Knowledge Distillation for Fast and Accurate Monocular Depth Estimation on Mobile Devices

Yiran Wang, Xingyi Li, Min Shi, Ke Xian, and Zhiguo Cao*
Key Laboratory of Image Processing and Intelligent Control, Ministry of Education
School of Artificial Intelligence and Automation, Huazhong University of Science and Technology
{wangyiran, zgcao}@hust.edu.cn

Abstract

Fast and accurate monocular depth estimation on mobile devices is a challenging task as one should always trade off the accuracy against the inference time. Most monocular depth methods adopt models with large computation overhead, which are not applicable on mobile devices. However, directly training a light-weight neural network to estimate depth can yield poor performance. To remedy this, we utilize knowledge distillation, transferring the knowledge and representation ability of a stronger teacher network to a light-weight student network. Experiments on Mobile AI 2021 (MAI2021) dataset demonstrate that our solution helps increase the fidelity of the output depth map and maintain fast inference speed. Specifically, with 94.7% less parameters than teacher network, the si-RMSE of student network only decrease by 10%. ***Moreover, our method ranks second in the MAI2021 Monocular Depth Estimation Challenge, with a si-RMSE of 0.2602, a RMSE of 3.25, and the inference time is 1197 ms tested on the Raspberry Pi 4.***

1. Introduction

Real-time monocular depth estimation on mobile devices is a task in great demand. For example, accurate depth estimation helps robots sense the surroundings. Meanwhile, depth estimation is a preliminary task for many applications, such as semantic segmentation [7], bokeh effect rendering [31], and relighting [15]. Many other applications also need to be deployed on mobile devices like smartphones such as image enhancement [22]and super resolution [43]. However, most depth estimation methods adopt convolutional neural networks (CNNs) with complex architectures and large computation overhead, which makes them infeasible for real-time depth estimation on mobile devices due to the limited computing power.*

*Corresponding author.

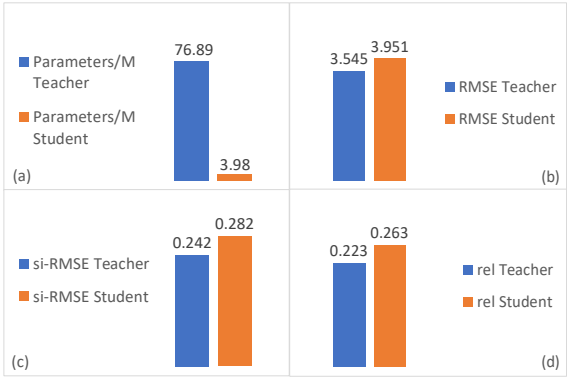

Figure 1. Comparison of teacher and student networks. with 94.7% less parameters than teacher network, the RMSE of student network only decrease by 10%, which can show the efficiency of our knowledge distillation strategy. These metrics are defined in section 4.1. (a) Number of parameters; (b) RMSE; (c) si-RMSE; (d) rel.

A straightforward way to implement real-time depth estimation is to reduce the complexity of CNN. For dense prediction tasks, most methods adopt a fully convolutional network with an encoder and a decoder. One can apply light-weight encoder and decoder, e.g., MobileNet [18] and its variants [36], to extract features from input images. Model complexity can also be reduced by pruning redundant parameters [14]. Nonetheless, trade-off always exists between the accuracy and the efficiency. Directly training a light-weight network can not obtain depth maps with high accuracy and fidelity. Naturally, we come to the question that how we can take advantage of a complex and deep model with strong capability to enhance the performance of a light-weight network. To address this, in this paper, we transfer the strong representation ability of a teacher network to a light-weight student network via knowledge distillation [33]. Specifically, during training, the student network learns to extract similar feature maps at different scales as

the teacher network does. We evaluate our method on Mobile AI (MAI2021) dataset [2], and the results demonstrate the knowledge distillation framework can improve the accuracy of the light-weight student networks. With a reduction of 94.7% parameters, compared to teacher network, the RMSE of student network only decrease by 10%. Our method ranks second in the Mobile AI 2021 Monocular Depth Estimation Challenge [19].

In summary, our main contributions are:

- We utilize knowledge distillation for fast and accurate monocular depth estimation on mobile devices.
- We evaluate our method on MAI2021 dataset and show the effectiveness of our method.
- We rank the second place in the MAI2021 Monocular Depth Estimation Challenge.

2. Related work

In this section, we introduce some past research achievements on monocular depth estimation, knowledge distillation, and neural networks for mobile devices.

