Promoting CNNs with Cross-Architecture Knowledge Distillation for Efficient Monocular Depth Estimation

Zhimeng Zheng Zhejiang University Hangzhou, China 22110145@zju.edu.cn

Gongsheng Li Zhejiang University Hangzhou, China 22210102@zju.edu.cn

ABSTRACT

Recently, the performance of monocular depth estimation (MDE) has been significantly boosted with the integration of transformer models. However, the transformer models are usually computationallyexpensive, and their effectiveness in light-weight models are limited compared to convolutions. This limitation hinders their deployment on resource-limited devices. In this paper, we propose a cross-architecture knowledge distillation method for MDE, dubbed DisDepth, to enhance efficient CNN models with the supervision of state-of-the-art transformer models. Concretely, we first build a simple framework of convolution-based MDE, which is then enhanced with a novel local-global convolution module to capture both local and global information in the image. To effectively distill valuable information from the transformer teacher and bridge the gap between convolution and transformer features, we introduce a method to acclimate the teacher with a ghost decoder. The ghost decoder is a copy of the student's decoder, and adapting the teacher with the ghost decoder aligns the features to be studentfriendly while preserving their original performance. Furthermore, we propose an attentive knowledge distillation loss that adaptively identifies features valuable for depth estimation. This loss guides the student to focus more on attentive regions, improving its performance. Extensive experiments on KITTI and NYU Depth V2 datasets demonstrate the effectiveness of DisDepth. Our method achieves significant improvements on various efficient backbones, showcasing its potential for efficient monocular depth estimation.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; *Machine learning approaches.*

KEYWORDS

Monocular Depth Estimation, Cross-architecture Knowledge Distillation, Computational Efficiency, Local-Global Convolution Module

1 INTRODUCTION

Monocular depth estimation(MDE) [3, 9] has achieved remarkable success by the adventure of automatic feature engineering in deep

Tao Huang* The University of Sydney Sydney, Australia thua7590@uni.sydney.edu.au

> Zuyi Wang Zhejiang University Hangzhou, China zuyiwang@zju.edu.cn



Figure 1: RMSE (lower is better) and FLOPs comparisons of existing MDE models and our DisDepth models on KITTI dataset. DisDepth variants (green squares) obtain competitive performance with significant superiority on efficiency.

neural networks. With the advent of better global feature representation in vision transformers [41], recent state-of-the-art MDE methods resort to complicated transformer-based backbones [1, 27, 28] and decoder structures [1, 4], which are computational-expensive and hard to be deployed on resource-constrained devices. For example, PixelFormer [1] with Swin-L backbone has ~704B FLOPs and ~271M parameters, which is too heavy for edge devices to inference.

In this paper, we aim to obtain competitive performance while maintaining an efficient inference of MDE model by exploring a CNN-based framework without transformers. This is achieved by proposing (1) an efficient CNN-based MDE framework and (2) a novel cross-architecture knowledge distillation (KD) method that adjusts and distills the transformer features into CNNs. Concretely, as shown in Figure 2, we first construct a simple and efficient CNNbased MDE framework that contains a CNN backbone, a simple decoder with convolutions and upsamplings, and two heads for predicting bin centers and bin probabilities [4]. Then, a local-global convolution (LG-Conv) module is proposed to enhance the capability of global representations in CNNs. Our LG-Conv is inspired by our analysis that the role of self-attention in transformers is to aggregate and broadcast the information globally among tokens, while this global information exchange can also be done with simple

^{*}Corresponding author.

Preprint, ,

convolution and pooling operations. The resulting global branch in LG-Conv is a supplement of the original local convolution, and can be directly utilized on a pretrained backbone without retraining on pretraining dataset. Our LG-Conv is efficient and friendly to deployment, and experiments show that it effectively extracts the global information and boosts the performance.

Besides, we propose a new KD method for cross-architecture distillation from transformers to CNNs. Our basic motivation is to leverage the state-of-the-art transformer-based MDE models to improve our CNN models with KD. However, we empirically find that due to capacity and architecture gaps, the student is difficult to imitate teacher's features, and result in poor distillation performance, and even worse than the CNN teacher with significant worse performance. To understand what restricts the distillation performance, we conduct experiments and find that the devil of cross-architecture KD is in the architecture-intrinsic information¹. Therefore, we intuitively decouple the information in teacher feature into two parts, namely intrinsic information and extrinsic information. The intrinsic information is architecture-independent. and small students with different architectures are arduous in learning this information. While the extrinsic information is crucial for generating the task predictions, and is common and invariant to different architectures; it can be easily adapted and distilled to students.

To this end, we propose to acclimate the transformer teacher with a ghost decoder, which is a copy of the student's decoder, so that we can obtain adapted teacher features that are more appropriate for distillation. Specifically, we inject a feature acclimation module (FAM, a transformer block) into the teacher backbone after each output layer, which is used to acclimate the teacher features. The resulting adapted features are fed into the ghost decoder to get the predictions, which are then passed to the task loss function. This error between prediction and ground-truth is used to optimize the FAM, and other modules including original backbone and ghost decoder are fixed. Through this adaptation, the intrinsic information in teacher features is repressed and task-relevant extrinsic information is acclimated, as the student-trained decoder prefers student-like features. Furthermore, instead of directly optimizing the distillation loss between adapted teacher feature and student feature, we introduce a attentive KD loss, which learns valuable regions in the teacher features, and then use the learned region importances to guide the student to focus more on valuable features.

