2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

# Is Pre-training Applicable to the Decoder for Dense Prediction?

Chao Ning[1,2], Wanshui Gan[1,2], Weihao Xuan[1,2] and Naoto Yokoya[1,2,†]

***Abstract*— Encoder-decoder networks are commonly used model architectures for dense prediction tasks, where the encoder typically employs a model pre-trained on upstream tasks, while the decoder is often either randomly initialized or pre-trained on other tasks. In this paper, we introduce ×Net, a novel framework that leverages a model pre-trained on upstream tasks as the decoder, fostering a "pre-trained encoder × pre-trained decoder" collaboration within the encoder-decoder network. ×Net effectively addresses the challenges associated with using pre-trained models in the decoding, applying the learned representations to enhance the decoding process. This enables the model to achieve more precise and high-quality dense predictions. By simply coupling the pre-trained encoder and pre-trained decoder, ×Net distinguishes itself as a highly promising approach. Remarkably, it achieves this without relying on decoding-specific structures or task-specific algorithms. Despite its streamlined design, ×Net outperforms advanced methods in tasks such as monocular depth estimation and semantic segmentation. The code is available at https://2j472no.github.io/xNet/.**

## I. Introduction

Since 2015, Jonathan et al. [1] have reinterpreted classification networks as fully convolutional architectures, fine-tuning these models based on their pre-learned representations. Pre-trained models excel at extracting multi-scale features, capturing both local details and global context. Such representations are also central to robotics, where dense perception supports geometry- and semantics-aware decision making for navigation and manipulation.

Following this line, numerous studies have explored improved network structures to decode these visual features, enabling effective solutions for dense prediction tasks such as monocular depth estimation (MDE) [2], [3] and semantic segmentation [4]. In these tasks, pre-trained models typically serve as encoders, while decoding is often performed by randomly initialized modules that progressively refine coarse features into pixel-level predictions.

Recently, scaling up training data has led to substantial gains across self-supervised learning [5], semantic segmentation [6], and depth prediction [7], enabling classical architectures to rival more complex designs. For robotics, this improved generalization in dense predictions is particularly valuable for robust perception under domain shifts.

Despite these advances, in encoder-decoder networks for dense prediction, the encoder is typically pre-trained while the decoder is usually randomly initialized. This raises an important question: if more data improves performance, why is the decoder not pre-trained as well?

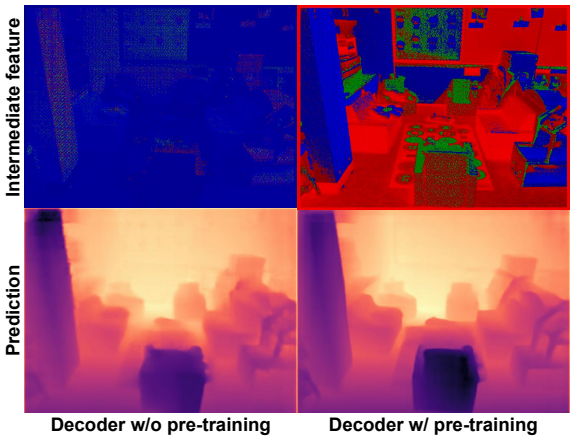


Fig. 1. Comparison of pre-trained and non-pre-trained decoders. When paired with a consistent encoder and architecture, a pre-trained decoder enhances the semantic richness of intermediate feature maps and refines the final predictions.

To address this, we examine why encoders benefit from pre-training, whereas decoders typically do not. Using a pre-trained model as a decoder introduces two key challenges: (1) the decoder's structure differs from that of the pre-trained model, preventing direct loading of pre-trained parameters; and (2) decoding requires producing dense predictions from features, which differs from the encoder's objective of extracting features from images, so upstream pre-trained parameters may not transfer effectively.

In this paper, we introduce ×Net, which enables a "pre-trained encoder × pre-trained decoder" collaboration. It tackles the above challenges and allows pre-learned knowledge to be leveraged within the decoder, yielding substantial improvements. As shown in Figure 1, ×Net enriches intermediate feature maps with semantic information acquired through pre-training, leading to sharper edges and finer details in dense predictions—properties that are particularly valuable for robotics applications requiring precise scene understanding.

In summary, our core contributions are as follows:

- We propose applying a pre-trained model as a decoder in a dense prediction network and address the associated challenges of using pre-trained models in decoding.
- We analyze why a pre-trained decoder matters for dense prediction, highlighting its role in injecting semantic information crucial for high-quality outputs.

[1]Department of Complexity Science and Engineering, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa, Chiba 277-8561, Japan
[2]RIKEN Center for Advanced Intelligence Project, 1-4-1 Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan
[†]Corresponding author

- Extensive experiments show that a pre-trained decoder, without additional task-specific designs, can achieve advanced performance.

## II. Related work

### A. Model pre-training

Model pre-training improves visual representation learning and enhances downstream task performance. Supervised image classification remains a standard pre-training approach, with ImageNet-1k [8] comprising 1.28 million images across 1,000 categories and serving as a common benchmark. Pre-training on larger datasets or with more categories further enriches learned features. Recent advances, such as BEiT [9] (masked image modeling), CLIP [10] (image-language pairing), and DINOv2 [5] (self-supervised learning for vision transformers), leverage extensive data to improve model robustness. Traditionally, these pre-trained networks are used as encoders for feature extraction in dense prediction tasks [11]. In this work, we are the first to employ such networks as decoders for feature decoding.