Monocular depth estimation Monocular depth estimation has become an active field in computer vision in recent decades. Its fundamental task is to recover the corresponding depth information from a single RGB three-channel image captured by a monocular camera. At the early stage, depth estimation is obtained by image retrieval. Saxena *et al*. [37] extract hand-crafted features from input images and select most similar samples from a pre-collected database to obtain depth map patch by patch. After that, with deep learning and convolutional neural networks achieving tremendous progress in many tasks (e.g. classification, object detections), deep learning-based methods become the mainstream. Eigen *et al*. [11] first utilize the power of deep learning to estimate depth maps and surpass the previous methods by a large margin. A constellation of learning-based method are purposed afterwards and achieve higher accuracy and robustness. For example, Liu *et al*. [28] combine convolutional neural networks (CNN) with conditional random field (CRF) . They explore the unary and pairwise potentials of the CRF using a unified CNN framework. Li *et al*. [26] reckon depth estimation as a classification task, where depth value at each pixel is discretized into different classes. They predict the depth categories for each pixel. They designed attention-based network architecture which can predict both indoor and outdoor scenes quite well. Eigen *et al*. [10] simultaneously estimate depth maps, surface normals and semantic labels using a common network architecture. Based on the residual learning framework [13], Laina *et al*. [23] build a deeper network and achieve high accuracy.

Besides works on network architectures, there have been plenty of progress on other aspects of the learning-based methods. For instance, Eigen *et al*. [11] present their scale-invariant loss function aiming to measure depth relations and accuracy , irrespective of the absolute global scale. . Xian *et al*. [41] propose a new dataset with great images diversity and dense depth labels. They also introduce an improved ranking loss to solve the imbalanced ordinal relations. Li *et al*. [27] use multi-view Internet images as an unlimited data source and present a new dataset named Megadepth with great amounts of photos in it. Ranftl *et al*. [35] develop a method in which depth prediction models can be trained on mixed multiple datasets despite their incompatible labels.

It is also worth noticing that some methods also take the inference time and model complexity into account, which makes them applicable on mobile devices like smart phones or embedded systems. Ranftl *et al*. [35] achieve great success dealing with the trade off between accuracy and speed. Their Midas-v2.1 small achieve 30 FPS on iPhone 11. FastDepth [40] is another exemplar in this field. They deploy a real-time depth estimation method on embedded systems by designing a efficient model architecture and a pruning strategy to further reduce the model complexity. In our approach, we adopt the same network architecture as FastDepth [40] and apply knowledge distillation on it to improve its performance.

Knowledge distillation Reducing the model complexity and computation overhead while maintaining the performance has long been a popular topic. One feasible way is to simplify the model, e.g., pruning the redundant parameters [14], model quantization [34]. Here, we focus on an elegant way called knowledge distillation, which is first purposed by Hinton *et al*. [16]. Knowledge distillation aims at transferring knowledge from a cumbersome teacher network to a compact student network. This training strategy was initially designed for image classification and gradually transferred to other tasks such as semantic segmentation [29], objection detection [6] and depth prediction [33]. At the early stage, the pixel-wise distillation strategy was mainly utilized to distill the class probabilities distribution for each pixels. Shen *et al*. [30] expand pixel-wise distillation to pair-wise distillation and holistic distillation. They purpose a generic structured knowledge distillation framework for dense prediction tasks. In holistic distillation part, they use conditional generative adversarial learning [32] and introduce the discriminator to formulate the holistic distillation problem. We mainly refer to [30] and [33] to build our own training framework.

Neural networks for mobile devices There have been lots of effort in designing efficient networks for mobile