In sum, our contribution is threefold.

- (1) We propose a simple CNN framework for MDE, with a supplementary local-global convolution module to enhance the global representations. Our framework achieves competitive performance and is more efficient than current state-of-theart transformer-based frameworks.
- (2) We propose a novel cross-architecture knowledge distillation method, which effectively represses the architectureintrinsic information and adapts task-relevant information in the transformer teacher features for better distillation to

CNN student. Additionally, an attentive KD loss is introduced to learn and distill valuable regions in teacher features.

(3) Extensive experiments on KITTI and NYU Depth V2 datasets demonstrate our effectiveness. As shown in Figure 1, our DisDepth achieves significant efficiency and performance superiorities on various model scales.

2 RELATED WORK

2.1 Monocular Depth Estimation

Monocular depth estimation (MDE) is an active area of research that aims to regress a dense depth map from a single RGB image. The pioneer, [9], which initiated the trend with its coarse-to-fine network and innovative scale-invariant log loss (SIlog) for optimisation. Building on this foundation, the field embraced architectural advances such as residual networks [23], multi-scale fusion [24], and transformers [4, 44]. On the other hand, some works are proposed to improve the optimization techniques, including reverse Huber loss [23], classification-based schemes [6], and ordinal regression [10]. The field has also seen a move towards ordinal regression methods, which are similar to mask-based segmentation in their approach to depth discretisation. By associating each depth value with a confidence mask, the culmination in depth prediction becomes a weighted interplay of these discretised values.

2.2 Knowledge Distillation

Knowledge distillation (KD) [7, 14, 16, 18, 20, 43, 45, 46] is an idea seeded in the realm of image recognition by [16], it serves as a conduit to transfer knowledge from a high-capacity teacher model to a more compact student model. Its versatility has been affirmed with its myriad applications across various computer vision tasks, including domain adaptation [14], object detection [7], and learning from noisy labels [26]. In the MDE context, [32] and [2] spearheaded efforts to harness the power of distillation, setting the stage for subsequent innovations. The application of distillation in depth estimation for resource-constrained environments, particularly mobile systems, is of great significance [42], [33], [43]. The challenge is balancing depth estimation accuracy with computational efficiency, crucial for real-time applications like autonomous driving. This balance becomes even more pronounced in bespoke hardware setups, like dual-pixel sensors [11]. While the journey has begun, the landscape of knowledge distillation in MDE, especially under constraints, remains a dynamic and evolving field.

3 OUR APPROACH

In this section, we present our method DisDepth. We begin by discussing the design of our CNN-only MDE framework. We then elaborate on our local-global convolution, which is designed to enhance the global representations of the model. Additionally, we introduce our cross-architecture knowledge distillation approach, where we adapt the transformer teacher with a ghost decoder. Finally, we propose an attentive knowledge distillation loss that helps identify valuable regions for the student model to focus on.

¹For a brief, we use a fixed and pretrained student decoder to adapt the teacher features, and the teacher still obtains similar performance, while a significant gain on distillation performance has been obtained.

Preprint, ,

DisDepth



Figure 2: Framework of our cross-architecture KD. We introduce (feature acclimation module) FAMs to adapt the outputs of teacher encoder, followed by a (loss attention module) LAM to identify valuable regions in the features, and these modules are optimized with a ghost decoder and task loss. The distillation is conducted on student features and the adapted teacher features. The parameters of teacher encoder and ghost decoder are fixed.

3.1 Frustratingly Simple CNN Framework

To improve the performance of MDE, recent state-of-the-art approaches have often relied on complex and heavy decoder structures. For instance, AdaBins [4] proposes a mViT module with multiple transformer layers to enhance the decoded feature pyramid. DepthFormer [27] introduces an HAHI module that combines deformable cross-attention and self-attention to bridge the encoder and decoder. PixelFormer [1] utilizes a UNet structure with advanced feature fusion modules, including window cross-attention. However, these sophisticated decoder designs come at the cost of reduced efficiency and deployment challenges. The heavy computational requirements and complexity make it difficult to deploy these models in real-world scenarios.

In our research, we have discovered that by using a powerful backbone encoder, a simple CNN-based decoder can still achieve competitive performance in MDE. This finding has led us to develop a surprisingly simple CNN framework. For instance, in Table 5, our framework achieves an RMSE of 2.058 on the KITTI dataset when using the Swin-Large backbone. In comparison, the PixelFormer model, which employs an advanced decoder, achieves an RMSE of 2.081. This observation suggests that we can effectively replace complex and advanced decoder approaches with a simpler CNNbased decoder, allowing us to focus primarily on improving the encoder. This insight opens up new possibilities for developing efficient and effective MDE models that strike a balance between performance and simplicity.

