arXiv:2301.02229v2 [cs.CV] 14 Feb 2023

All in Tokens: Unifying Output Space of Visual Tasks via Soft Token

Jia Ning[1,4*], Chen Li[2,4*], Zheng Zhang[4*], Zigang Geng[3,4], Qi Dai[4], Kun He[1], Han Hu[4]

[1]Huazhong University of Science and Technology [2]Xi'an Jiaotong University
[3]University of Science and Technology of China [4]Microsoft Research Asia

{t-jianing,t-chenli1,zhez,t-ziganggeng,qid,hanhu}@microsoft.com
brooklet60@hust.edu.cn

Abstract

Unlike language tasks, where the output space is usually limited to a set of tokens, the output space of visual tasks is more complicated, making it difficult to build a unified visual model for various visual tasks. In this paper, we seek to unify the output space of visual tasks, so that we can also build a unified model for visual tasks. To this end, we demonstrate a single unified model that simultaneously handles two typical visual tasks of instance segmentation and depth estimation, which have discrete/fixed-length and continuous/varied-length outputs, respectively. We propose several new techniques that take into account the particularity of visual tasks: 1) Soft token. We employ soft token to represent the task output. Unlike hard tokens in the common VQ-VAE which are assigned one-hot to discrete codebooks/vocabularies, the soft token is assigned softly to the codebook embeddings. Soft token can improve the accuracy of both the next token inference and decoding of the task output; 2) Mask augmentation. Many visual tasks have corruption, undefined or invalid values in label annotations, i.e., occluded area of depth maps. We show that a mask augmentation technique can greatly benefit these tasks. With these new techniques and other designs, we show that the proposed general-purpose task-solver can perform both instance segmentation and depth estimation well. Particularly, we achieve 0.275 RMSE on the specific task of NYUv2 depth estimation, setting a new record on this benchmark. The general-purpose task-solver, dubbed AiT, is available at https://github.com/SwinTransformer/AiT.

1. Introduction

A unified model for various tasks across modalities is one of the most ambitious goals of artificial intelligence (AI). However, this is challenging due to the diversity and complexity of real-world tasks. Despite the difficulties, large-scale language models in the field of natural language processing (NLP), *e.g.*, GPT-3 [6], have demonstrated impressive capabilities as a general-purpose solver for language tasks. Encouraged by the success of NLP, this paper attempts to investigate the possibility of universal models for various computer vision tasks.

Most existing related research for universal computer vision models focuses on the unification of architectures [19, 40] and pre-training [2]. A few works aim for developing one model for multiple visual tasks, yet they are usually applicable to limited tasks: [1] handles only tasks with language as output; CLIP models [33] and the follow-ups [48, 49, 52] tackle only retrieval and image classification tasks; [9] deals with tasks that have describable and sequential outputs. In this paper, we aim for a truly general-purpose solver for various vision tasks. To this end, we first notice that a key obstacle was overlooked and under-explored: unlike language tasks whose inputs and outputs are represented by language tokens of the same form, the output forms for different computer vision tasks are more diverse. For example, the output of object detection is a set of coordinates and labels; the output of semantic segmentation is a discrete label map; the output of depth estimation is an image with floating-point values; and the output of flow estimation is a vector field.

In this paper, we address this obstacle by unifying the output space of various visual tasks through a general tokenizer to encode the output space into a set of tokens. Specifically, the proposed framework consists of three components: *tokenizer*, *detokenizer*, and *task solver*, as illustrated in Figure 1. The *tokenizer* and *detokenizer* are instantiated by VQ-VAE [38], which encodes the task output into a set of tokens, which are then reconstructed into the original output by the decoder. The *task solver* for different vision tasks is instantiated using an auto-regressive encoder-decoder model. The encoder-decoder model takes images as inputs and predicts a token sequence in a causal mode,

*Equal Contribution. Jia, Chen, and Zigang are interns at MSRA.

which is decoded into the original task-specific output form by the VQ-VAE decoder.

To improve the effectiveness, we propose several new techniques that take into account the particularity of visual tasks. First, we propose soft token instead of hard ones as used in language tasks or by the original visual VQ-VAE framework. The soft token is represented by probability vectors, with each value in a vector denoting the probability belonging to a codebook. When a soft token is fed to the input of the detokenizer or to the input of the next token prediction network, its input embedding is computed by the weighted average of the codebook embeddings based on the probability values. This indicates that the soft token embedding has spanned an interpolable continuous space, which may better represent the visual output, especially when it's continuous. The continuous nature of the soft token also allows the introduction of an auxiliary loss which learns the task output end-to-end, back from the detokenizer output to the task-solver input. In experiments, we show that the three usages of the soft token can all lead to better results.

Second, we propose a mask augmentation technique to deal with visual tasks which have corrupted, undefined, or invalid values in annotations. The depth estimation is a typical example of such a task, where the occluded area is not defined [37], as shown in Figure 3. The undefined regions make it difficult to the training of VQ-VAE tokenizers and detokenizers because it does not know what to reconstruct in the undefined area. To solve this problem, we randomly mask several patches of the input depth map when training VQ-VAE. Unlike undefined areas, manually masked patches have ground truth annotations, which help train the VQ-VAE network to be able to recover the ground truth for undefined areas. In experiments, we show this technique to notably improve the accuracy of depth estimation. The mask augmentation technique is also expected to be useful for other visual problems with similar undefined or corrupted annotations, and therefore strengthens the generality of our unified task solver.

Third, we systematically study the impact of architectures in the VQ-VAE model. We find that a very lightweight VQ-VAE with up to 5 convolution layers, 2 residual blocks, and 128 codebooks can serve as a very powerful tokenizer. The number of parameters and computations can be as small as 2M and 0.06G FLOPs. This suggests that the overhead of the VQ-VAE model is small, which improves the practicality of our framework.