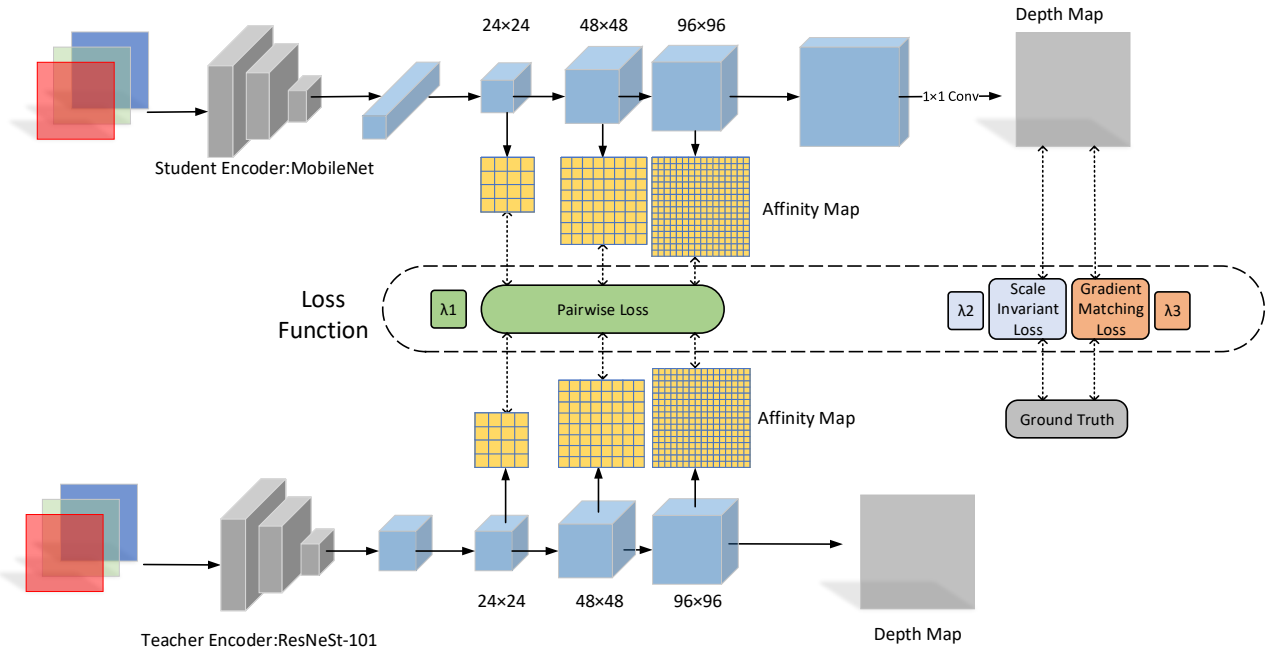

Figure 2. The overview of our method. The above network represents the student model while the below stands for the teacher model. We build our distillation strategy between the decoders of two networks. Meanwhile, we build the scale invariant loss and gradient matching loss between the student model output and ground-truth.

devices. These networks with extraordinary architectures pave the way of many real-time tasks on mobile devices. MobileNet-V1 [18] reduces the parameters by 31.9 times compared to VGG16 [38] by replacing the standard convolutions with the depth-wise separable convolutions. MobileNet-V2 [36] introduces the inverted residuals and linear bottlenecks structure. To further show the potential of MobileNet V2, the original model is developed to an object detection framework SSDLite and a light-weight semantic segmentation framework DeepLab-V3 [8]. MobileNet V3 [17] utilizes neural architecture search (NAS) [12] to search optimal architectures, and achieves 3.2% higher top-1 accuracy than MobileNet-v2 on ImageNet [9]. EfficientNet [39] is another popular network for mobile devices, which mainly focuses on model scaling on convolutional neural networks.They design their baseline network and scale it up to get a series of models by using ConvNet scaling for three dimensions of network: width, depth, and resolutions. In our solution, we compare the performance of different compact student networks and submit the MobileNet-NNConv5 network architecture [40] as our final student model.

Running Deep Neural Networks on mobile devices Despite the computational power of mobile devices, e.g., smart phones, tablets, has increased rapidly, many applications based on deep neural networks are not executed on mobile devices directly. Because the computation overhead of unoptimized models is too large even for high-end devices. Thus, a constellation of attempts has been made for acceleration. CNNdroid [24] purposes a GPU-accelerated library for the CNNs. SoC manufacturers also develop their own SDKs for deep neural networks, e.g., SNPE [3] from Qualcomm, HiAI platform [1] from HiSilicon and NeuroPilot from MediaTek. [25]. In order to simplify the optimization process, TensorFlow Lite [4] provides a series of optimized kernels, pre-fused activations. Besides, TensorFlow Lite can convert the model pretrained by TensorFlow [5] to .tflite format, which can be easily used for inference on mobile devices. AI benchmark [21] comprehensively analyzes the acceleration techniques for mobile devices and establish a benchmark to evaluate the efficiency and performance of different chipsets. Readers can refer to [21] and [20] for more details.

3. Our method

Fig. 2 gives an overview of our method. Our method consists of two networks: a teacher network with strong capability and a compact student network. During training, we adopt 3 different loss functions: pairwise loss, gradient

matching loss and scale invariant loss. Under the supervision of the pairwise loss between the feature maps of two networks, the representation ability of our teacher network is transferred to the student network via knowledge distillation. Scale-invariant loss and gradient matching loss are adopted to measure the discrepancy between the estimated depth map and the ground truth depth map. During inference, only the light-weight student network are needed. The rest part of this section discusses each part in details.

3.1. Networks description

In this section, we introduce the architecture of our student network and teacher network in details. The two networks share similar architecture with an encoder and a decoder.