### B. Encoder-decoder networks for dense prediction

The pioneering work [1] first employed a decode head to fine-tune networks that achieved success in image benchmarks, specifically for pixel-level semantic segmentation. This innovative approach inspired the advancement of numerous dense prediction tasks, which now focus on the efficient decoding of features extracted by pre-trained encoders. Inspired by the FPN, some works enhance network performance by improving network structures. These improvements include a decoding architecture [12] adapted to ViT encoders, and decoders utilizing Transformer architectures [4], [13]. On the other hand, the dense prediction performance is improved through specialized algorithms tailored to specific tasks. For instance, masked attention [4] in image segmentation, normal distance assistance [14], and ground embedding [15]. Our experiments demonstrate that introducing a pre-trained decoder can surpass advanced decoding structures and specialized algorithms.

## III. From pre-trained model to decoder

In this section, we investigate how to effectively integrate pre-trained model into the decoder, addressing two major challenges. First, decoding pixel-level predictions from low-resolution feature maps cannot utilize pre-trained parameters. Common dense prediction network decoders cannot utilize pre-trained parameters. Second, even if pre-trained parameters can be loaded into the decoder, ensuring effective use of the pre-learned representations during the decoding process remains a challenge. We conduct a theoretical analysis of these two challenges, propose solutions, and finally summarize our method, introducing $\times$Net, a dense prediction network that efficiently leverages a pre-trained decoder. We select the widely utilized hierarchical model pre-trained on image classification tasks to represent the pre-trained models. Additionally, we employ an encoder-decoder network with hierarchically connected pyramidal structures to represent common dense prediction networks.

### A. Why pre-training is not available for decoder?

**Reversed structure.** As illustrated in Figure 2, the primary reason pre-training is not applied to decoders is that the structure of decoders is reversed compared to that of pre-trained models. Pre-trained models typically reduce the spatial size of features progressively to extract image categories, whereas dense prediction requires reconstructing high-resolution pixel-level predictions from low-resolution feature maps. The resolutions of feature maps in traditional decoders and pre-trained models can be described as follows:

$$\begin{cases} F^{\frac{1}{32}} \xrightarrow{\mathcal{D}_1} F^{\frac{1}{16}} \xrightarrow{\mathcal{D}_2} F^{\frac{1}{8}} \xrightarrow{\mathcal{D}_3} F^{\frac{1}{4}} \xrightarrow{\mathcal{D}_4,\text{head}} F^{1} \\ F^{1} \xrightarrow{\text{stem},\mathcal{P}_1} F^{\frac{1}{4}} \xrightarrow{\mathcal{P}_2} F^{\frac{1}{8}} \xrightarrow{\mathcal{P}_3} F^{\frac{1}{16}} \xrightarrow{\mathcal{P}_4} F^{\frac{1}{32}} \end{cases}. \quad (1)$$

Here, $\mathcal{D}$ and $\mathcal{P}$ denote the stages of the decoder and the pre-trained model, respectively, with the subscript indicating their sequential order within the model. $F$ represents the feature map, and the superscript denotes the downsampling factor of the width and height. The reversed decoding structure is a prerequisite for utilizing pre-trained parameters. Obviously, employing a pre-trained decoder requires input features at the original resolution and output pixel-level predictions.

**Reversed decoder input.** As described in Equation 1, the input to the decoder is $F^{\frac{1}{32}}$, which must be reshaped to an appropriate resolution, such as $F^1$, to be compatible with the pre-trained decoder. We propose a Re-Shape (RS) module to directly re-shape the encoded feature maps to high resolution for reversed decoder structure. We propose a simple RS module that directly reshapes the encoder output to a size suitable for input into the reversed decoder structure. Specifically, the RS module consists of a normalization layer and a two-layer MLP, with upsampling achieved through a PixelShuffle module positioned of the MLP.

**Reversed decoder prediction.** According to Equation 1, the traditional decoder is required to produce pixel-level predictions ($F^1$). Achieving pixel-level predictions directly from the output of a pre-trained model ($F^{\frac{1}{32}}$) is challenging. Therefore, we propose the Post Feature Pyramid (PFP), which reshapes the feature maps from each of the four stages of the pre-trained decoder to the same resolution using the RS module. These reshaped features are then concatenated along the channel dimension and subsequently fused through another RS module.

With the aforementioned adjustments, the pre-trained model can be used as a decoder, enabling pixel-level predictions in dense prediction tasks.

### B. Motivation for fine-tuning pre-trained decoder

**Optimization challenges.** Inspired by prior work [1], which first fine-tuned image classification baseline networks for semantic segmentation by only modifying the final output head without altering earlier data processing, we modify the decoder's feature map input to facilitate better transfer of pre-learned representations during decoding.

To effectively fine-tune a pre-trained decoder and fully leverage its learned representations during decoding, it is important to mitigate the optimization challenges. As illustrated

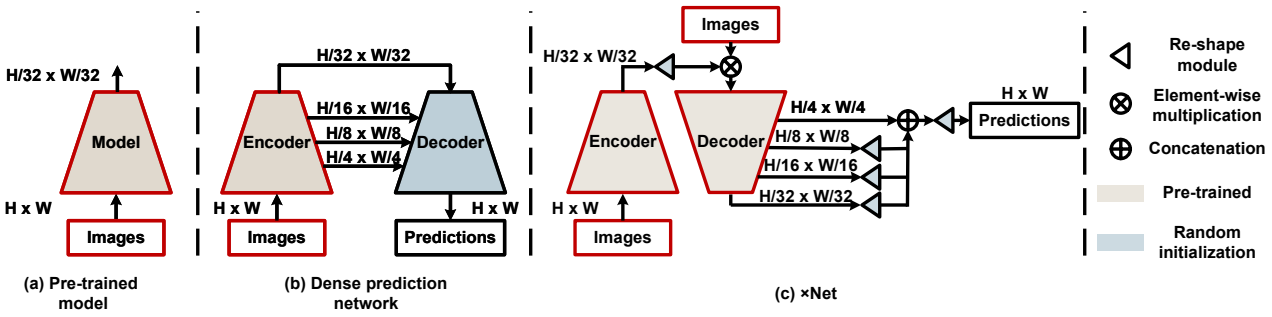


Fig. 2. Overview of the image classification model (left), encoder-decoder network (middle), and our ×Net (right). The pre-trained model represents the commonly used hierarchical structure. The channels are scaled with model size; detailed settings are available in our repository.

in Figure 2, decoders in common dense networks are required to process features from multiple stages of the encoder's output. In contrast, pre-training typically involves extracting features and recognizing objects from image inputs. Thus, to facilitate optimization, decoding should be aligned in a manner that complements the pre-learned representations. Simply, this section aligns the decoder's data processing pipeline with that of the original pre-training.