We choose two classical visual tasks with varying output spaces to validate our approach: depth estimation, and instance segmentation, whose output spaces are in formats of the floating-point maps and binary masks, and have fixed size and variable size, respectively. We achieve competitive accuracy on COCO instance segmentation [26], and state-of-the-art accuracy for NYUv2 depth estimation [37]. The proposed framework and techniques are generic and can be applied to other visual tasks, which will be our future work.

2. Related Works

Unified Frameworks in Computer Vision Encouraged by the success of T5 [34]/GPT [6] in NLP, the exploration of a single unified model for various tasks in computer vision has emerged. However, most existing works [1, 40, 48, 49, 52] are focused on the training algorithm or model architectures. This makes these models either only available as pre-trained models [40, 48, 49] or only for VL-related tasks [1, 52]. Perceiver-IO [19] presents a framework that can process different vision tasks, and it adopts the learned positional encoding or Fourier feature to unify the output of different tasks.

Very recently, Pix2SeqV2 [9] proposed to unify different vision tasks into tokens and the tokens in Pix2SeqV2 need to be designed manually for different tasks. For example, they use the polygon to represent the instance segmentation.

The most related works with ours are UViM [21] and Unified-IO [29]. They also adopt VQ-GAN/VQ-VAE as a general tokenizer/detokenizer and an auto-regressive Transformer encoder/decoder to solve different tasks. However, they are direct applications of these techniques without an in-depth consideration of the particularity of visual problems.

Our study, while concurrently started, takes more in-depth consideration of the particularity of visual problems. We propose techniques of soft token and mask augmentation, which prove beneficial generally for visual tasks or a part of them. We also extensively investigate the architecture of the VQ-VAE, which shows that this part can be made very light-weight, and thus make the framework more practical.

Vector-Quantization Discretized token output space is widely used in generative models, such as DALL-E [35], VQGAN [14], and VQ-Diffusion [16], to represent high-dimensional complex data. Models like VQ-VAE [38], dVAE [3] define a discrete latent space with the encoder-decoder architecture and a fixed size of codebook, The input is mapped to the discrete tokens of the codebook. We adopt the VQ-VAE framework to build our token space, and our soft token approach that treats the token space as a continuous one instead of the original discrete one expands the usage of VQ-VAE.

Monocular Depth Estimation Monocular depth estimation is a fundamental task for 3D perception. Deep learning dominates the depth estimation since Eigen *et al*. [13] introduces it into the depth task. The follow-up works include proposing powerful network [23, 24, 36], designing novel

augmentation [18, 20], making use of the geometric constraints [32,47], exploring pairwise relationship [10,22,53], combing with conditional random field [27, 44, 50].

Some works [30, 36, 39, 46, 51] combine depth estimation with other tasks, such as semantic segmentation, and edge estimation. However, they design different heads and loss functions for different tasks respectively. There are also some methods [4, 5, 12, 15, 25] discretizing the continuous depth and cast the depth estimation as a per-pixel classification task. Our approach represents the depth maps as a set of tokens, and unifies it with other visual tasks, *i.e*. instance segmentation, in a unified network structure and output space. More importantly, we show that a general framework for various visual tasks can achieve state-of-the-art accuracy on the NYUv2 depth estimation task.

Instance Segmentation Instance segmentation aims to predict the segments of each instance. There are many works [17,41,45] studying how to represent the masks. For example, MaskRCNN [17] used a binary mask, Dense RepPoints [45] adopts a set of deformable points to represent the segments, and PolarMask [41] models the segments by polygons. While their representation is specific to instance segmentation, we model the instance segmentation by a set of discrete (soft) tokens, which is more general for visual representations.

3. Framework

The goal of this work is to unify the output space of visual tasks into discrete tokens and to build a single model that can handle different tasks simultaneously. In this section, we present the framework to achieve this goal, which is shown in Figure. 1. The framework consists of three modules, a *tokenizer* that encodes the task output to the discrete tokens, a *detokenizer* that decodes tokens to the task output, and a *task-solver* that predicts tokens from images. In our approach, the encoder and decoder of VQ-VAE are used as the tokenizer and detokenizer, and the task-solver is implemented by an auto-regressive encoder-decoder model. During training, task annotations are first mapped by the tokenizer as discrete tokens and used as supervision to train the task-solver. In inference, the tokens predicted by the task-solver are decoded by the detokenizer into task outputs.

3.1. Tokenizer and Detokenizer

VQ-VAE is an encoder-decoder model with a set of latent codes. It was originally proposed to learn discrete representation for natural images. In this work, we use its encoder and decoder as the tokenizer and detokenizer. In training, the input image is encoded as a set of contiguous embeddings, and these embeddings are assigned to their nearest latent codes. In the decoder, the corresponding codes are used as inputs instead of contiguous embeddings and then decoded into the image. By minimizing the reconstruction loss between the input image and decoded images, the encoder, decoder and latent codes can be trained.

Since we adopt discrete tokens as targets in the task-solver, the accuracy of reconstruction has an upper-bound on the performance of the whole framework. In addition, both training and inference of task-solver require the tokenizer and detokenizer, the fast inference speed is also desired. The original network architecture of VQ-VAE is designed for natural images, which have more complex textures and colors than the task output, making it not the optimal design for us. Therefore, we have exhaustively studied the effects of VQ-VAE with different design choices in our framework.

Typical reconstruction losses (*e.g*. l-1 loss, MSE loss, *etc*.) cannot directly reflect the realistic performance of the task, we use standard evaluation metrics for different tasks to measure VQ-VAE. As shown in Table 2 and Table 4. We found that a very lightweight VQ-VAE can achieve promising results in depth estimation and instance segmentation.