Student Network For student network, we adopt the same model architecture as FastDepth [40], which is designed for embedded systems. As shown in Fig. 3, the student network has a typical encoder-decoder structure with skip connections. We adopt MobileNet [18] as the backbone to extract features, which use depthwise and pointwise convolution to reduce the computation overhead and number of parameters.

The decoder gradually upsamples the feature maps extracted by the encoder and outputs an estimated density map which has the same resolution as the input image. As the encoder reduces the resolution by $32\times$, the decoder includes 5 blocks. In each block, the number of channels is halved and the spatial resolutions of feature maps are doubled. Like many methods designed for dense prediction, skip connections are used between the encoder and decoder, as they are conducive to combining the high-level semantic information with low-level image details.

Given an input image I, the student network output a estimated depth map as $\hat{d}_s = \mathcal{F}_s(I, \theta_s)$, where θ_s denotes the parameters of the student network, and $\hat{d}_s$ denotes the estimated depth map.

Teacher Network Similar to the student network, feature maps in the encoder are merged to the decoder with skip connections. We adopt the same feature fusion operation as Xian *et al*. [41], where the feature maps from the encoder are first processed by a residual convolution module and then merged into feature maps of the decoder by addition. The spatial resolution of the fused feature maps are doubled by bilinear interpolation. At the final stage, the feature maps are fed into an adaptive output module and the estimated depth maps are obtained. In practice, we test a series of backbone, and adopt ResNeSt-101 [42] as the encoder for its superior performance.

We denote the teacher network as $\mathcal{F}_t(I)$. $\mathcal{F}_t$ transfer the input image I into a depth map $\hat{d}_t$ by $\hat{d}_t = \mathcal{F}_t(I, \theta_t)$, where θ_t denotes the parameters of the teacher network.

3.2. Knowledge distillation

Knowledge distillation aims at transferring the representation ability of the teacher network to the student network. Targeting this, the student network are taught to output feature maps similar to the counterparts in the teacher network. We mainly refer to Pilzer *et al*. [33] and Liu *et al*. [30] to build our knowledge distillation strategy. To be specific, given the teacher network $\mathcal{F}_t$ and the student network $\mathcal{F}_s$, we extract a series of intermediate feature maps from $\mathcal{F}_t$ and $\mathcal{F}_s$, i.e., M_t from the teacher network and M_s from the student network. Then we select K pairs of feature maps $\{M_{t,i}, M_{s,i}\}, i = 1, 2, \cdots, K$, to construct a knowledge distillation framework.

During training, a pairwise distillation loss $\mathcal{L}_{pa}$ is designed as optimization target, which will be discussed in section 3.3. The total distillation loss function can be denoted by

$$\mathcal{L}_d = \sum_{i=1}^{K} \mathcal{L}_{pa}\left(M_{t,i}, M_{s,i}, \theta_s\right) \tag{1}$$

In practice, we train the teacher network on MAI2021 dataset and select a model with best performance on the validation set. When training the student network, the parameters of the teacher network are fixed and the feature maps are extracted to calculate the distillation loss $\mathcal{L}_d$. Only the parameters of student network are updated by optimizing the overall loss function 1.

3.3. Loss functions

Scale-invariant loss We adopt scale-invariant loss purposed by Eigen *et al*. [11] to measure the discrepancy between the output of the student network and the ground truth depth map without being disturbed by the difference in scales. Given the predicted depth map $d \in \mathbb{R}^{H\times W}$ and the ground truth depth map $d^* \in \mathbb{R}^{H\times W}$, each with n pixels indexed by i, scale-invariant loss $\mathcal{L}_s$ is defined as

$$\mathcal{L}_s\left(d, d^*, \theta_s\right) = \frac{1}{n}\sum_i g_i^2 - \frac{1}{n^2}(\sum_i g_i)^2, \tag{2}$$

Where $g_i = \log d_i - \log d_i^*$. g_i denotes the pixel-wise difference between the predicted and ground truth depth map in log space.