**Optimization of decoder stages.** In dense prediction networks, the pyramidal connections between the encoder and decoder can introduce optimization difficulties for the pre-trained decoder. The traditional feature pyramid structure can be defined as follows:

$$F^{\frac{1}{k}} = \mathrm{D}_i(E_i^{\frac{1}{k}} + \mathrm{D}_{i-1}(F^{\frac{1}{2k}})), 2 \leq i \leq 4. \tag{2}$$

Here $\mathrm{D}_i$ is the $i$th decoder stage, $F$ denotes the feature map of the decoder, $E_i$ is encoded feature map for $i$th decoder stage and $\frac{1}{k}$ denotes a down-sampling factor of the width and height. Consider $E_i$ input into pre-trained $\mathrm{D}_i$ as follows:

$$E_i = \phi(X_i), \tag{3}$$

where $\phi$ represents the variation from the pre-learned representation $X_i$. For each $\mathrm{D}_i$, there is an variation as follows:

$$\mathcal{L}_i = \|\Delta_i\|_2^2, \quad \Delta_i = \mathrm{D}_i(E_i) - \mathrm{D}_i(X_i). \tag{4}$$

For $\mathrm{D}_2, \mathrm{D}_3$, and $\mathrm{D}_4$, optimization can be approximately defined as:

$$\min_{\mathrm{D}_i} \|\mathrm{D}_i(E_i) - \mathrm{D}_i(X_i)\|^2 + \|\mathrm{D}_i(\Delta_{i-1})\|^2. \tag{5}$$

Here the input $E_i$ of FP introduces an additional term to the optimization objective ($\|\mathrm{D}_i(E_i) - \mathrm{D}_i(X_i)\|^2$). Hence, we remove the FP structure for eliminating the optimization of $\|\mathrm{D}_i(E_i) - \mathrm{D}_i(X_i)\|^2$.

**Optimization of decoder input.** According to Equation 3, the variation of decoder input can be defined as:

$$\phi(I) = \hat{\mathrm{E}}(I), \tag{6}$$

where $I$ denotes the input image, and $\hat{\mathrm{E}}$ is the encoder. Then the optimization target can be approximately described as follows:

$$\begin{cases} \min_{\mathrm{D}_1} \|\mathrm{D}_1(\hat{\mathrm{E}}(I)) - \mathrm{D}_1(I)\|^2 \\ \min_{\mathrm{D}_i} \|\mathrm{D}_i(\Delta_{i-1})\|^2, 2 \leq i \leq 4 \end{cases}. \tag{7}$$

The key to reducing optimization difficulty lies in ensuring that $D_i$ processes encoded features in a manner consistent with image processing. For instance, if at a certain stage the decoding of encoded features is equivalent to processing the image, subsequent stages can align with the pre-trained representations without further optimization. However, retrieving image information from encoded features is challenging, as the encoder's output is abstract and not a simple transformation of the image. Therefore, we mix the input to the pre-trained decoder with the original image, integrating it into the features through element-wise multiplication. This allows $\hat{\mathrm{E}}(I)$ to be viewed as a transformation of $I$, and optimization can proceed along its trajectory.

With these designs, the decoding process more closely mirrors the image processing of pre-training, reducing the difficulty of optimization and facilitating the transfer of pre-learned representations to the decoding.

### C. Dense prediction using pre-trained decoder

In this subsection, we summarize the solutions to the challenges faced by pre-trained decoders in dense prediction and introduce ×Net, a network capable of performing dense prediction with a **"pre-trained Encoder × pre-trained Decoder"** architecture. As illustrated in Figure 2, compared to conventional dense networks, ×Net features a reversed decoder and employs a PFP to achieve pixel-level predictions. By removing the pyramidal connections between the encoder and decoder and integrating image data into the encoded features, the pre-trained representations are more effectively incorporated into the decoding process. The red-highlighted parts in Figure 2 clearly show that ×Net's encoder-decoder structure and workflow closely resemble that of the pre-trained model. For the method of mixing encoded features into image processing, we propose two approaches, as illustrated in Figure 3. The first approach, used in ×Net, mixes the encoded features with the output of the stem. The second approach, applied in ×Net-I, mixes the encoded features directly with the RGB image. The mixing

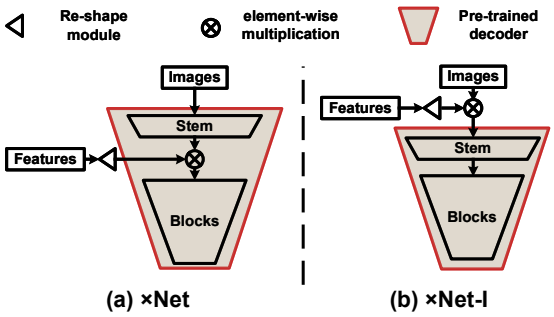


Fig. 3. The difference between ×Net and ×Net-I.

method of ×Net-I provides an intuitive visualization analysis strategy; however, since it compresses the encoded features to a smaller size (approximately $\frac{1}{4}$ to $\frac{1}{2}$) of the pixel count of the stem features, hence its performance is slightly inferior to that of ×Net.

