Unleashing Vanilla Vision Transformer with Masked Image Modeling for Object Detection

Yuxin Fang[1*] **Shusheng Yang**[1*] **Shijie Wang**[1*] **Yixiao Ge**[2] **Ying Shan**[2] **Xinggang Wang**[1†]

[1] School of EIC, Huazhong University of Science & Technology

[2] ARC Lab, Tencent PCG

Abstract

We present an approach to efficiently and effectively adapt a masked image modeling (MIM) pre-trained vanilla Vision Transformer (ViT) for object detection, which is based on our two novel observations: (i) A MIM pre-trained vanilla ViT encoder can work surprisingly well in the challenging object-level recognition scenario even with *randomly sampled partial* observations, *e.g.*, only 25% $\sim$ 50% of the input embeddings. (ii) In order to construct multi-scale representations for object detection from single-scale ViT, a *randomly initialized compact convolutional* stem supplants the pre-trained large kernel patchify stem, and its intermediate features can naturally serve as the higher resolution inputs of a feature pyramid network without further upsampling or other manipulations. While the pre-trained ViT is only regarded as the 3^{rd}-stage of our detector's backbone instead of the whole feature extractor. This results in a ConvNet-ViT *hybrid* feature extractor. The proposed detector, named MIMDET, enables a MIM pre-trained vanilla ViT to outperform hierarchical Swin Transformer by 2.5 AP^{box} and 2.6 AP^{mask} on COCO, and achieves better results compared with the previous best adapted vanilla ViT detector using a more modest fine-tuning recipe while converging 2.8$\times$ faster. Code and pre-trained models are available at https://github.com/hustvl/MIMDet.

1 Introduction

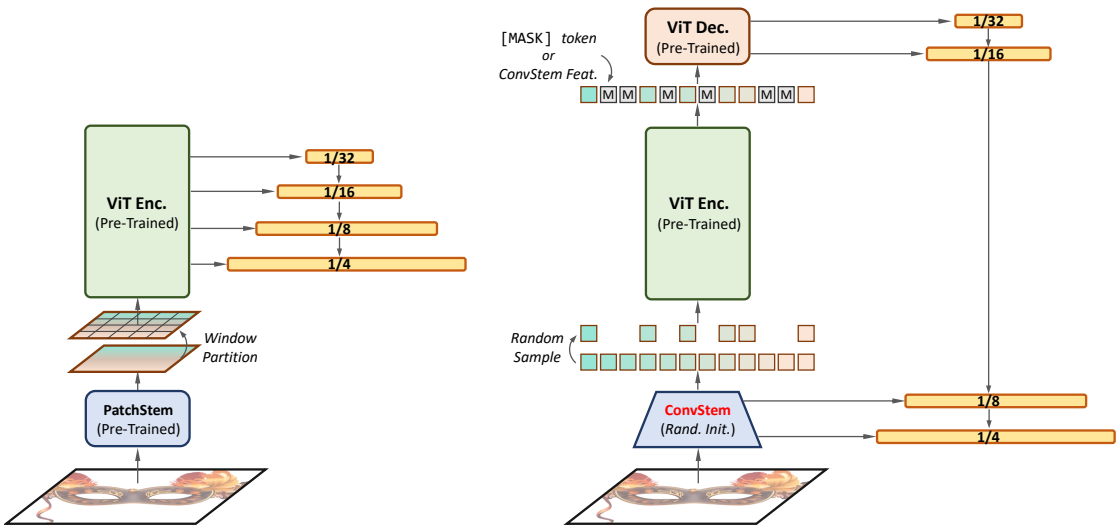
Transformer [52] is born to transfer. Entering the 2020s, Vision Transformer (ViT) [15] is rapidly transferring the viewpoint of machine vision in both architectural design [16, 24, 34, 54, 62] as well as representation learning [5, 18, 22, 61, 63]. From the perspective of architectural design, Swin Transformer [34], as a representative, seamlessly incorporates the local window attention into a hierarchical macro architecture. The window attention enables Swin Transformer to efficiently process high-resolution inputs with feasible costs, and the inherent pyramidal feature hierarchy can better handle the large variations of visual entities' scales and sizes, which is crucial for object-level understanding. Meanwhile, from the perspective of general visual representation learning, masked image modeling (MIM) [5, 22] pre-trained vanilla ViT[1] demonstrates promising scalability and is superior to the supervised isotropic or hierarchical ViT counterparts when transferred to image as well as video classification tasks [4, 14, 55, 63] with low-resolution inputs.

The compelling transfer learning performance of MIM pre-trained ViTs in image- and video-level recognition tasks motivate us to ponder: Is it possible for the more challenging *object* or *instance*-level recognition tasks, *e.g.*, object detection and instance segmentation, to also benefit from the powerful

[*]Equal contribution. [†]Corresponding author (xgwang@hust.edu.cn). This work was done when Shusheng Yang was interning at ARC Lab, Tencent PCG.