Since the input value ranges for depth estimation and instance segmentation are different. We train two VQ-VAE models for two tasks separately, and they have similar architecture. For depth estimation, the encoder consists of 5 convolution layers (kernel size is 3 and stride is 2) and follows 2 residual blocks. The output feature map has a downsample ratio of 32, and the channel dimension is progressively increased from 16 to 256. The architecture of the decoder is symmetrical to the encoder, only replacing the convolution layers with the deconvolution layer. For instance segmentation, we reduce the convolution and deconvolution of the encoder and decoder from 5 layers to 4 layers and keep all others the same. Subsequently, the downsample ratio is changed to 16.

In addition, compared to the standard VQ-VAE usually adopts a large codebook size (*e.g*. 8192), our codebook size can be reduced to 128. We note that though larger codebook size consistently improves VQ-VAE reconstruction ability, they show no difference when applied to task-solver (see Table 1 and Table 2). There are two speculations on the effectiveness of a small codebook: 1) the output space of depth and segmentation is simple, without the need for a large codebook; 2) the large codebook may increase the learning difficulty for task-solver.

3.2. Task-solver

The task-solver is an auto-regressive encoder-decoder network. The encoder is a Swin Transformer with 6 additional standard transformer blocks, each block consists of a self-attention and an FFN. The decoder has 6 blocks, each block consists of a self-attention, a cross-attention, and an FFN. The architecture we used is similar to [8].

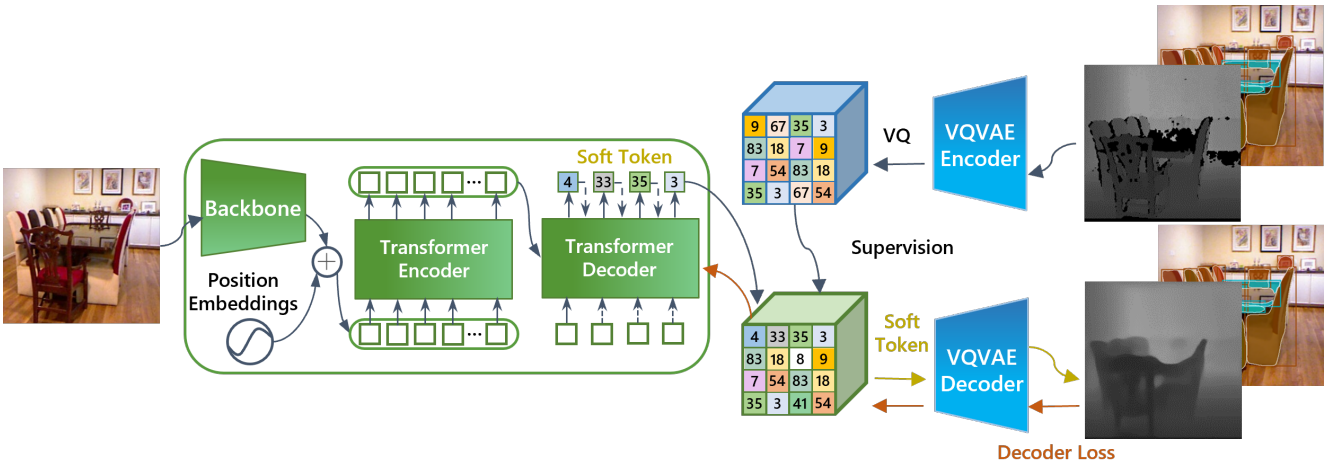

Figure 1. Illustration of our unified framework with two stages. In this framework, various vision task outputs are transferred to discrete token space by a VQ-VAE tokenizer. In this way, discrete or continuous visual tasks can be converted into one discrete classified task,

Given an input image, the encoder is first applied to learn a generic representation of all tasks. Then, based on the given *task token*, the decoder is used to predict a token sequence in an auto-regressive manner. For different tasks, we customize their sequence formats, as described in the following:

Depth Estimation The task token of depth estimation is denoted as `[DEP]`. It has a straightforward format, which is a token sequence of length $\frac{HW}{32^2}$, where H and W are the height and width of input images, respectively.

Instance Segmentation The sequence format of instance segmentation is more complicated than depth estimation, which consists of three parts: bounding box coordinates, class of bounding box, and a binary mask. We follow the practice of Pix2Seq [8] for representing coordinates and the class of a bounding box. The coordinates are manually quantized into 2000 bins, and different classes are represented by different tokens (including a background class, *e.g.* COCO dataset has 81 class tokens in total). For the binary mask, we use 4×4 tokens to represent one mask. It is worth noting that since the computational complexity of auto-regressive is proportional to the square of the output sequence length, we have to use very few tokens to represent the mask. Nevertheless, benefitting from our powerful detokenizer, these tokens can be decoded to a 64×64 mask. Based on these designs, each instance is represented by a total of 21 tokens (*i.e.* 4 tokens for coordinates, 1 token for class, 16 tokens for mask), and we use `[INS]` as the task token of instance segmentation, as shown in Figure 2. We append the meaningless zero mask for the noise box and do not add loss on those mask tokens during training.

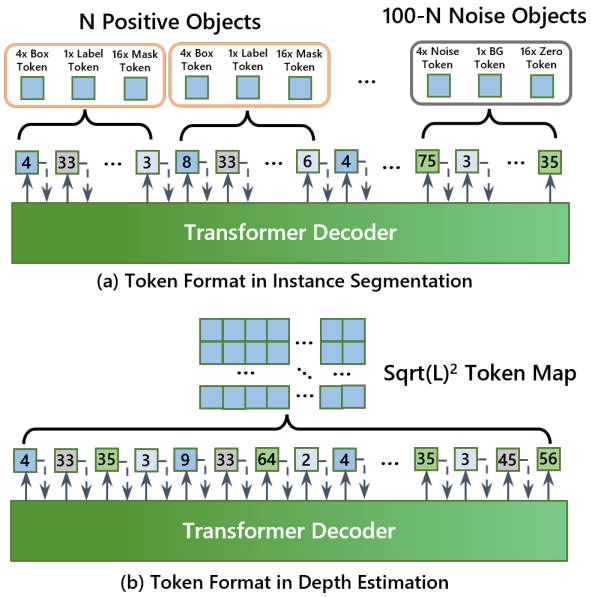

Figure 2. Illustration of instance segmentation and depth estimation token format. (a) We organize every object by 21 tokens. For positive objects, this format includes 4 box, 1 label and 16 mask tokens. While for noise tokens, 4 noise, 1 background and 16 zero tokens are employed; (b) For dense tasks like depth estimation, we treat every patch a token to form a token map.