Gradient matching loss Only applying scale-invariant loss is not enough to train a network which outputs high-quality depth maps. Thus, gradient matching loss [35] is adopted to force the network to output depth map with sharp edges and smooth internal areas, which coincide with the ground truth depth maps. Gradient matching loss $\mathcal{L}_{gml}$ is

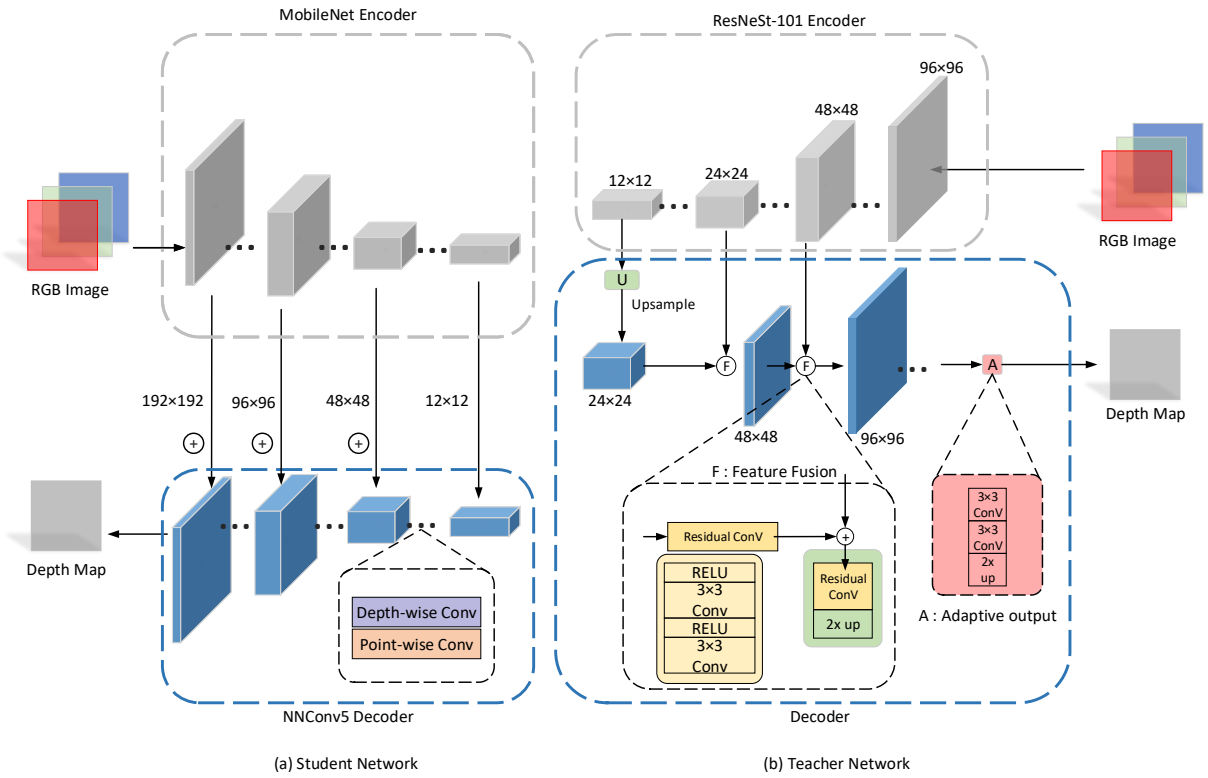

Figure 3. The architecture of student network and teacher network. (a) The network architecture of our light-weight student model. We use MobileNet as encoder to extract features. The decoder contains 5 layers that consists of depthwise separable convolution to half the number of channels and bilinear interpolation to double the spatial resolution. The skip connection is used to recover the detailed information. (b) The architecture of our teacher model. We select ResNeSt-101 as encoder as its strong feature extraction capability. As for its decoder, we mainly adopt the feature fusion block and the adaptive output block to recover the depth map from feature maps obtained by the encoder.

defined as:

$$\mathcal{L}_{gml}\left(d, d^*, \theta_s\right) = \frac{1}{M}\sum_{k=1}^{K}\sum_{i=1}^{M}(|\nabla_x^k W_x^k| + |\nabla_y^k W_y^k|), \quad (3)$$

where ∇_x^k and ∇_y^k denotes the gradient of the prediction results in x and y direction, respectively. We select 4 different levels of depth map resolution indexed by k. For each resolution, we calculate the gradient of ground truth and the prediction results and convert the gradient of ground truth in x and y direction to weight W_x^k and W_y^k.

Pairwise distillation loss Pairwise distillation loss [30] supervises the knowledge transfer process by comparing the feature maps of the teacher network and the student network. Assume that we have a pair of feature maps, including feature maps $m_t \in \mathbb{R}^{h\times w\times c_1}$ from the teacher net and feature maps $m_s \in \mathbb{R}^{h\times w\times c_2}$ from the student net. Note that these two feature maps must have the same spatial resolution. The pairwise distillation loss $\mathcal{L}_{pa}$ is calculated in two steps. First, affinity maps are built for these two feature maps. Then, in the second step, we compute the mean square error between the affinity maps of the two feature maps as defined by

$$\mathcal{L}_{pa}(m_s, m_t) = \frac{1}{w\times h}\sum_i\sum_j(a_{ij}^s - a_{ij}^t)^2, \quad (4)$$

where a_{ij} denotes the element in the affinity maps.