## IV. Experiment Settings

In this section, we present the dense prediction tasks discussed in this paper, the datasets employed, and the experimental setup.

### A. Dense prediction tasks

Similar to previous works [12], [16] on dense prediction, we select two popular dense prediction tasks: MDE and semantic segmentation.

**MDE.** MDE is a pixel-level regression task that involves predicting 3D depth information from a single 2D image. The widely used approach for this task employs a pre-trained encoder to extract features from the image, followed by a decoder module, which is randomly initialized, to decode depth information from the extracted features. Recent methods achieve improved depth prediction by designing advanced model architectures [13], while others use camera parameters to derive geometric information [14], [15] from images to assist in more accurate depth regression.

**Semantic segmentation.** Semantic segmentation is a pixel-level classification task that requires predicting the precise category for each pixel. Due to the potentially large differences in target sizes, typical semantic segmentation methods utilize a multi-branch decoding structure [4] to capture richer contextual information, enabling accurate classification for each pixel.

### B. Datasets and settings

To comprehensively evaluate our approach, we conducted experiments on four MDE datasets and two semantic segmentation datasets. For all experiments, we adopt the commonly used training configurations for each respective task to ensure that training xNet does not introduce any additional computational overhead.

**MDE settings.** We conducted training on three classic MDE datasets: NYU-Depth-V2 [17], KITTI [18], and DDAD [19], using the same training and validation splits as the comparison methods. Additionally, the model trained on the indoor dataset NYU-Depth-V2 is subjected to zero-shot validation on another indoor dataset, SUN RGB-D [20]. We follow the data preprocessing described in [13] for the NYU-Depth-V2 and KITTI datasets, and follow the procedures outlined in [15] for the DDAD dataset. In NYU-Depth-V2, KITTI, and DDAD datasets, the maximum depth is capped at 10m, 80m, and 200m, respectively.

**Semantic segmentation settings.** We performed semantic segmentation experiments on the ADE20K [21] dataset, which contains 150 categories, and the Cityscapes [22] dataset, comprising 19 categories. Following previous work [23], we use the mmsegmentation toolbox [24] for standard training. The crop size for ADE20K is 512×512, and for Cityscapes, it is 512×1024.

## V. Role of Pre-trained Decoders

In this section, we first conduct ablation experiments on ×Net to evaluate the impact of each design component on the model's performance. We then use ×Net-I to analyze how the pre-trained decoder contributes to performance improvements in dense prediction tasks. **All encoders are ImageNet-22k pre-trained ConvNeXt-T** [25], and all decoders are ConvNeXt-T based structures.

TABLE I

Ablation study and pre-training comparisons on NYU-Depth-V2 MDE and ADE20K semantic segmentation. The first group is the ablation study result. The second group involves comparisons among various decoders based on their pre-training volumes. Specifically, "0" indicates no pre-training, while "1K" and "22K" refer to pre-training on ImageNet-1K and ImageNet-22K, respectively. "×Net-22K (add)" indicates that the fusion operation is changed from element-wise multiplication to addition.

| Variant | NYU MDE | | ADE20k Seg. | |
|---|---|---|---|---|
| | $\delta_1$ ↑ | AbsRel ↓ | mAcc↑ | mIoU ↑ |
| FPN | 0.889 | 0.106 | 52.92 | 42.82 |
| +Rev. | 0.896 | 0.104 | 52.42 | 42.31 |
| +22k | 0.909 | 0.099 | 55.32 | 44.71 |
| +PFP | 0.910 | 0.098 | 55.47 | 44.82 |
| -FP | 0.911 | 0.098 | 55.97 | 45.34 |
| ×Net-I-22k | 0.912 | 0.096 | 56.57 | 45.80 |
| ×Net-22k (add) | 0.914 | **0.095** | 56.71 | 45.89 |
| ×Net-22k | **0.916** | **0.095** | **56.82** | **45.91** |
| ×Net-I-0 | 0.899 | 0.104 | 52.72 | 42.67 |
| ×Net-I-1k | 0.908 | 0.100 | 55.50 | 44.83 |
| ×Net-I-22k | **0.912** | **0.096** | **56.57** | **45.80** |

### A. Ablation study

In this subsection, we progressively modify an FPN into ×Net, as shown in the first group of Table I. The ConvNeXt-T structure is suboptimal for dense prediction decoding, and reversing the decoding structure offers marginal MDE improvement but degrades segmentation performance. After structure reversal, pre-trained parameters can be loaded into the decoder, with ImageNet-22k pre-training substantially