3.3. Soft Token

In a typical auto-regressive prediction procedure, the token with the maximal predicted probability is selected as the output, and used its embedding as the input to the decoder for the next prediction step. This approach is called *hard-inference*. However, since the tokens learned by VQ-VAE

Figure 3. There are some corrupted regions (black regions/pixels) in the GT depth map. While we have ignored these regions in training VQ-VAE as well, the reconstructed regions are still abnormal, which is reflected in the shadows in reconstruction results. This phenomenon can be alleviated by adding masked augmentation.

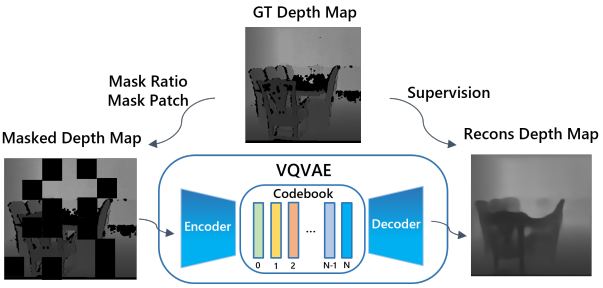

Figure 4. Illustration of our masked augmentation. In order to make the tokenizer have a stronger complement capability, we add random mask noise under a certain mask ratio and patch size to the original GT depth maps as inputs. But we still use the GT depth maps to apply reconstruction loss.

are not completely independent of each other, the correlation between the tokens may affect the token prediction accuracy, making the hard inference probably not optimal. To leverage the correlation, a *soft token* technique is presented in the inference: instead of directly using the embedding of a single token, the soft token is the weighted averaged embedding of different tokens by their prediction probability. In addition to being applied in task-solver to predict the next token more accurately, the same idea can also be used in detokenizor to get better reconstruction results.

Furthermore, the soft token results in the embedding space being spanned to an interpolable continuous space. Therefore, we can introduce an auxiliary loss that learns the task-specific output targets in an end-to-end manner by backing from the detokenzior output to the task-solver input.

We examined the soft token in both instance segmentation and depth estimation. It can consistently improve the performance without any additional computational cost, which is a *free-lunch* technique in inference.

3.4. Mask Augmentation in Depth Estimation

The ground-truth depth maps in the depth estimation dataset often have some corrupted regions that are not annotated with depth information. In the conventional depth estimation frameworks, these regions are ignored during training. However, the same solution cannot be applied to our framework. There are two challenges: First, while we have ignored these regions in training VQ-VAE as well, the reconstructed regions are still abnormal (see Figure 3) and further affect the training of the task-solver and make the final result also have many artifacts; Second, a token predicted by VQ-VAE corresponds to a 32^2 patch, which may contain both normal and corrupted pixels. Therefore, it is hard to deal with this issue by ignoring the tokens.

As shown in Figure 4, we present to introduce mask augmentation in the training of VQ-VAE to alleviate this challenge. Specifically, we randomly mask some regions in the input depth images and then use their original depth information as supervision. In this way, the VQ-VAE can complete/recover some corrupted regions with reasonable results. Figure 3 shows the visualization. In Table 8, we notice that applying mask augmentation can improve the performance of depth estimation.

4. Experiments

4.1. Tasks and Datasets

To examine the generalizability of our framework, we choose depth estimation and instance segmentation, which are two tasks with very different output spaces.

Instance Segmentation The instance segmentation requires predicting the location, the class, and the mask of each instance. The COCO2017 benchmark is one of the most challenging datasets for this task. It consists of 117K training images, 5K validation images, and 41K test images. A total of 80 classes annotation are provided. In our experiments, we follow the common setting of previous works [17, 41, 45] that report the performance on the validation set for comparison.

Depth Estimation Depth estimation is a fundamental problem in computer vision, which requires estimating the depth for each pixel. Unlike segmentation whose output is a binary mask, the depth map is a floating point image. In this work, we use the NYUv2 Depth dataset, which consists of 24K training images and 654 validation images, and the RMSE is used as the major metric.

4.2. Implementation Details

We train two separate VQ-VAE for depth estimation and instance segmentation. For depth estimation, the input image size used in depth is 480^2, with a batch size of 8. The Adam optimizer is used with the base learning rate of 3e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$. An exponential learning rate schedule is applied with the learning rate decay of 0.98 and a total of 20 training epochs. For instance segmentation. The input image size is 64^2 with a batch size of 512. The Adam optimizer is used with the base learning rate of 3e-4, $\beta_1 = 0.9$, $\beta_2 = 0.999$. A cosine learning rate schedule is applied with a total of 20 training epochs. By default, the EMA model update technique is used for all VQ-VAE models.

For the task-solver, we adopt the auto-regressive encoder-decoder architecture, which is similar to Pix2Seq [8]. It consists of a backbone, 6 encoder layers, and 6 decoder layers. We use the SwinV2 [28] as the backbone, which is pre-trained with SimMIM [43]. Most experiments in the ablation study are separately trained on depth estimation and instance segmentation.

In depth estimation, we use the AdamW optimizer with a base learning rate of 2e-4 and 1e-4, the weight decay of 0.05 and 0.075 for SwinV2-B and SwinV2-L, respectively. The β_1 and β_2 are set to 0.9 and 0.999, and drop path rate is set to 0.1. The total training length is 25 epochs with the batch size of 24. The step learning rate schedule is used and the learning rate dropped to 2e-5 at the 18th epoch. For data augmentation, the random cropping of 480^2 and horizontal flip with probability 0.5 are employed. We also append random brightness contrast, random gamma, and hue saturation value.