Overall loss function The over all loss function can be defined as follows:

$$\begin{aligned}\mathcal{L}(I, d^*, \theta_s) =& \lambda_s \cdot \mathcal{L}_s\left(d, d^*, \theta_s\right) + \lambda_{gml} \cdot \mathcal{L}_{gml}(d, d^*, \theta_s) \\ &+ \lambda_d \cdot \mathcal{L}_d(M_t, M_s, \theta_s),\end{aligned} \quad (5)$$

where λ_s, λ_{gml} and λ_{pa} are pre-defined weights for different loss functions.

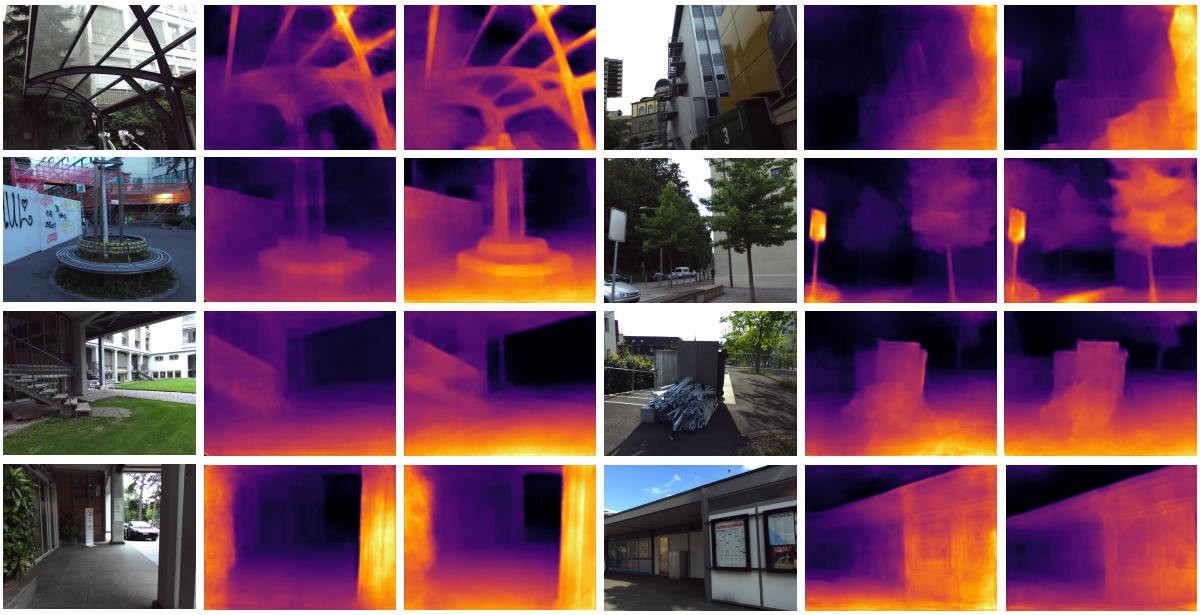

Figure 4. The visualizations of our method. For each image, from left to right alternating: input RGB image, the prediction result of our student model and the result of teacher model.

4. Experiments

In this section, we introduce our experiments and analyze their results to demonstrate the effectiveness of our solution. First, we define the evaluation metrics and elaborate the dataset. Then the detailed results of ablation study and accuracy evaluation are given. We also evaluate the inference time on mobile devices to show that our method can obtain high-throughput depth estimation on mobile devices.

4.1. Experiments setup

Dataset The MAI2021 Monocular Depth Dataset is proposed by MAI2021 Monocular Depth Estimation Challenge. It includes 7385 RGB images of outdoor scene and the corresponding depth maps. The pixel values of depth maps are in uint16 format ranging from 0 to 40000, which represent depth values from 0 to 40 meters. In our experiments, we randomly select 6725 images as the training set and the other 660 images as the validation set to evaluate the performance of our models.