Implementation details. Our framework is illustrated in Figure 2 (see the bottom-half student model). With an input image, it first encodes its features using an encoder (backbone), then passes the features of the last four stages to get the feature pyramid. A simple convolutional and upsampling decoder [3] is employed to fuse and extract features from the feature pyramid. The decoder

combines information from different scales to generate a final feature map. Following the approach introduced in AdaBins [4], the feature map is passed through two heads. One head predicts the bin centers, while the other head predicts the bin probabilities. The predicted bin centers and probabilities are further processed to obtain the final predicted depth map.

3.2 Enhanced CNNs with Local-Global Convolutions

Compared to CNNs, transformers [41] are acknowledged to have better global representation capability due to the global self-attention. Some recent methods [27] have successfully shown that transformers can significantly boost the MDE performance due to better representation power and relationship modeling. However, the computation of self-attention requires has a large peak memory footprint and has not been widely optimized and supported by edge computation frameworks, which have achieved remarkable success by deploying CNNs into real-life productions (see supplementary material for details).

To enhance the CNNs with global dependency, we propose a convolution-based module named local-global convolution (LG-Conv), which works as a supplement of the original local convolution layers. Different from recent approaches in bringing long-term dependency into CNNs, which focus on mixing transformers and convolutions into one block or design complicated computation scheme that cannot be efficiently deployed, our LG-Conv is composed with only convolutions and poolings, and has the following superiorities. (1) Our LG-Conv does not require retraining the backbone. (2) Our LG-Conv is efficient. (3) Our LG-Conv is deployment-friendly.

Implementation details. As shown in Figure 3, which a pretrained CNN backbone composed of multiple convolution layers,



Figure 3: Illustration of our proposed local-global convolution (LG-Conv).

we expand each 3×3 convolutions with an additional global branch in LG-Conv. The global branch takes the same input feature as the original local convolution. The feature is first transformed using a Conv-ReLU structure. Then, the hidden feature is obtained by concatenating the local feature with the global feature, which is obtained through average pooling. We utilize the hidden feature to predict multi-head attentions over the spatial dimension, which are then used to aggregate the pixel features into one feature. Additionally, we introduce a broadcast branch to predict probabilities that determine whether the aggregated global information should be applied to each pixel. To ensure that the pretrained semantic information is not disturbed by directly supplementing the global branch's feature to the original feature, we propose initializing the scale parameters y in the last batch normalization layer with a small value (e.g., 0.001). This initialization ensures that the global branch has minimal impact on the original features during the initial stages of training, allowing the backbone to evolve smoothly.

3.3 Acclimating Teacher with A Ghost Decoder

Among existing methods, the transformer-based models outperform CNN-based models by a large margin. Therefore, to enhance our efficient model, an intuitive idea is to use the state-of-the-art transformer models as the teacher to distill our model (student). However, we empirically find that directly imitating the features of transformer teacher does not gains significant improvements as expected. For instance, as summarized in Table 3, we train our EfficientNet-B0 student with EfficientNet-B5 and Swin-L teachers, and show that, although the Swin-L has an obvious superiority in performance (2.06 RMSE vs. 2.60 RMSE), its distillation performance (2.95 RMSE) is largely behind the EfficientNet-B5 teacher (2.78 RMSE), which has a similar architecture to the student.

To better understand the distinction between CNN features and transformer features, we conduct experiments and make an interesting observation. By using a fixed student-trained decoder to replace the original decoder in the pretrained teacher model, and optimizing the teacher with task loss, we find that the new teacher model achieves similar performance compared to the original teacher. This trial successfully acclimate the teacher features to be similar to the student's features, while still preserving the task-relevant information necessary for good performance.

Algorithm 1 Cross-architecture KD with DisDepth

Require: Student model S, pretrained teacher model T, train dataset D_{tr} , and number of iterations *I*.

- 1: Integrate FAMs into teacher model
- 2: **for** iteration *i* in 1, ..., *I* **do**
- 3: Get a batch of input *X* and target \hat{Y} from \mathcal{D}_{tr}
- 4: Copy the weights of student decoder to ghost decoder
- 5: Compute the outputs $F^{(t)}$ of teacher encoder with X
- 6: $\tilde{F}^{(t)} \leftarrow \text{LAM}(\text{FAM}(F^{(t)}))$
- 7: Get predictions $Y^{(t)}$ with ghost decoder
- 8: Compute features $F^{(s)}$ and predictions $Y^{(s)}$ of student
- 9: Optimize student with $\mathcal{L}_{task}(Y^{(s)}, \hat{Y})$ and $\mathcal{L}_{kd}(F^{(s)}, \tilde{F}^{(t)})$
- 10: Optimize FAMs and LAMs with $\mathcal{L}_{task}(Y^{(t)}, \hat{Y})$
- 11: end for
- 12: return Trained student model S^*

Based on this observation, we propose a hypothesis that the teacher features can be decoupled into two types of information: architecture intrinsic information and extrinsic information. The architecture intrinsic information is architecture-independent, and the student model finds it challenging to mimic this information due to its smaller capacity and architectural differences. On the other hand, the extrinsic information is task-relevant, and it is common and consistent across different architectures. These features can be easily adapted and imitated by the student model.