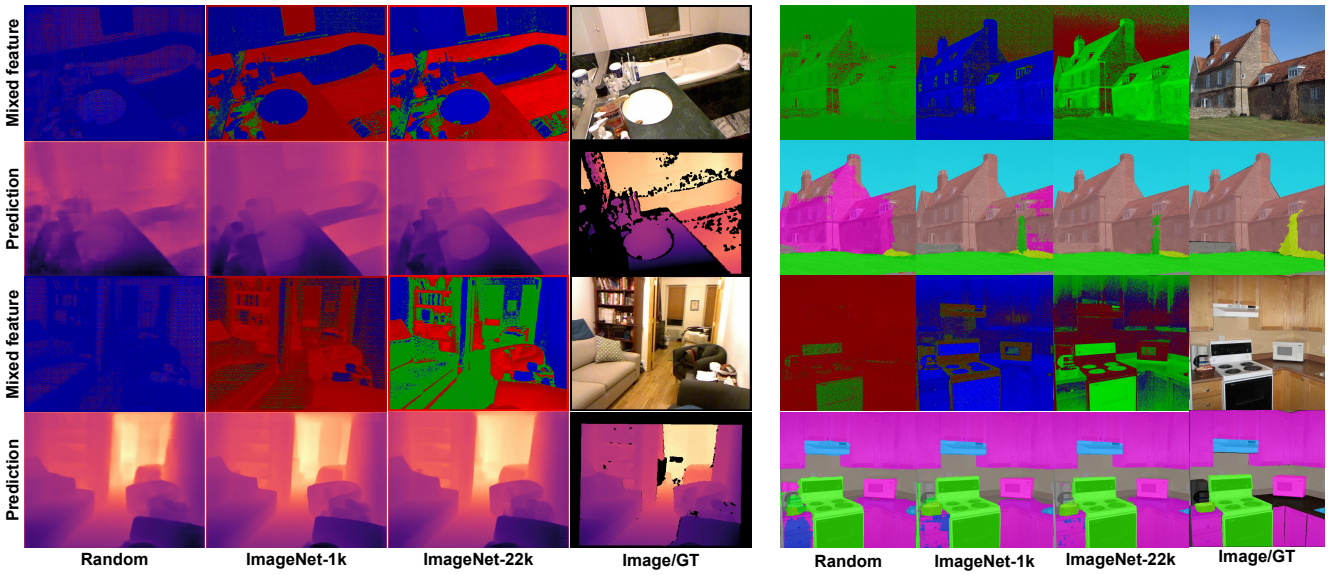


Fig. 4. Qualitative comparisons of decoder pre-training on MDE (left) and semantic segmentation (right). For mixed feature visualization, extract the maximum value from the RGB components, setting the remaining values to zero.

improving results (AbsRel reduced by 4.81%, mIoU increased by 4.78%). Adding PFP and removing FP further improves performance. Although removing the feature pyramid gives only a small improvement, the gain is consistent across both tasks and reduces unnecessary computation. Introducing feature mixing facilitates the transfer of pre-trained representations: the $\times$Net-I mixing reduces AbsRel by 2.04% and increases mIoU by 1.01%, while $\times$Net mixing yields reductions of 3.06% and 1.26%, respectively. Replacing element-wise multiplication with addition in the mixing operation yields similar results, suggesting that the fusion method is not critical, and the primary benefit comes from incorporating pre-trained representations in the decoder.

In summary, the pre-trained decoder is a key component of $\times$Net. $\times$Net is the first architecture to enable loading pre-trained weights into the decoder. The next subsection analyzes how this design improves performance.

### *B. Why does the decoder need pre-training?*

In this subsection, we utilize $\times$Net-I to investigate the role of pre-trained decoders on network performance. We conduct both quantitative and qualitative analyses. We compare same decoder with different initializations: randomly initialized, pre-trained on ImageNet-1k classification, and pre-trained on ImageNet-22k classification.

**Quantitative analyses.** As shown in the second group of Table I, compared to the non-pre-trained decoder, the ImageNet-1k pre-trained decoder achieves a 4.84% reduction in AbsRel, along with improvements of 5.27% in mAcc and 5.06% in mIoU. In contrast, the ImageNet-22k pre-trained decoder demonstrates significant enhancements, with a 7.69% reduction in AbsRel and improvements of 7.30% in mAcc and 7.34% in mIoU.

**Qualitative analyses.** For qualitative analyses, we visualize the mixed features and the final predictions. For $\times$Net-I, due to its mixed feature being a three-channel feature map integrated into the RGB image, it inherently provides a visualization advantage.

The qualitative comparisons are shown in Figure 4. There is an interesting observation: despite utilizing the same encoder, differences in the pre-training of the decoder resulted in significant variations in the mixed features. It is evident that the features of the decoder without pre-training are mostly concentrated in the same channels, whereas the pre-trained decoder allows for the observation of semantic information in the mixed feature. As explained in Subsection III-B, $\times$Net-I aligns the optimization of pre-trained decoder closely with image processing. As a result, the pre-learned representations in the decoder are effectively transferred into the decoding process. For instance, the sofa in the third row of the MDE as shown in Figure 4 is well-separated into channels distinct from surrounding objects. This separation is more pronounced with the ImageNet-22k pre-trained decoder compared to the ImageNet-1k pre-trained decoder, indicating higher confidence with ImageNet-22k pre-training. Given that MDE supervises the depth prediction and the model cannot gain semantic information from MDE training. Such clustering of semantic information can only be derived from the knowledge gained during pre-training.

As illustrated by the prediction comparisons in Figure 4, the inclusion of semantic information in the mixed feature contributes to improvements beyond mere accuracy. In MDE predictions, decoders without pre-training produce coarser outputs, where noticeable jagged edges can be observed upon magnification. In contrast, predictions from pre-trained decoders exhibit sharper details. For semantic segmentation, there is a more coherent output with pre-trained decoders, whereas decoders without pre-training tend to predict differ-

ent categories within the same object. Pre-trained decoders significantly mitigate this issue. The semantic information embedded in the mixed feature enables the decoding process to effectively leverage pre-learned representations for improved prediction.

It can be concluded that pre-trained knowledge is effectively transferable to the decoder, endowing the network's mixed features with semantic information. This semantic enrichment can be leveraged in dense prediction tasks to provide sharper details and more coherent predictions.

## VI. Dense prediction performances

In this section, we compare ×Net with other advanced methods. To thoroughly evaluate the advantages of the pre-trained decoder, **we do not incorporate task-specific designs in ×Net**. Instead, we employed a simple linear probe for prediction. To ensure a fair comparison, we use the same encoder as the main competing methods, including identical model architecture and encoder pre-training. Training was conducted using only standard data augmentation without introducing additional strategies. Moreover, we strictly controlled the computational cost of ×Net using FPS as a metric, ensuring that performance improvements are entirely due to the advantages of the pre-trained decoder.

