

Figure 4: **Visualized results on KITTI dataset.** (Left to right) input RGB image, ground truth, and depth estimations from Alhashim et al. [2] and DepthNet Nano. DepthNet Nano was able to produce high-quality depth estimations despite requiring 42× fewer MAC operations than Alhashim et al. [2].

4.4 Qualitative Analysis

In additional to quantitatively evaluating the performance DepthNet Nano, a qualitative analysis is also conducted to present areas that may be not be evaluated by the error metrics. Figure 4 shows examples from the KITTI dataset. Each RGB image has a corresponding ground-truth dense depth map and predicted dense depth maps from [2] and DepthNet Nano. It can be observed that the dense depth maps produced by DepthNet Nano are visually comparable to that produced by [2], despite DepthNet Nano requiring 42× fewer MAC operations. This observation further reinforces the strong balance between accuracy, network architecture complexity, and computational complexity achieved by the proposed DepthNet Nano that makes it very well suited for embedded depth estimation for edge scenarios such as autonomous driving.

5 Conclusion

In this study, we introduced DepthNet Nano, a highly compact self normalizing network that is tailored for embedded monocular depth estimation and designed using a human machine collaborative design strategy. By coupling human-driven principled network design prototyping and machine-driven design exploration, the resulting DepthNet Nano network architecture exhibited highly customized macroarchitecture and microarchitecture designs, as well as self-normalizing characteristics that provide a strong balance between architecture complexity, computational complexity, and depth estimation performance. Experimental results on the KITTI benchmark dataset demonstrated that the proposed DepthNet Nano possesses a significantly more architecturally and computationally efficient network architecture compared with state-of-the-art networks while achieving comparable performance. Furthermore, experiments demonstrated that DepthNet Nano had significantly faster inference speeds and energy efficiency on the Jetson AGX Xavier embedded module. For future work, we plan to explore strategies for incorporating temporal information into the DepthNet Nano architecture in a way that improves performance while maintaining low architecture and computational complexity.

References

[1] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011.

[2] I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," *arXiv preprint arXiv:1812.11941*, 2018.

[3] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *2016 Fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 239–248.

[4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[5] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "Fastdepth: Fast monocular depth estimation on embedded systems," *arXiv preprint arXiv:1903.03273*, 2019.

[6] T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, V. Sze, and H. Adam, "Netadapt: Platform-aware neural network adaptation for mobile applications," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 285–300.

[7] A. Wong, M. J. Shafiee, B. Chwyl, and F. Li, "Ferminets: Learning generative machines to generate efficient neural networks via generative synthesis," *arXiv preprint arXiv:1809.05989*, 2018.

[8] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, "Yolo nano: a highly compact you only look once convolutional neural network for object detection," *arXiv preprint arXiv:1910.01271*, 2019.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[11] A. Wong, "Netscore: Towards universal metrics for large-scale performance analysis of deep neural networks for practical on-device edge usage," in *International Conference on Image Analysis and Recognition*. Springer, 2019, pp. 15–26.

[12] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Advances in neural information processing systems*, 2017, pp. 971–980.

[13] M. Tan, B. Chen, R. Pang, V. Vasudevan, and Q. V. Le, "Mnasnet: Platform-aware neural architecture search for mobile," *arXiv preprint arXiv:1807.11626*, 2018.

[14] X. Chu, B. Zhang, R. Xu, and J. Li, "Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search," *arXiv preprint arXiv:1907.01845*, 2019.

[15] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems*, 2014, pp. 2366–2374.

[16] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[17] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *ACM SIGGRAPH 2004 Papers*, 2004, pp. 689–694.

[18] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2017.

Appendix

Microarchitecture details

The microarchitecture details for each layer of the proposed DepthNet Nano is shown in Table 3.

Table 3: **DepthNet Nano microarchitecture details.**

Layer	Size
input	$384 \times 1280 \times 3$
conv7 $\times$ 7-pool	$192 \times 640 \times 14$
PBEP 1.1	$96 \times 320 \times 13$
PBEP 1.2	$96 \times 320 \times 15$
PBEP 1.3	$96 \times 320 \times 18$
PBEP 1.4	$96 \times 320 \times 13$
...	...
PBEP 1.6	$96 \times 320 \times 17$
conv1 $\times$ 1-pool	$48 \times 160 \times 30$
PBEP 2.1	$48 \times 160 \times 9$
PBEP 2.2	$48 \times 160 \times 13$
PBEP 2.3	$48 \times 160 \times 13$
PBEP 2.4	$48 \times 160 \times 18$
...	...
PBEP 2.12	$48 \times 160 \times 19$
conv1 $\times$ 1-pool	$24 \times 80 \times 79$
PBEP 3.1	$24 \times 80 \times 13$
PBEP 3.2	$24 \times 80 \times 14$
PBEP 3.3	$24 \times 80 \times 18$
PBEP 3.4	$24 \times 80 \times 15$
...	...
PBEP 3.32	$24 \times 80 \times 14$
conv1 $\times$ 1-pool	$12 \times 40 \times 117$
PBEP 4.1	$12 \times 40 \times 17$
PBEP 4.2	$12 \times 40 \times 13$
PBEP 4.3	$12 \times 40 \times 14$
PBEP 4.4	$12 \times 40 \times 12$
...	...
PBEP 4.32	$12 \times 40 \times 11$
conv1 $\times$ 1	$12 \times 40 \times 176$
upconv 1A	$24 \times 80 \times 86$
upconv 1B	$24 \times 80 \times 112$
upconv 2A	$48 \times 160 \times 47$
upconv 2B	$48 \times 160 \times 48$
upconv 3A	$96 \times 320 \times 28$
upconv 3B	$96 \times 320 \times 25$
upconv 4A	$192 \times 640 \times 17$
upconv 4B	$192 \times 640 \times 24$
conv3 $\times$ 3	$192 \times 640 \times 1$
output	$384 \times 1280 \times 1$