Consequently, to achieve effective and student-friendly crossarchitecture distillation, we propose acclimating the transformer teacher model with the student's decoder and distilling knowledge from the acclimated features. This process is illustrated in Figure 2. To facilitate this acclimation, we introduce a ghost decoder, which is a continuous copy of the student's decoder throughout the entire training period. The original teacher features are adjusted using additional trainable feature acclimation modules (FAMs), the ghost decoder, and task loss, which is a transformer block in the teacher model. Subsequently, we perform distillation between the acclimated features of the teacher model and the features of the student model. The iterative procedure of our DisDepth is summarized in Algorithm 1. Note that the FAMs are trained simultaneously with the student model, at the early of training, the student decoder is not converged and the acclimated features are with low performance, so we add a warmup that starts distillation after a few epochs trained.



Figure 4: Architecture of loss attention module (LAM).

3.4 Attentive Knowledge Distillation

We now have successfully built our method of acclimating teacher features. In feature distillation, how to select valuable features from the dense feature map for effective distillation is also a crucial problem [13, 19]. Considering that the teacher features are acclimated with the task loss, we can naturally get the importance of features by injecting a attention module between acclimated teacher feature and decoder. The attention module is also optimized with task loss. If a region is important to the performance, its attention weights should be large; in contrast, if a feature is useless to the prediction, we just multiply a zero weight to remove it. Therefore, our proposed loss attention module (LAM), is designed with a cross-attention layer. As illustrated in Figure 4, LAM uses the average of input feature as the query to capture the importance of each pixel, then the output feature is weighted by the importance scores. Besides, the scores, which represents the importances of each pixel, is also returned to compute the distillation loss.

Attentive distillation loss. Given the same image as input, we have the output features $F_l^{(t)} \in \mathbb{R}^{C \times H \times W}$ and $F_l^{(s)} \in \mathbb{R}^{C \times H \times W}$ of the teacher and student backbones, where C, H, W denote the numbers of channels, feature height, feature width, respectively, and l = 1, ..., L is the index of last L = 4 stages of backbone. Our attentive KD first generates the spatial attentions A_l using the teacher features, then these attentions representing the importance of each pixel to the depth predictions, are multiplied onto the teacher and student features to guide the student focus on imitating the important regions of teacher features. The attentive KD loss is formulated as

$$\mathcal{L}_{\rm kd} := \sum_{l=1}^{L} ||A_l(F_l^{(s)} - F_l^{(t)})||_2^2.$$
(1)

Task loss. Following previous works [1], we use a scaled version of the Scale-Invariant loss (SILog) [9] as our task loss. With the ground-truth depth d_i^* and the predicted depth \hat{d}_i at each pixel index of *n* pixels, we first calculate the logarithmic distance $g_i =$

 $\log(\hat{d}_i) - \log(d_i^*)$, then the SILog is computed as

$$\mathcal{L}_{\text{SILog}} = \alpha \sqrt{\frac{1}{n} \sum_{i=1}^{n} g_i^2 - \frac{\beta}{n^2} \left(\sum_{i=1}^{n} g_i\right)^2},$$
 (2)

we set $\alpha = 10$ and $\lambda = 0.85$ in all our experiments.

Overall loss. As a result, our overall loss is the summation of task loss \mathcal{L}_{SILog} , teacher's task loss \mathcal{L}_{SILog} , and attentive KD loss \mathcal{L}_{kd} , *i.e.*,

$$\mathcal{L} = \mathcal{L}_{\text{SILog}} + \mathcal{L}_{\text{SILog}}^{(t)} + \lambda \mathcal{L}_{\text{kd}}, \tag{3}$$

where λ is the factor for balancing loss terms.

4 EXPERIMENTS

Datasets. Following previous works, we use two popular benchmark datasets KITTI [12] and NYU Depth V2 [37] to validate our efficacy on outdoor and indoor depth estimations, respectively.

Models. The evaluate our method sufficiently, we conduct experiments with various backbone variants, including EfficientNet [39], GhostNet [15], ResNet [40], and Swin Transformer [29].

Evaluation Metrics. We use standard metrics following previous studies [4], such as average relative error (Abs Rel), root mean squared error (RMSE), and average log error (log10). In addition, we use threshold accuracy (δ_i) at thresholds δ = 1.25, 1.25², 1.25³ to compare our method to state-of-the-art methods used in earlier works. For KITTI evaluation, we additionally use square relative error (Sq Rel).

Training strategies. The DisDepth is implemented in Pytorch [30].We use Adam optimizer [22] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, with a batch size of 8 and weight decay of 10^{-2} . After a warmup of 7 epochs, distillation begins.We use 25 epoch for both KITTI and NYU-V2 dataset with an initial learning rate of 4×10^{-5} , which is decreased linearly to 4×10^{-6} across the training iterations. We use a similar test protocol as in [4, 44]. We set KD weight $\lambda = 0.05$ in all experiments.

Compared methods. We compare our DisDepth with recent state-of-the-art methods, including NeWCRFs [44], BTS [24], P3Depth [31], AdaBins [4], DepthFormer [28], BinsFormer [25], PixelFormer [1], and LightDepth [21].