Training of instance segmentation from scratch is expensive because of the long sequence length. To reduce the cost, we first train an object detection model and then fine-tuning on instance segmentation. For object detection pre-training, the AdamW optimizer with a base learning rate of 1e-3, a weight decay of 0.05, a drop path rate of 0.3, and a layer decay of 0.85, linear decay learning rate scheduler are applied, and a total of 100 training epochs with a batch size of 128 are performed. For instance segmentation fine-tuning, we only initialized the backbone and encoder with detection pre-trained model, and randomly initialized the decoder. In addition, AdamW optimizer with a base learning rate of 1e-4, a weight decay of 0.05, and a layer decay of 0.85, linear decay learning rate scheduler are applied, and the total training length is 50 epochs with a batch size of 16. Large-scale jittering with the range of 0.1 to 3.0 and crop size of 640^2 are used for both object detection and instance segmentation. $\beta_1 = 0.9$, $\beta_2 = 0.999$ are used for AdamW in all experiments.

4.3. Ablation Study

We ablate the key design choices and techniques in this section. By default, we train the model separately for each task in the ablation study for better illustration, and SwinV2-B is used as the default backbone. For depth estimation experiments, a VQ-VAE with codebook size of 128, downsample rate of 32 and mask ratio of 0.5 is used by default. For instance segmentation, the codebook size is 128 and the downsample rate is 16. If not specified, we use the soft token but do not apply auxiliary loss for all ablation experiments.

Table 1. Ablation study on codebook size of VQ-VAE on depth and instance segmentation in reconstruction.

Width	#tokens	Depth	Instance Seg.
		RMSE	Mask mAP
1.0×	64	0.1025	88.94
1.0×	128	0.0966	89.34
1.0×	256	**0.0902**	**90.22**

Table 2. Ablation study on codebook size of VQ-VAE on depth and instance segmentation in task-solver.

Width	#tokens	Depth	Instance Seg.	
		RMSE	Box mAP	Mask mAP
1.0×	64	0.3090	43.5	33.0
1.0×	128	**0.3080**	**43.6**	33.2
1.0×	256	0.3119	43.4	**33.4**

Table 3. Ablation study on the width of VQ-VAE on depth and instance segmentation in reconstruction.

Width	#tokens	Depth	Instance Seg.
		RMSE	Mask mAP
0.5×	128	0.1196	88.97
1.0×	128	**0.0966**	89.34
2.0×	128	0.1025	**90.92**

Table 4. Ablation study on the width of VQ-VAE on depth and instance segmentation in task-solver.

Width	#tokens	Depth	Instance Seg.	
		RMSE	Box mAP	Mask mAP
0.5×	128	0.3127	43.4	33.1
1.0×	128	**0.3080**	**43.6**	**33.2**
2.0×	128	0.3124	43.2	33.1

Architecture of VQ-VAE We study how different designs of VQ-VAE affect performance. We first evaluate the reconstruction performance of different codebook sizes. To more accurately and intuitively observe the reconstruction performance, our evaluation is performed on the validation

Table 5. Ablation study on the downsample ratio of VQ-VAE on depth and instance segmentation in reconstruction.

Downsample Ratio	Depth	Instance Seg.
	RMSE	Mask mAP
32	0.0966	70.91
16	0.0696	**89.34**
8	**0.0515**	-

Table 6. Ablation study on the downsample ratio of VQ-VAE on depth and instance segmentation in task-solver.

Downsample Ratio	Depth	Instance Seg.	
	RMSE	Box mAP	Mask mAP
32	**0.3080**	42.7	30.3
16	0.3304	**43.6**	**33.2**
8	0.3514	-	-

Table 7. Ablation on the effectiveness of soft token.

Description	Depth	Instance Seg.	
	RMSE	box mAP	mask mAP
Baseline (hard-inf)	0.3174	43.6	31.1
On. task-solver	0.3127	43.5	32.1
On. detokenizor	0.3120	43.6	32.3
On. both	0.3080	43.6	33.2
On. both + aux. loss	**0.3052**	**43.3**	**34.2**

set of different tasks and adopts mask mAP and RMSE as metrics. The results are shown in Table 1. We find that although the large codebook size (*e.g.* 256) benefits the reconstruction performance, the small codebook size (*e.g.* 64) can also yield sufficiently good reconstruction performance. Further applying to the task-solver, we found different codebook size has little effect on final performance (see Table 2).

Using the same evaluation method, we study the effect of the width of VQ-VAE. Table 3 shows the reconstruction performance and Table 4 shows the performance of applying to task-solver. Similar to the observation on codebook size, we find that the network width has little effect on the final performance.

The downsample ratio of VQ-VAE is another key that may affect network performance. We vary the downsample ratio in [8, 16, 32]. We note that the instance segmentation cannot support a downsample ratio of 8 because it results in too long sequences. Table 5 shows the reconstruction performance, as the downsample ratio increases, the reconstruction performance gets worse, satisfying the intuition. However, we find that better reconstruction performance does not always lead to better performance when applying the VQ-VAE in task-solver. In Table 6, the best performance is achieved at downsample ratio of 32. We explain this phenomenon is that a smaller downsample ratio facilitates reconstruction, but it also increases the length of the token sequence, which is detrimental to the task-solver.

Soft Token To leverage the correlation between tokens, we introduce the soft token techniques, which can be used to improve the performance in the inference stage for free. We examined this technique in Table 7. Compared with the hard inference baseline, applying the soft token in task-solver and detokenzior alone can bring performance gains, and further performance improvement is achieved when used in two stages at the same time: the depth performance is improved by +0.009 RMSE and the instance segmentation is improved by +2.1 mAP. On top of it, adding the auxiliary loss on the output of detokenizor enlarges the gain to +0.012 RMSE and +3.1 mAP.