TABLE II

MDE results on the NYU-Depth-V2 dataset. All methods are trained/fine-tuned on the NYU-Depth-V2 dataset.

| Method | Backbone | $\delta_1 \uparrow$ | AbsRel↓ | log10↓ |
|---|---|---|---|---|
| *GT camera-required methods* | | | | |
| NDDepth [14] | Swin-L [23] | 0.936 | 0.087 | 0.038 |
| Metric3Dv2 [7] | DINOv2-L [5] | 0.989 | 0.047 | 0.020 |
| *GT camera-free methods* | | | | |
| BinsFormer [13] | Swin-L [23] | 0.925 | 0.094 | 0.040 |
| Depth Anything V2 [11] | DINOv2-L [5] | 0.984 | 0.056 | 0.024 |
| ×ConvNeXtV2-T (ours) | Swin-L [23] | 0.936 | 0.086 | 0.037 |
| ×ConvNeXtV2-B (ours) | DINOv2-L [5] | **0.990** | **0.045** | **0.020** |

TABLE III

MDE results on the KITTI dataset. All methods are trained/fine-tuned on the KITTI dataset.

| Method | Backbone | $\delta_1 \uparrow$ | AbsRel↓ | RMS_log↓ |
|---|---|---|---|---|
| *GT camera-required methods* | | | | |
| NDDepth [14] | Swin-L [23] | 0.978 | 0.050 | **0.075** |
| Metric3Dv2 [7] | DINOv2-L [5] | 0.985 | 0.044 | 0.060 |
| *GT camera-free methods* | | | | |
| BinsFormer [13] | Swin-L [23] | 0.974 | 0.052 | 0.079 |
| Depth Anything V2 [11] | DINOv2-L [5] | 0.983 | 0.045 | 0.067 |
| ×ConvNeXt-B (ours) | Swin-L [23] | 0.978 | 0.048 | 0.075 |
| ×ConvNeXtV2-B (ours) | DINOv2-L [5] | **0.988** | **0.037** | **0.056** |

### A. Quantitative comparisons

**NYU-Depth-V2 and KITTI results.** Table II and Table III compare the MDE results of ×Net with methods specifically designed for MDE, and ×Net shows state-of-the-art performance. With the same Swin-L backbone, it surpasses NDDepth on both the NYU-Depth-V2 dataset (0.086 AbsRel vs. 0.087 AbsRel) and the KITTI dataset (0.048 AbsRel vs. 0.050 AbsRel). Unlike NDDepth, which requires using camera intrinsics to compute Normal-Distance for depth prediction, our method simply employs a pre-trained ConvNeXtV2-T [26] or ConvNeXt-B [25] for decoding. For the Metric3Dv2 [7] trained on large-scale datasets, we achieve significant improvements over Metric3Dv2's fine-tuning results on the KITTI dataset (0.037 AbsRel vs. 0.044 AbsRel) using Metric3Dv2 pre-trained DINOv2 × ConvNeXtV2-B [26].

TABLE IV

Zero-shot performances on the SUN RGB-D dataset.

| Method | Backbone | $\delta_1 \uparrow$ | AbsRel↓ | log10↓ |
|---|---|---|---|---|
| *GT camera-required methods* | | | | |
| NDDepth [14] | Swin-L [23] | 0.820 | 0.137 | 0.060 |
| *GT camera-free methods* | | | | |
| BinsFormer [13] | Swin-L [23] | 0.805 | 0.143 | 0.061 |
| ×ConvNeXtV2-T (ours) | Swin-L [23] | **0.824** | **0.133** | **0.059** |

TABLE V

MDE performances on the online KITTI evaluation server. "D2L×C2B" denotes DINOv2-L encoder × ConvNeXtV2-B decoder.

| Method | SILog↓ | sqRel↓ | absRel↓ | iRMSE↓ |
|---|---|---|---|---|
| BinsFormer [13] | 10.14 | 1.69 | 8.23 | 10.90 |
| NDDepth [14] | 9.62 | 1.59 | 7.75 | 10.62 |
| UniDepth [27] | 8.13 | 1.09 | 6.54 | 8.24 |
| D2L×C2B | **7.51** | **0.93** | **6.14** | **7.62** |

**Zero-shot results of SUN RGB-D and online evaluation on KITTI server.** Following previous works [13], [28], [29], [14], we evaluate the NYU-Depth-V2 trained model on the SUN RGB-D dataset. The results in Table IV show that our pre-trained ConvNeXtV2-T decoder achieved the best generalization performance, surpassing previous methods across all three metrics. Additionally, we uploaded the KITTI benchmark training results to the KITTI server for validation. At the time of submission, our model **ranks 1st** among all submissions. As shown in Table V, our method significantly outperforms the previous state-of-the-art method [27] across all four metrics.

TABLE VI

MDE performances on the DDAD dataset.

| Method | Backbone | AbsRel↓ | Sq Rel↓ | RMS_log↓ |
|---|---|---|---|---|
| *GT camera-required methods* | | | | |
| DepthFormer(GEV)† [15] | Swin-L [23] | 0.149 | 2.121 | 0.240 |
| DepthFormer(GEA)† [15] | Swin-L [23] | 0.145 | 2.119 | 0.237 |
| *GT camera-free methods* | | | | |
| DepthFormer [29] | Swin-L [23] | 0.152 | 2.230 | 0.246 |
| BinsFormer [13] | Swin-L [23] | 0.149 | 2.142 | 0.244 |
| ×ConvNeXt-B (ours) | Swin-L [23] | **0.144** | **2.116** | **0.235** |