4.1 Results on KITTI

On the Eigen split of the KITTI dataset, we compare several stateof-the-art depth estimation methods with our proposed family of DisDepth models, as shown in Table 1. We can see that, the methods with transformer backbones have a large FLOPs, parameters, and latency, while our models enjoy significant efficiency and performance superiority compared to existing methods. For example, our DisDepth obtains 2.546 RMSE with only 35.7B, surpassing the 42.8B FLOPs LightDepth by a large margin of 0.377. With ResNet-50 backbone, DisDepth-R50 obtains similar performance compared to state-of-the-art approaches PixelFormer, while saving ~ 30% FLOPs.

4.2 Results on NYU Depth v2

We summarize the results on NYU Depth v2 dataset in Table 2. Similar to the results on KITTI, our models achieve competitive

Method	Backbone	FLOPs (B)	Params (M)	Latency (ms)	Abs Rel↓	Sq Rel↓	RMSE↓	log10↓	$\delta_1 \uparrow$	$\delta_2\uparrow$	$\delta_3\uparrow$
DepthFormer	Swin-T	1256.6	273.7	112.1	0.052	0.158	2.143	0.079	0.975	0.997	0.999
BinsFormer	Swin-L	972.7	254.6	73.0	0.079	0.151	2.092	0.079	0.974	0.997	0.999
P3Depth	ResNet101	738.6	94.3	38.9	0.071	0.270	2.842	0.103	0.953	0.993	0.999
NeWCRFs	Swin-L	722.7	360.7	52.6	0.052	0.155	2.129	0.079	0.974	0.997	0.999
PixelFormer	Swin-L	703.7	270.9	46.2	0.051	0.149	2.081	0.077	0.976	0.997	0.999
AdaBins	MViT	481.6	78.3	36.0	0.058	0.190	2.360	0.088	0.964	0.995	0.999
BTS	ResNext101	474.6	112.8	28.6	0.059	0.241	2.756	0.096	0.956	0.993	0.998
LightDepth	DenseNet196	42.8	42.6	-	0.070	0.294	2.923	-	-	-	-
DisDepth-Swin-L	Swin-L (teacher)	800.0	237.0	47.1	0.050	0.144	2.058	0.076	0.976	0.997	0.999
DisDepth-B5	EfficientNet-B5	122.5	61.9	22.0	0.058	0.190	2.321	0.088	0.964	0.995	0.999
DisDepth-B3	EfficientNet-B3	61.1	23.1	12.2	0.060	0.201	2.366	0.096	0.957	0.995	0.999
DisDepth-B0	EfficientNet-B0	35.7	8.0	6.7	0.065	0.231	2.545	0.101	0.949	0.993	0.999
DisDepth-R50	ResNet-50	499.7	98.6	17.4	0.054	0.171	2.168	0.081	0.972	0.997	0.999
DisDepth-R18	ResNet-18	72.2	18.0	5.0	0.056	0.184	2.298	0.086	0.967	0.996	0.999

 Table 1: Performance comparisons on Eigen split od KITTI dataset. All the FLOPs and latencies are measured on a NVIDIA

 3090 GPU with 1120 × 352 input image size.LightDepth have no PyTorch implementation for speedtest.

Table 2: Performance comparisons on NYU-Depth-v2 dataset. All the FLOPs and latencies are measured on a NVIDIA 3090 GPU with 640×480 input image size.

Method	Backbone	FLOPs (B)	Params (M)	Latency (ms)	Abs Rel↓	RMSE↓	log10↓	$\delta_1 \uparrow$	$\delta_2\uparrow$	$\delta_3\uparrow$
DepthFormer	Swin-T	976.7	273.7	81.5	0.096	0.339	0.041	0.921	0.989	0.998
BinsFormer	Swin-L	756.6	254.6	56.3	0.094	0.330	0.041	0.925	0.989	0.997
P3Depth	ResNet101	575.5	94.5	29.7	0.104	0.356	0.043	0.898	0.981	0.996
NeWCRFs	Swin-L	561.2	270.4	40.6	0.095	0.316	0.041	0.922	0.992	0.998
PixelFormer	Swin-L	546.2	280.3	35.0	0.090	0.322	0.039	0.929	0.991	0.998
AdaBins	MViT	375.2	78.3	24.0	0.103	0.364	0.044	0.903	0.984	0.997
BTS	ResNext101	369.8	112.8	22.8	0.111	0.399	0.049	0.880	0.977	0.994
DisDepth-Swin-L	Swin-L (teacher)	621.3	237.0	35.5	0.090	0.338	0.041	0.920	0.991	0.998
DisDepth-B5	Efficientnet-B5	95.5	61.9	17.0	0.103	0.369	0.044	0.901	0.985	0.997
DisDepth-B0	Efficientnet-B0	27.8	8.0	5.3	0.125	0.398	0.048	0.872	0.977	0.995

performance with obvious reductions on FLOPs, parameters, and latency. For example, our DisDepth-B0 achieves similar performance with only 7.5% of FLOPs compared to BTS.