Table 8. Affects of mask augmentation in training VQ-VAE on depth. The patch size of all models is set to 16.

Mask Ratio	VQ-VAE	task-solver
0.0	**0.0831**	0.3105
0.3	0.0893	0.3093
0.5	0.0966	**0.3080**
0.7	0.1196	0.3225

Mask Augmentation We evaluate the effectiveness of mask augmentation in depth estimation. The different mask ratios vary from 0.3 to 0.7 are used, and Table 8 shows the results. The best performance is achieved by using the mask ratio of 0.5, which is +0.003 better than the baseline.

4.4. Preliminary Study on Parallel Prediction

In previous sections, we mainly demonstrate the use of auto-regressive models to unify various visual tasks. Note the bi-directional or even qua-directional natural may encourage parallel decoding approaches. Therefore, we make some preliminary studies on the parallel decoder instead of the auto-regressive decoder.

As parallel decoder has been prevalent in object detection and instance segmentation, *i.e.* DETR [7] and Mask2Former [11], we mainly studied the parallel decoder for the depth estimation problem. As shown in Table 9, the parallel decoder performs better than the auto-regressive counterpart by around 0.01 RMSE (0.275 *vs.* 0.284). This indicates a strong potential than the auto-regressive models.

Our techniques such as soft token also benefit the parallel decoder, as shown in Table 9. This implies the generality of the proposed techniques.

4.5. Comparison with Other Unified Frameworks and State-of-the-arts in Depth Estimation

UViM and Unified-IO are the most relevant works to ours. We compare the performance with these methods on the overlap task, *i.e.* NYUv2 depth estimation. The results are shown in Table. 9. Our auto-regressive approach achieves 0.284 RMSE, which is 0.183 and 0.101 better than

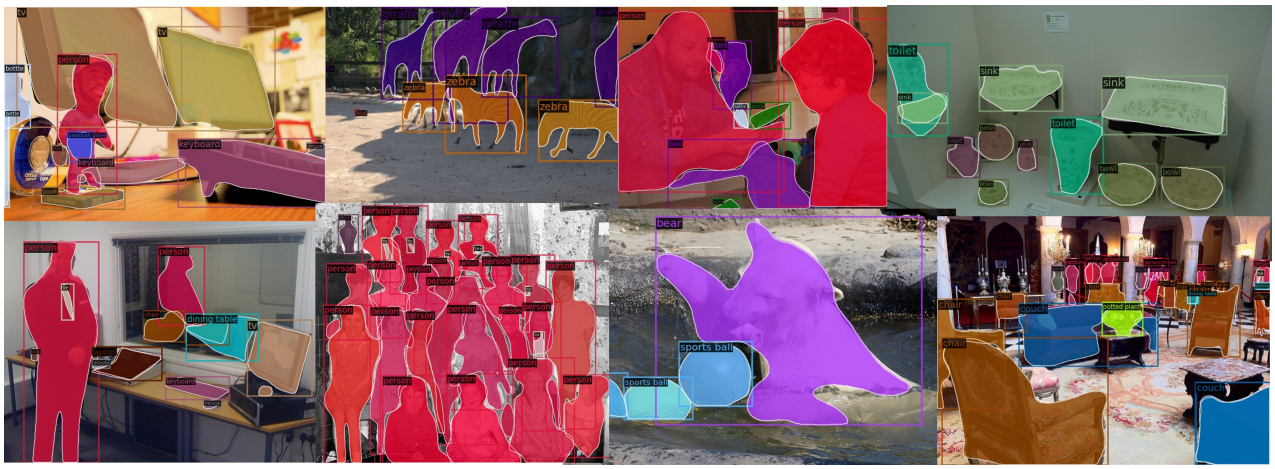

Figure 5. Visualization on instance segmentation task of our method.

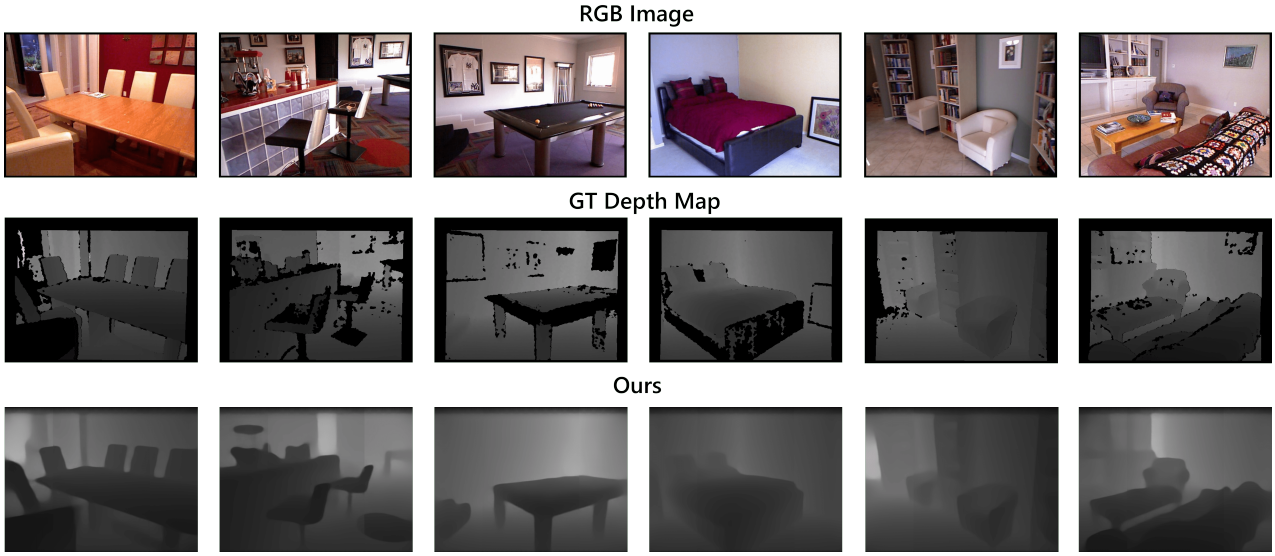

Figure 6. Visualization of depth estimation task. With the well-designed structure and techniques, our model is capable for this task.