**DDAD results.** Table VI presents the results on the DDAD dataset. ×Net significantly outperforms many representative

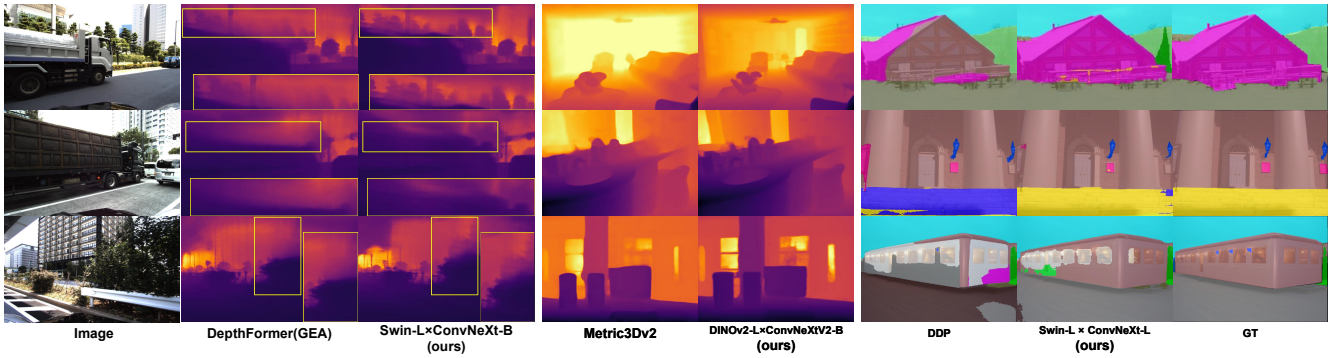


Fig. 5. Qualitative comparisons on DDAD MDE (left) and NYU-Depth-V2 MDE (middle) and ADE20k semantic segmentation (right).

models [29], [28], [13]. Furthermore, even when DepthFormer incorporates ground embedding [15], which requires extra camera parameter inputs, our model achieves better results with a simple combination of Swin-L × ConvNeXt-B.

**ADE20K results.** Table VII presents the comparison results on the ADE20K dataset, where our combination of Swin-L×ConvNeXt-L achieved the best results. Using only a 512×512 crop size, we surpassed UperNet trained with a 640×640 crop size using Swin-L, ConvNeXt-L, and ConvNeXt-XL backbones. Additionally, this simple combination also outperformed the 3-step diffusion model, DDP [16].

TABLE VII

SEMANTIC SEGMENTATION PERFORMANCES ON THE ADE20K DATASET.

| Method | Backbone | crop size | mIoU ↑ |
|---|---|---|---|
| *Image segmentation-specific methods* | | | |
| SegFormer [30] | MiT-B5 [30] | $512^2$ | 51.0 |
| MaskFormer [31] | Swin-L [23] | $640^2$ | 54.1 |
| *General-purpose methods* | | | |
| Swin-UperNet [23], [32] | Swin-L [23] | $640^2$ | 52.1 |
| ConvNeXt-UperNet [25], [32] | ConvNeXt-L [25] | $640^2$ | 52.1 |
| ConvNeXt-UperNet [25], [32] | ConvNeXt-XL [25] | $640^2$ | 53.6 |
| DDP (step 3) [16] | Swin-L [23] | $512^2$ | 53.2 |
| ×ConvNeXt-L (ours) | Swin-L [23] | $512^2$ | **54.3** |

TABLE VIII

SEMANTIC SEGMENTATION PERFORMANCES ON THE CITYSCAPES DATASET. "S.S." AND "M.S." DENOTE SINGLE-SCALE AND MULTI-SCALE INFERENCE RESULTS, RESPECTIVELY.

| Method | Backbone | mIoU (s.s.) ↑ | mIoU (m.s.)↑ |
|---|---|---|---|
| *Image segmentation-specific methods* | | | |
| Segmenter [33] | ViT-L [34] | 79.10 | 81.30 |
| Mask2Former [4] | Swin-L [23] | 83.30 | 84.30 |
| *General-purpose methods* | | | |
| DiversePatch [35] | Swin-L [23] | 82.70 | 83.60 |
| DDP (step 3) [16] | ConvNeXt-L [25] | 83.21 | 83.92 |
| ×ConvNeXt-L (ours) | Swin-L [23] | **83.63** | **84.32** |

**Cityscapes results.** As shown in Table VIII, ×Net surpasses various specialized semantic segmentation methods on the Cityscapes dataset. By using ConvNeXt-L for decoding, it exceeds the performance of Mask2Former [4] (83.63 mIoU vs. 83.30 mIoU), even when Mask2Former employs a dual decoder structure with pixel and Transformer decoders and is trained with additional loss functions, all while using the same backbone.

TABLE IX

EFFICIENCY COMPARISONS. † DENOTES FPS IS EVALUATED ON A RTX 2080 GPU, WHILE ‡ DENOTES FPS IS EVALUATED ON AN A100 GPU.

| Model | Backbone | Dataset | FPS (image/s) |
|---|---|---|---|
| BinsFormer [13] | Swin-L [23] | NYU | 2.20† |
| NDDepth [14] | Swin-L [23] | NYU | 2.23† |
| ×ConvNeXtV2-T | Swin-L [23] | NYU | **5.21**† |
| Depth Anything [11] | DINOv2-L [5] | NYU | 1.85† |
| Metric3Dv2 [7] | DINOv2-L [5] | NYU | 0.89† |
| ×ConvNeXtV2-B | DINOv2-L [5] | NYU | **2.00**† |
| BinsFormer [13] | Swin-L [23] | KITTI | 1.70† |
| NDDepth [14] | Swin-L [23] | KITTI | 1.82† |
| ×ConvNeXt-B | Swin-L [23] | KITTI | **3.32**† |
| Depth Anything [11] | DINOv2-L [5] | KITTI | 0.46† |
| Metric3Dv2 [7] | DINOv2-L [5] | KITTI | 0.63† |
| ×ConvNeXtV2-B | DINOv2-L [5] | KITTI | **1.33**† |
| ConvNeXt-UperNet [25], [32] | ConvNeXt-XL [25] | ADE20K | 13.97‡ |
| DDP [16] | Swin-L [23] | ADE20K | 15.07‡ |
| ×ConvNeXt-L | Swin-L [23] | ADE20K | **15.74**‡ |
| DDP [16] | Swin-L [23] | CityScapes | 4.15‡ |
| ×ConvNeXt-L | Swin-L [23] | CityScapes | **5.95**‡ |