4.3 Ablation Study

Comparisons of student distilled by teacher with or without FA. In Table 3, we compare our method with the traditional KD method without feature acclimation (FA). The results show that, without feature acclimation, the cross-architecture distillation with Swin-L teacher obtains poor performance compared to EfficientNet-B5 teacher. While our FA can improve the distillation performance of both teachers, and the Swin-L teacher obtains better results since it is more advanced. This implies that the FA is crucial and effective for cross-architecture distillation.

Effects of the proposed components. We validate the effects of each of our proposed components in Table 4. (1) **LG.** The student model with our local-global convolution achieves a 0.18 improvement on RMSE. (2) **KD.** Compared to the independent training, the

Table 3: Comparisons of student distilled by teacher with or without feature acclimation (FA) on the KITTI dataset.

Teacher	Student	Abs Rel↓	Sq Rel↓	RMSE↓
Eff-B5	-	0.0662	0.2337	2.6020
Eff-B5	Eff-B0	0.0772	0.2896	2.7807
Eff-B5 w/ FA	Eff-B0	0.0717	0.2599	2.6559
Swin-L	-	0.0503	0.1448	2.0585
Swin-L	Eff-B0	0.0791	0.3320	2.9471
Swin-L w/ FA	Eff-B0	0.0650	0.2329	2.5446

classical KD method further brings ~ 0.20 decrease. (3) **FAM**. Our feature acclimation significantly improves the performance by 0.27. (4) **LAM**. Our complete method with loss attention module obtains the optimal performance.

Performance of ghost teacher. To show how the teacher's performance is affected by the ghost decoder, we report the performance of the original teacher and ghost teacher in Table 5. We

DisDepth

Table 4: Ablation study of the proposed Disdepth on the KITTI dataset. LG: Local-Global convolution; KD: knowledge distillation; FAM: feature acclimation module; LAM: loss attention module.

LG	KD	FAM	LAM	Abs Rel↓	Sq Rel↓	RMSE↓
				0.0895	0.4271	3.3232
\checkmark				0.0791	0.3896	3.1454
\checkmark	\checkmark			0.0743	0.3320	2.9471
\checkmark	\checkmark	\checkmark		0.0721	0.2659	2.6786
\checkmark	\checkmark	\checkmark	\checkmark	0.0650	0.2329	2.5446

can see that, the ghost decoder only has a minor degradation of the performance. This demonstrate our hypothesis that the taskrelevant extrinsic feature can be easily adapted and removing the architecture intrinsic feature would not affect the performance.

 Table 5: Performance of the teacher with ghost decoder and

 Swin-L backbone on the KITTI dataset.

Method	Abs Rel↓	Sq Rel↓	RMSE↓
Teacher	0.050	0.1440	2.0580
leacher w/ ghost decoder	0.053	0.1562	2.1406

Warmup epochs. Table 6 examines the performance of different warmup epochs. We obtain the optimal performance when the number of warmup epochs is set to 7.

Table 6: Effect of warmup epochs on the KITTI dataset

Warmup		Abs Rel↓	Sq Rel↓	RMSE↓
3	1	0.06876	0.2653	2.6917
5		0.06684	0.2394	2.6019
7		0.06501	0.2329	2.5446
9		0.06692	0.2339	2.5521
10		0.07077	0.2562	2.6204

Loss Weight. In Table 7, we conduct experiments to tune the KD loss weight λ . We can see that, the $\lambda = 0.05$ obtains the optimal performance.

Table 7: Ablation of loss weight λ on KITTI dataset.

Loss Weight	Abs Rel↓	Sq Rel↓	RMSE↓
0.01	0.07232	0.2641	2.6083
0.05	0.06501	0.2329	2.5446
0.1	0.06870	0.2537	2.5960

Efficiency comparisons of student with or without localglobal convolution. By incorporating Local-Global Convolutions (LG-Conv) into CNNs, there is a small rise in FLOPs, parameters, and latency, as shown in Table 8. Nevertheless, the benefit on performance is obvious. This indicates that LG-Conv is an effective tool for enhancing the capabilities of CNNs without imposing significant computational overloads.

 Table 8: Efficiency comparisons of student with or without

 Local-Global Convolutions on KITTI dataset.

Method	FLOPs (B)	Params (M)	Latency (ms)	RMSE↓
Eff-B0	30.4	5.9	5.2	3.3232
Eff-B0 w/ LG	35.7	8.0	6.7	3.1454

5 CONCLUSION

This work presents DisDepth, an advanced monocular depth estimation (MDE) method that combines the robustness of transformer models with the efficiency of CNN architectures. By incorporating a specially designed local-global convolution module, DisDepth effectively captures both fine-grained details and broader scene context. The introduction of a ghost decoder mechanism streamlines the knowledge transfer process from transformers to CNNs, ensuring digestible knowledge alignment. Additionally, the attentive KD loss focuses on the most valuable features, resulting in more accurate depth predictions. Experimental results on the KITTI and NYU Depth V2 datasets demonstrate DisDepth's superior performance, achieving a harmonious balance between depth accuracy and computational efficiency. DisDepth represents a significant advancement in MDE and holds promise for various applications in computer vision.