Evaluation metrics We use si-RMSE, RMSE and rel to evaluate the accuracy of estimated depth map. If we denote d_i as the prediction depth map and d_i^* as the ground-truth, these three metrics are defined as:

$$g_i = \log d_i - \log d_i^*$$

$$\text{si-RMSE} = \sqrt{\frac{1}{n}\sum_i (g_i)^2 - \frac{1}{n^2}\left[\sum_i (g_i)\right]^2}, \quad (6)$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (d_i - d_i^*)^2}, \quad (7)$$

$$\text{rel} = \frac{1}{n}\sum_i \frac{|d_i - d_i^*|}{d_i^*}. \quad (8)$$

4.2. Quantitative results

Our method rank the second in Mobile AI 2021 monocular depth estimation challenge. The preliminary competition results are shown in Tab. 1. The username of our team is KX_SMART. Teams are sorted according the submission score, which takes both the accuracy and the inference time into account. The inference time is tested on a Raspberry Pi 4. Our method achieves a moderate balance between accuracy and inference time.

4.3. Ablation study

4.3.1 Effectiveness of knowledge distillation

In this section, we show how the knowledge distillation strategy improves the performance of the student network. As shown in Tab. 2, baseline means the student network MobileNet-NNConv5 is directly trained on the training set with scale-invariant loss and gradient matching loss and the

Table 1. The result of Mobile AI 2021 monocular depth estimation challenge.

Username	si-RMSE	RMSE	rel	Runtime (ms)	Submission Score
Parkzyzhang	0.2836	3.56	0.2690	97	129.41
KX_SMART	0.2602	3.25	0.2678	1197	14.51
dujinhua	0.2408	3.00	0.2389	1933	11.75
root12321	0.2449	3.02	0.2648	2130	10.08
Jacob.Yao	0.2902	3.91	0.4700	1275	8.98
helloworld3	0.3128	3.89	0.3228	958	8.74
jey	0.2761	9.68	0.9951	2531	5.5
zhyl	0.2332	2.72	0.2189	6146	4.11
weichi	0.4659	7.56	0.5992	582	1.72
shayanj	0.3543	4.16	0.3862	3466	1.36
fanhuanhuan	0.2678	5.96	0.5152	26494	0.59
faustChok	0.3737	9.08	0.8573	9392	0.38

baseline+KD means the same model is trained using scale-invariant loss, gradient matching loss and knowledge distillation. The knowledge distillation framework improved the si-RMSE from 0.295 to 0.281.

Table 2. With/without knowledge distillation

Method	si-RMSE	RMSE	rel
baseline	0.295	4.042	0.276
baseline+KD	**0.282**	**3.951**	**0.263**

4.3.2 Comparison of different teacher networks

To find a teacher network with higher performance, we compare the performance of teacher network with different backbones. As shown in Tab. 3, we evaluate ResNeSt-101 [42], EfficientNet-B7 [39], and ResNet-101 [13]. According to the result, we choose ResNeSt-101 as the backbone of our teacher network, which achieves the best accuracy on the MAI2021 dataset with a si-RMSE of 0.242 and a rel of 0.223.

Table 3. Teacher networks with different backbones

Teacher Net	si-RMSE	RMSE	rel
ResNet-101	0.340	5.720	0.329
EfficientNet-B7	0.264	5.829	0.235
ResNeSt-101	**0.242**	**3.545**	**0.223**

4.3.3 Comparison of different student networks

In this section, we compare the performance of different backbones on the student networks. We keep the teacher model and distillation strategy fixed and compare the accuracy of different student networks. The results are shown in Tab. 4. MobileNet-NNConv5 obtains the highest accuracy among all the models, with a si-RMSE of 0.282 and a rel of 0.263.

Table 4. Different backbones on the student networks.

Student Net	si-RMSE	RMSE	rel
MobileNet-V2	0.330	5.81	0.485
EfficientNet-B0	0.295	3.996	0.267
MobileNet	**0.282**	**3.951**	**0.263**

4.4. Inference time

To verify that our method can achieve high-throughput monocular depth estimation on mobile devices. We convert our model to TensorFlow-Lite and test the inference time on a smartphone with multiple processors include Snapdragon 888 andKirin 970 . We test the model using AI Benchmark [20]. The resolution of input and output images is 640×480, and the data type is float (32 bit). As shown in Tab. 5, our student network with MobileNet as backbone can obtain high-throughput inference with about 20 fps on smart phones with Snapdragon 888 processor.

Table 5. Inference time of our student network(AI Benchmark)

SoC	Device	Average/ms	STD/ms
Snapdragon 888	CPU	102.00	4.02
Snapdragon 888	GPU Delegate	46.90	0.79
Kirin 970	CPU	261.00	4.20
Kirin 970	GPU Delegate	76.80	5.84

5. Conclusion

We have introduced knowledge distillation for fast and accurate depth estimation on mobile devices. By knowledge distillation, we transfer the knowledge from a stronger teacher network to a lightweight student network, maintaining the performance while reducing the inference time. Exhaustive visualizations and ablation studies are presented to validate and demonstrate the effects of our method.