### *B. Qualitative analysis and efficiency comparison*

As illustrated in Figure 5, and as mentioned in Subsection V-B, pre-trained decoder predictions exhibit more robust semantic information compared to other methods, resulting in sharper and more coherent predictions. For instance, in the DDAD MDE predictions, Swin-L × ConvNeXt-B demonstrates sharper detail than DepthFormer (GEA), maintaining precise contours for both large and small objects. In the DDAD MDE predictions, Metric3Dv2 is noted to cause some object deformations, whereas Swin-L × ConvNeXt-L preserves accurate outlines. In the ADE20k semantic segmentation comparisons, DDP tends to show inconsistent

errors in the central areas of large objects, yet Swin-L × ConvNeXt-L maintains consistent predictions.

To ensure a fair comparison, we avoided performance improvements that might arise from using computationally intensive decoders in the decoder section. Table IX compares the computational efficiency of the primary competing methods, demonstrating that ×Net consistently maintains the highest efficiency across all comparisons.

## VII. Conclusion

This paper is the first to introduce the use of a pre-trained decoder in an encoder-decoder structure, termed ×Net. It achieves efficient dense prediction performance through the "pre-trained encoder × pre-trained decoder" collaboration. The pre-trained decoder enriches the network's intermediate features with semantic information, which improves the accuracy and the quality of details of dense predictions.

## VIII. Acknowledgments

This work was supported in part by JST SPRING GX (Grant Number JPMJSP2108) and JST FOREST (Grant Number JPMJFR206S). Weihao Xuan and Wanshui Gan are supported by RIKEN Junior Research Associate (JRA) Program.

## References

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.

[2] C. Ning and H. Gan, "Trap attention: Monocular depth estimation with manual traps," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 5033–5043.

[3] C. Ning, W. Xuan, W. Gan, and N. Yokoya, "Lr$^2$depth: Large-region aggregation at low resolution for efficient monocular depth estimation," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2025, pp. 618–625.

[4] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 1290–1299.

[5] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[6] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," in *Int. Conf. Comput. Vis.*, 2023, pp. 4015–4026.

[7] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen, "Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation," *arXiv preprint arXiv:2404.15506*, 2024.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[9] H. Bao, L. Dong, S. Piao, and F. Wei, "BEiT: BERT pre-training of image transformers," in *Int. Conf. Learn. Represent.*, 2022.

[10] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[11] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao, "Depth anything v2," *Adv. Neural Inform. Process. Syst.*, vol. 37, pp. 21 875–21 911, 2024.

[12] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 12 179–12 188.

[13] Z. Li, X. Wang, X. Liu, and J. Jiang, "Binsformer: Revisiting adaptive bins for monocular depth estimation," *IEEE Trans. Image Process.*, 2024.

[14] S. Shao, Z. Pei, W. Chen, X. Wu, and Z. Li, "Nddepth: Normal-distance assisted monocular depth estimation," in *Int. Conf. Comput. Vis.*, 2023, pp. 7931–7940.

[15] X. Yang, Z. Ma, Z. Ji, and Z. Ren, "Gedepth: Ground embedding for monocular depth estimation," in *Int. Conf. Comput. Vis.*, 2023, pp. 12 719–12 727.

[16] Y. Ji, Z. Chen, E. Xie, L. Hong, X. Liu, Z. Liu, T. Lu, Z. Li, and P. Luo, "Ddp: Diffusion model for dense visual prediction," in *Int. Conf. Comput. Vis.*, 2023, pp. 21 741–21 752.

[17] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Eur. Conf. Comput. Vis.* Springer, 2012, pp. 746–760.

[18] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[19] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon, "3d packing for self-supervised monocular depth estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2485–2494.

[20] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 567–576.

[21] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 633–641.

[22] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3213–3223.

[23] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Int. Conf. Comput. Vis.*, 2021, pp. 10 012–10 022.

[24] M. Contributors, "MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark," https://github.com/open-mmlab/mmsegmentation, 2020.

[25] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 976–11 986.

[26] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie, "Convnext v2: Co-designing and scaling convnets with masked autoencoders," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 16 133–16 142.

[27] L. Piccinelli, Y.-H. Yang, C. Sakaridis, M. Segu, S. Li, L. Van Gool, and F. Yu, "Unidepth: Universal monocular metric depth estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024, pp. 10 106–10 116.

[28] A. Agarwal and C. Arora, "Attention attention everywhere: Monocular depth prediction with skip attention," in *WACV*, 2023, pp. 5861–5870.

[29] Z. Li, Z. Chen, X. Liu, and J. Jiang, "Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation," *Machine Intelligence Research*, vol. 20, no. 6, pp. 837–854, 2023.

[30] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Adv. Neural Inform. Process. Syst.*, vol. 34, pp. 12 077–12 090, 2021.

[31] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Adv. Neural Inform. Process. Syst.*, vol. 34, pp. 17 864–17 875, 2021.

[32] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, and J. Sun, "Unified perceptual parsing for scene understanding," in *Eur. Conf. Comput. Vis.*, 2018, pp. 418–434.

[33] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Int. Conf. Comput. Vis.*, 2021, pp. 7262–7272.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *Int. Conf. Learn. Represent.*, 2021.

[35] C. Gong, D. Wang, M. Li, V. Chandra, and Q. Liu, "Vision transformers with patch diversification," *arXiv preprint arXiv:2104.12753*, 2021.