REFERENCES

- Ashutosh Agarwal and Chetan Arora. 2023. Attention attention everywhere: Monocular depth prediction with skip attention. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 5861–5870.
- [2] Filippo Aleotti, Giulio Zaccaroni, Luca Bartolomei, Matteo Poggi, Fabio Tosi, and Stefano Mattoccia. 2020. Real-time single image depth perception in the wild with handheld devices. *Cornell University - arXiv,Cornell University - arXiv* (Jun 2020).
- [3] Ibraheem Alhashim and Peter Wonka. 2018. High quality monocular depth estimation via transfer learning. arXiv preprint arXiv:1812.11941 (2018).
- [4] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. 2021. Adabins: Depth estimation using adaptive bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 4009–4018.
- [5] Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. 2021. Understanding and overcoming the challenges of efficient transformer quantization. arXiv preprint arXiv:2109.12948 (2021).
- [6] Yuanzhouhan Cao, Zifeng Wu, and Chunhua Shen. 2018. Estimating Depth from Monocular Images as Classification Using Deep Fully Convolutional Residual Networks. *IEEE Transactions on Circuits and Systems for Video Technology* (Nov 2018), 3174–3182. https://doi.org/10.1109/tcsvt.2017.2740321
- [7] Guobin Chen, Wongun Choi, Xiang Yu, TonyX. Han, and Manmohan Chandraker. 2017. Learning efficient object detection models with knowledge distillation. *Neural Information Processing Systems, Neural Information Processing Systems* (Dec 2017).
- [8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).
- [9] David Eigen, Christian Puhrsch, and Rob Fergus. 2014. Depth map prediction from a single image using a multi-scale deep network. Advances in neural information processing systems 27 (2014).
- [10] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. 2018. Deep Ordinal Regression Network for Monocular Depth Estimation. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. https: //doi.org/10.1109/cvpr.2018.00214
- [11] Rahul Garg, Neal Wadhwa, Sameer Ansari, and JonathanT. Barron. 2019. Learning Single Camera Depth Estimation using Dual-Pixels. Cornell University arXiv,Cornell University - arXiv (Apr 2019).
- [12] A Geiger, P Lenz, C Stiller, and R Urtasun. 2013. Vision meets robotics: The KITTI dataset. The International Journal of Robotics Research (Sep 2013), 1231–1237. https://doi.org/10.1177/0278364913491297
- [13] Jianyuan Guo, Kai Han, Yunhe Wang, Han Wu, Xinghao Chen, Chunjing Xu, and Chang Xu. 2021. Distilling object detectors via decoupled features. In Proceedings

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2154–2164.

- [14] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. 2016. Cross Modal Distillation for Supervision Transfer. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2016.309
- [15] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. 2020. Ghostnet: More features from cheap operations. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 1580–1589.
- [16] GeoffreyE. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv: Machine Learning, arXiv: Machine Learning (Mar 2015).
- [17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017).
- [18] Tao Huang, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2022. Knowledge distillation from a stronger teacher. Advances in Neural Information Processing Systems 35 (2022), 33716–33727.
- [19] Tao Huang, Yuan Zhang, Shan You, Fei Wang, Chen Qian, Jian Cao, and Chang Xu. 2022. Masked Distillation with Receptive Tokens. In *The Eleventh International Conference on Learning Representations*.
- [20] Tao Huang, Yuan Zhang, Mingkai Zheng, Shan You, Fei Wang, Chen Qian, and Chang Xu. 2024. Knowledge diffusion for distillation. Advances in Neural Information Processing Systems 36 (2024).
- [21] Fatemeh Karimi, Amir Mehrpanah, and Reza Rawassizadeh. 2022. LightDepth: A Resource Efficient Depth Estimation Approach for Dealing with Ground Truth Sparsity via Curriculum Learning. arXiv preprint arXiv:2211.08608 (2022).
- [22] DiederikP. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. arXiv: Learning, arXiv: Learning (Dec 2014).
- [23] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. 2016. Deeper Depth Prediction with Fully Convolutional Residual Networks. In 2016 Fourth International Conference on 3D Vision (3DV). https: //doi.org/10.1109/3dv.2016.32
- [24] JinHan Lee, MyungKyu Han, DongWook Ko, and IlHong Suh. 2019. From big to small: Multi-scale local planar guidance for monocular depth estimation. arXiv: Computer Vision and Pattern Recognition, arXiv: Computer Vision and Pattern Recognition (Jul 2019).
- [25] Li Li, Zhenyu, Wang Wang, Xuyang, Liu Liu, Xianming, Jiang Jiang, and Junjun. 2022. BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation. Cornell University - arXiv,Cornell University - arXiv (Apr 2022).
- [26] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. 2017. Learning from Noisy Labels with Distillation. In 2017 IEEE International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv.2017.211
- [27] Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. 2022. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. arXiv preprint arXiv:2203.14211 (2022).
- [28] Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. 2022. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. arXiv preprint arXiv:2203.14211 (2022).
- [29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF international conference on computer vision. 10012–10022.
- [30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch.(2017).
- [31] Vaishakh Patil, Christos Sakaridis, Alexander Liniger, and Luc Van Gool. 2022. P3depth: Monocular depth estimation with a piecewise planarity prior. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 1610–1621.
- [32] Andrea Pilzer, Stephane Lathuiliere, Nicu Sebe, and Elisa Ricci. 2019. Refine and Distill: Exploiting Cycle-Inconsistency and Knowledge Distillation for Unsupervised Monocular Depth Estimation. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). https://doi.org/10.1109/cvpr.2019.01000
- [33] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia. 2018. Towards Real-Time Unsupervised Monocular Depth Estimation on CPU. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/ 10.1109/iros.2018.8593814
- [34] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. 2023. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems* 5 (2023).
- [35] Rene Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. 2021. Vision Transformers for Dense Prediction. International Conference on Computer Vision, International Conference on Computer Vision (Jan 2021).
- [36] Brendan C Reidy, Mohammadreza Mohammadi, Mohammed E Elbtity, and Ramtin Zand. 2023. Efficient Deployment of Transformer Models on Edge TPU Accelerators: A Real System Evaluation. In Architecture and System Support for Transformer