References

[1] Huawei hiai engine introduction. https://developer.huawei.com/consumer/en/doc/2020315. Retrieved on: 30.09.2018. 3

[2] Mobile ai 2021 monocular depth estimation challenge. https://competitions.codalab.org/competitions/28122#learn_the_details. 2

[3] Snapdragon neural processing engine sdk. https://developer.qualcomm.com/docs/snpe/overview.html. Retrieved on: 30.09.2018. 3

[4] Tensorflow lite: Deploy machine learning models on mobile and iot devices. https://www.tensorflow.org/lite. Retrieved on: 17.04.2021. 3

[5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016. 3

[6] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 742–751, 2017. 2

[7] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 1

[8] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 3

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[10] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 2

[11] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283*, 2014. 2, 4

[12] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, et al. Neural architecture search: A survey. *J. Mach. Learn. Res.*, 20(55):1–21, 2019. 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 7

[14] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017. 1, 2

[15] Majed El Helou, Ruofan Zhou, Sabine Süsstrunk, Radu Timofte, Mahmoud Afifi, Michael S Brown, Kele Xu, Hengxing Cai, Yuzhong Liu, Li-Wen Wang, et al. Aim 2020: Scene relighting and illumination estimation challenge. *arXiv preprint arXiv:2009.12798*, 2020. 1

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[17] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019. 3

[18] 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. *arXiv preprint arXiv:1704.04861*, 2017. 1, 3, 4

[19] Andrey Ignatov, Grigory Malivenko, David Plowman, Samarth Shukla, and Radu Timofte. Fast and accurate single-image depth estimation on mobile devices, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2021. 2

[20] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 3, 7

[21] Andrey Ignatov, Radu Timofte, Andrei Kulik, Seungsoo Yang, Ke Wang, Felix Baum, Max Wu, Lirong Xu, and Luc Van Gool. Ai benchmark: All about deep learning on smartphones in 2019. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3617–3635. IEEE, 2019. 3

[22] Andrey Ignatov, Radu Timofte, Thang Van Vu, Tung Minh Luu, Trung X Pham, Cao Van Nguyen, Yongwoo Kim, Jae-Seok Choi, Munchurl Kim, Jie Huang, et al. Pirm challenge on perceptual image enhancement on smartphones: Report. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018. 1

[23] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 2

[24] Seyyed Salar Latifi Oskouei, Hossein Golestani, Matin Hashemi, and Soheil Ghiasi. Cnndroid: Gpu-accelerated execution of trained deep convolutional neural networks on android. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1201–1205, 2016. 3

[25] Yen-Lin Lee, Pei-Kuei Tsung, and Max Wu. Techology trend of edge ai. In *2018 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pages 1–2. IEEE, 2018. 3

[26] Ruibo Li, Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, and Lingxiao Hang. Deep attention-based classification network for robust depth prediction. In *Asian Conference on Computer Vision*, pages 663–678. Springer, 2018. 2

[27] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2041–2050, 2018. 2

[28] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5162–5170, 2015. 2

[29] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2604–2613, 2019. 2

[30] Yifan Liu, Changyong Shun, Jingdong Wang, and Chunhua Shen. Structured knowledge distillation for dense prediction. *arXiv preprint arXiv:1903.04197*, 2019. 2, 4, 5

[31] Xianrui Luo, Juewen Peng, Ke Xian, Zijin Wu, and Zhiguo Cao. Bokeh rendering from defocus estimation. In *European Conference on Computer Vision*, pages 245–261. Springer, 2020. 1

[32] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 2

[33] Andrea Pilzer, Stephane Lathuiliere, Nicu Sebe, and Elisa Ricci. Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9768–9777, 2019. 1, 2, 4

[34] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. *arXiv preprint arXiv:1802.05668*, 2018. 2

[35] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*, 2019. 2, 4

[36] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 1, 3

[37] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Learning 3-d scene structure from a single still image. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007. 2

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[39] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 3, 7

[40] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. Fastdepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108. IEEE, 2019. 2, 3, 4

[41] Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, Yang Xiao, Ruibo Li, and Zhenbo Luo. Monocular relative depth perception with web stereo data supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 4

[42] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R. Manmatha, Mu Li, and Alexander Smola. Resnest: Split-attention networks, 2020. 4, 7

[43] Kai Zhang, Martin Danelljan, Yawei Li, Radu Timofte, Jie Liu, Jie Tang, Gangshan Wu, Yu Zhu, Xiangyu He, Wenjie Xu, et al. Aim 2020 challenge on efficient super-resolution: Methods and results. *arXiv preprint arXiv:2009.06943*, 2020. 1