UViM and Unified-IO. Moreover, our parallel approach further improves the performance to 0.275 RMSE. This result surpasses previous state-of-the-arts by 0.012 RMSE. While UViM and Unified-IO mainly conceptually propose unified frameworks for various visual tasks, we push more solid steps through in-depth study of the general visual task-solver.

4.6. One Model for Multiple Tasks

We train the instance segmentation and depth jointly using a shared task-solver. Table 10 shows that the joint training with shared model weights has marginal performance gradation compared to using separate task solvers.

5. Conclusion

In this work, we investigate the unification of output spaces for various vision tasks by a set of visual tokens, and further develop a unified auto-regressive encoder-decoder model. Two new techniques are proposed which take the particularity of visual tasks into account to improve the system: 1) Soft token can leverage the correlation between tokens to improve performance in the inference stage and enables end-to-end learning for the final visual targets; 2) Mask augmentation is used to alleviate the issue of corrupted/undefined areas of visual tasks, *i.e.* depth estimation. With these two techniques, our general method set a new state-of-the-art on the NYUv2 depth dataset, as well as competitive accuracy on the COCO 2017 object detection

Table 9. Results of monocular depth estimation task on NYUv2 [37]. Both AiT and AiT-P are our methods, where AiT indicates auto-regressive prediction, and AiT-P indicates parallel prediction.

Method	RMSE ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑	REL ↓	log10 ↓
DORN [15]	0.509	0.828	0.965	0.992	0.115	0.051
BTS [23]	0.392	0.885	0.978	0.995	0.110	0.047
AdaBins [4]	0.364	0.903	0.984	0.997	0.103	0.044
DPT [36]	0.357	0.904	0.988	0.998	0.110	0.045
LocalBins [5]	0.357	0.907	0.987	0.998	0.099	0.042
P3Depth [31]	0.356	0.898	0.981	0.996	0.104	0.043
BinsFormer [25]	0.339	0.921	0.989	0.998	0.096	0.041
NeWCRFs [50]	0.334	0.922	0.992	0.998	0.095	0.041
BinsFormer [25]	0.330	0.925	0.989	0.997	0.094	0.040
SwinV2-B [42]	0.303	0.938	0.992	0.998	0.086	0.037
SwinV2-L [42]	0.287	0.949	0.994	0.999	0.083	0.035
UViM [21]	0.467	-	-	-	-	-
Unified-IO$_{XL}$ [29]	0.385	-	-	-	-	-
AiT (SwinV2-B)	0.305	0.934	0.991	0.998	0.087	0.037
AiT (SwinV2-L)	0.284	0.949	0.993	0.999	0.079	0.034
AiT-P (SwinV2-B)	0.301	0.940	0.992	0.998	0.085	0.036
AiT-P (SwinV2-L)	**0.275**	**0.954**	**0.994**	**0.999**	**0.076**	**0.033**
AiT-P (SwinV2-L) w/o soft token	0.282	0.951	0.994	0.999	0.080	0.034

Table 10. Joint training of depth estimation and instance segmentation using a single task-solver. The performance of joint training is slightly worse compared to using separate task-solvers for each task.

Description.	Depth	Instance Seg.	
	RMSE	Box mAP	Mask mAP
separate training	**0.3052**	**43.3**	**34.2**
joint training	0.3103	42.2	34.1

and instance segmentation benchmark. We also conducted preliminary studies on the parallel decoder which is show promising results on the depth estimation problem.

A. Ablation on the Depths of VQ-VAE

In Table 3 and Table 4, we have studied how different widths of VQ-VAE affect the performance. In Table 11, we further examine the effect of depth, *i.e*. different number of residual blocks in VQ-VAE. The results show that increasing the number of residual blocks cannot bring more improvements and we find greater results variance when training tokenizer with deeper networks.

B. VQ-VAE *vs*. Interpolation Tokenizer

We use a lightweight VQ-VAE as the tokenizer to reduce the resolution and computation for the task-solver. A intuitive baseline to perform down-sampling is the interpolation method. For example, in instance segmentation, we can use a nearest-neighbor interpolation method to up-sample the binary masks. For depth estimation, we use a bilinear interpolation method and discretize the floating numbers between 0 to 10 by 128 bins. We select the same input and target resolution as our VQ-VAE tokenizer for a fair comparison. As shown in Table 12, the tokenizer using nearest-neighbor interpolation is much worse than our VQ-VAE method, showing the effectiveness of using VQ-VAE to represent the task output space.

Table 11. Ablation study on increasing the number of residual blocks on the 32× downsampling setting of depth estimation.

#Resblock	Tokenizor	Task-solver
	RMSE	RMSE
2	**0.0966**	**0.3080**
3	0.1055	0.3123
4	0.1082	0.3136
5	0.1033	0.3118

Table 12. Comparison of VQ-VAE and interpolation as the tokenizer.