Models (ASSYST@ ISCA 2023).

- [37] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. 2012. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV* 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12. Springer, 746–760.
- [38] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014).
- [39] Mingxing Tan and Quoc Le. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*. PMLR, 6105–6114.
- [40] Sasha Targ, Diogo Almeida, and Kevin Lyman. 2016. Resnet in resnet: Generalizing residual architectures. arXiv preprint arXiv:1603.08029 (2016).
- [41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. Advances in neural information processing systems 30 (2017).
- [42] Yan Wang, Zihang Lai, Gao Huang, Brian H. Wang, Laurens van der Maaten, Mark Campbell, and Kilian Q. Weinberger. 2019. Anytime Stereo Image Depth Estimation on Mobile Devices. In 2019 International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/icra.2019.8794003
- [43] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. 2019. FastDepth: Fast Monocular Depth Estimation on Embedded Systems.. In 2019 International Conference on Robotics and Automation (ICRA). https: //doi.org/10.1109/icra.2019.8794182
- [44] Weihao Yuan, Xiaodong Gu, Zuozhuo Dai, Siyu Zhu, and Ping Tan. 2022. New crfs: Neural window fully-connected crfs for monocular depth estimation. arXiv preprint arXiv:2203.01502 (2022).
- [45] Yuan Zhang, Weihua Chen, Yichen Lu, Tao Huang, Xiuyu Sun, and Jian Cao. 2023. Avatar knowledge distillation: self-ensemble teacher paradigm with uncertainty. In Proceedings of the 31st ACM International Conference on Multimedia. 5272–5280.
- [46] Yuan Zhang, Tao Huang, Jiaming Liu, Tao Jiang, Kuan Cheng, and Shanghang Zhang. 2023. FreeKD: Knowledge Distillation via Semantic Frequency Prompt. arXiv preprint arXiv:2311.12079 (2023).

A COMPARISONS OF DEPLOYING CNNS AND TRANSFORMERS IN EDGE COMPUTATION

A.1 Challenges and Main Difficulties of Deploying Transformers

Transformers, with their large number of parameters in architectures such as Vision Transformers (ViT) [35] and BERT-like [8] Transformers, present a challenge in edge deployments due to their large model sizes. This size, combined with the inherent computational complexity, especially with the self-attention mechanism, demands more power and resources. Such requirements can lead to rapid battery drain, especially in real-time applications such as speech recognition or translation, where latency can be significant on devices with limited processing power. The sequence parallelism inherent in the Transformer architecture aids in highly parallel training. However, efficient deployment of these models is problematic in practice because generative inference progresses one token at a time, and the computation for each token sequentially depends on previously generated tokens [34]. Moreover, while edge deployment frameworks aim to support a broad spectrum of deep learning operations, compatibility issues emerge with specific operations or implementations unique to Transformers. These architectures, which process variable-length sequences, often lead to dynamic computations [5]. Such computations might not align well with frameworks or devices optimized for static operations.

Deploying Transformer models on devices such as mobile phones, IoT devices, robots, and especially the Edge TPU presents additional challenges due to their resource constraints. The Edge TPU, in particular, has a limited amount of on-chip memory and computational resources, making it difficult to deploy large Transformer models [36]. Another obstacle is the mandatory conversion of Transformer models to the TensorFlow Lite (TFLite) format to make them

DisDepth

compatible with the Edge TPU. This conversion, although crucial, can be tedious and may require additional hardware resources. Furthermore, the TFLite format itself, as the only format supported by the Edge TPU, has its own set of operational limitations, adding complexity to the deployment process.

A.2 Advantages of CNNs over Transformers

Convolutional neural networks (CNNs), including architectures such as VGG[38] and ResNet[40], are characterised by their computational simplicity, using operations such as convolutions and pooling. In general, these models are smaller, with inventions such as MobileNet [17] and EfficientNet [39] that have specifically been optimised for edge devices. The long history of CNNs in this areas shows that tools including TensorRT, ONNX, and TensorFlow Lite have been made available with extensive optimisation techniques. Importantly, modern Graphical Processing Units (GPUs) and Application-Specific Integrated Circuits (ASICs) (such as Tensor Processing Units (TPUs) and Neural Processing Units (NPUs)) provide dedicated acceleration for convolution operations, thus making CNNs particularly efficient for deployment in real-world scenarios.