Description.	Depth	Instance Seg.	
	RMSE	Box mAP	Mask mAP
Ours	**0.3080**	43.6	**33.2**
interpolation	0.7544	**43.8**	4.2

C. Details for Joint Training

The AdamW optimizer with a base learning rate of 8e-4, $\alpha_1 = 0.9$, $\alpha_2 = 0.999$, a weight decay of 0.05, and a layer decay of 0.85, a drop path rate of 0.3, and the linear decay learning rate scheduler are applied. We train the model with 300k iterations and batch size 64. The decoder loss weights of 5.0 and 1.0 are used for instance segmentation and depth estimation respectively. A depth loss weight of 0.2 is used. We process these batch data individually *i.e*. different tasks do task-specific augmentation and generate their discrete training token sequence.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022. 1, 2

[2] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022. 1

[3] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 2

[4] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In *CVPR*, pages 4009–4018. Computer Vision Foundation / IEEE, 2021. 3, 9

[5] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Localbins: Improving depth estimation by learning local distributions. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *ECCV*, volume 13661 of *Lecture Notes in Computer Science*, pages 480–496. Springer, 2022. 3, 9

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1, 2

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 7

[8] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021. 3, 4, 6

[9] Ting Chen, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J Fleet, and Geoffrey Hinton. A unified sequence interface for vision tasks. *arXiv preprint arXiv:2206.07669*, 2022. 1, 2

[10] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. Single-image depth perception in the wild. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Neurips*, pages 730–738, 2016. 3

[11] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1290–1299, 2022. 7

[12] Raul Diaz and Amit Marathe. Soft labels for ordinal regression. In *CVPR*, pages 4738–4747. Computer Vision Foundation / IEEE, 2019. 3

[13] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Neurips*, pages 2366–2374, 2014. 2

[14] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 2

[15] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, pages 2002–2011. Computer Vision Foundation / IEEE Computer Society, 2018. 3, 9

[16] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022. 2

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 3, 5

[18] Yasunori Ishii and Takayoshi Yamashita. Cutdepth: Edge-aware data augmentation in depth estimation. *CoRR*, abs/2107.07684, 2021. 3

[19] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021. 1, 2

[20] Doyeon Kim, Woonghyun Ga, Pyunghwan Ahn, Donggyu Joo, Sewhan Chun, and Junmo Kim. Global-local path networks for monocular depth estimation with vertical cutdepth. *CoRR*, abs/2201.07436, 2022. 3

[21] Alexander Kolesnikov, André Susano Pinto, Lucas Beyer, Xiaohua Zhai, Jeremiah Harmsen, and Neil Houlsby. Uvim: A unified modeling approach for vision with learned guiding codes. *arXiv preprint arXiv:2205.10337*, 2022. 2, 9

[22] Jaehan Lee and Chang-Su Kim. Monocular depth estimation using relative depth maps. In *CVPR*, pages 9729–9738. Computer Vision Foundation / IEEE, 2019. 3

[23] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *CoRR*, abs/1907.10326, 2019. 2, 9

[24] Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. *CoRR*, abs/2203.14211, 2022. 2

[25] Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. Binsformer: Revisiting adaptive bins for monocular depth estimation. *CoRR*, abs/2204.00987, 2022. 3, 9

[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 2

[27] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, pages 5162–5170. IEEE Computer Society, 2015. 3

[28] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12009–12019, 2022. 6

[29] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*, 2022. 2, 9

[30] Vladimir Nekrasov, Thanuja Dharmasiri, Andrew Spek, Tom Drummond, Chunhua Shen, and Ian D. Reid. Real-time joint semantic segmentation and depth estimation using asymmetric annotations. In *ICRA*, pages 7101–7107. IEEE, 2019. 3

[31] Vaishakh Patil, Christos Sakaridis, Alexander Liniger, and Luc Van Gool. P3depth: Monocular depth estimation with a piecewise planarity prior. In *CVPR*, pages 1600–1611. IEEE, 2022. 9

[32] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *CVPR*, pages 283–291. Computer Vision Foundation / IEEE Computer Society, 2018. 3

[33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1

[34] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020. 2

[35] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 2

[36] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12159–12168. IEEE, 2021. 2, 3, 9

[37] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision*, pages 746–760. Springer, 2012. 2, 9

[38] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 1, 2

[39] Lijun Wang, Jianming Zhang, Oliver Wang, Zhe Lin, and Huchuan Lu. Sdc-depth: Semantic divide-and-conquer network for monocular depth estimation. In *CVPR*, pages 538–547. Computer Vision Foundation / IEEE, 2020. 3

[40] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022. 1, 2

[41] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12193–12202, 2020. 3, 5

[42] Zhenda Xie, Zigang Geng, Jingcheng Hu, Zheng Zhang, Han Hu, and Yue Cao. Revealing the dark secrets of masked image modeling. *arXiv preprint arXiv:2205.13543*, 2022. 9

[43] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9653–9663, 2022. 6

[44] Dan Xu, Wei Wang, Hao Tang, Hong Liu, Nicu Sebe, and Elisa Ricci. Structured attention guided convolutional neural fields for monocular depth estimation. In *CVPR*, pages 3917–3925. Computer Vision Foundation / IEEE Computer Society, 2018. 3

[45] Ze Yang, Yinghao Xu, Han Xue, Zheng Zhang, Raquel Urtasun, Liwei Wang, Stephen Lin, and Han Hu. Dense reppoints: Representing visual objects with dense point sets. In *European Conference on Computer Vision*, pages 227–244. Springer, 2020. 3, 5

[46] Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *ECCV*, 2022. 3

[47] Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In *ICCV*, pages 5683–5692. IEEE, 2019. 3

[48] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022. 1, 2

[49] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021. 1, 2

[50] Weihao Yuan, Xiaodong Gu, Zuozhuo Dai, Siyu Zhu, and Ping Tan. New crfs: Neural window fully-connected crfs for monocular depth estimation. *CoRR*, abs/2203.01502, 2022. 3, 9

[51] Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *CVPR*, pages 4106–4115. Computer Vision Foundation / IEEE, 2019. 3

[52] Xizhou Zhu, Jinguo Zhu, Hao Li, Xiaoshi Wu, Hongsheng Li, Xiaohua Wang, and Jifeng Dai. Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16804–16815, 2022. 1, 2

[53] Daniel Zoran, Phillip Isola, Dilip Krishnan, and William T. Freeman. Learning ordinal relationships for mid-level vision. In *ICCV*, pages 388–396. IEEE Computer Society, 2015. 3