[1]For disambiguation, in this paper, "vanilla ViT" refers to the Vision Transformer proposed by Dosovitskiy et al. [15] with the isotropic, single-scale architecture unless specified.

Preprint. Under review.



(a) Other Approaches **(b) Our MIMDet**

Figure 1: Overview of our MIMDET and comparisons with previous representative approaches adapting vanilla ViT for object detection (*e.g.*, Li et al. [29]). In MIMDET, a randomly initialized compact convolutional stem (ConvStem) replaces the pre-trained large kernel patchify stem (PatchStem), and the ViT encoder only receives and processes sampled partial input embeddings. The intermediate features of ConvStem can directly serve as the higher resolution inputs for a standard FPN [32].

MIM pre-trained representations? Unfortunately, it is quite slow for vanilla ViT to directly process high-resolution images required by object-level recognition, for the complexity of global attention scales quadratically with the spatial dimension. One way is to re-introduce window attention or its variants during the fine-tuning stage, which can help reduce the attention's costs [29]. However, using window attention causes a *discrepancy* between MIM pre-training and fine-tuning, as the window partition implicitly treats inputs as *2D continuous regular grids*, while the vanilla ViT is pre-trained to process *1D partial sequences*.

Inspired by the MIM pre-training [5, 22], this work pursues a radical solution to transfer a vanilla ViT for object-level recognition (Figure 1b): We feed the MIM pre-trained ViT encoder with *only a partial input*, *e.g.*, only 25% $\sim$ 50% of the input sequence of embeddings with *random sampling*, during the object detection fine-tuning stage. The output sequence fragments are then complemented with learnable tokens (*e.g.*, `[MASK]` tokens) and processed by a lightweight decoder to recover the full feature map. This seemingly bold approach turns out to be surprisingly good in terms of accuracy-resource trade-off. Our motivation is that: (i) For a long time, the 2D *continuous* regular grid is considered as the *de facto* input from of Convolutional Neural Networks (ConvNets) [27] as well as hierarchical ViTs in visual understanding. Unlike them, vanilla ViT treats the input as a sequence of individual tokens / embeddings. Therefore it is feasible for ViT to process *nonconsecutive* input subsets, which serves as the foundation of our approach. (ii) Visual signal has heavy spatial redundancy by nature, which encourages the recent MIM pre-training approaches to adopt a very high masking ratio (*e.g.*, 40% $\sim$ 75%) [5, 22]. The successful practice of generative MIM pre-training implies that it is also possible to perform holistic visual understanding or recognition given only an input subset, as we believe visual generation and recognition are two sides of the same coin in essence. Since our approach also encourages ViT to implicitly conduct a kind of MIM during fine-tuning, it introduces a smaller gap between upstream MIM pre-training and downstream detector fine-tuning compared with the window attention and its variants.

Another obstacle is the lack of pyramidal feature hierarchy in vanilla ViT, as well-established visual recognition task layers [21, 32, 33, 49] usually require multi-scale inputs. In this work, we don't aim to re-design specific task layers tailored for single-scale ViT, instead, we make minimal adaptations to re-introduce multi-scale representations from vanilla ViT for better leveraging the precious legacy of visual understanding study. Different from previous approaches, which treats the vanilla ViT as the *whole* feature extractor / backbone and artificially manipulates intermediate features to compromise with FPN [32], we consider ViT only as a *part* of a hierarchical backbone, *i.e.*, the 3^{rd}-stage, and a *randomly initialized neat convolutional* stem (ConvStem) supplants the pre-trained large kernel patchify stem as the 1^{th}- and 2^{nd}-stage. The newly introduced ConvStem is very compact (only 4.1M,

less than 5% of the ViT-Base encoder's size), and its intermediate features can directly serve as the higher resolution inputs of a feature pyramid network. The resulting feature extractor is essentially a ConvNet-ViT *hybrid* architecture with a neat ConvStem as the earlier stage and a strong ViT as the later stage, which is also consistent with the current trends of the state-of-the-art visual encoder design methodology that tries to combine the strengths of two kinds of layers [11, 53].

The odyssey of object detection research is like "a goose that laid the golden eggs" that can always inspire and generalize well to the study of other visual understanding tasks. Therefore we choose the ubiquitous & canonical Mask R-CNN [21] as a touchstone for our design. The detector, named MIMDET (**M**asked **I**mage **M**odeling for **Det**ection, Figure 1b), enables a MIM pre-trained vanilla ViT-Base model (128M) to obtain 51.7 AP^{box} and 46.1 AP^{mask} on the COCO dataset [31], outperforming the hierarchical Swin Transformer counterpart pre-trained on ImageNet-1K [45] with supervision by 2.5 AP^{box} and 2.6 AP^{mask}. Moreover, our approach can achieve even better results compared with the previous best adapted vanilla ViT detector [29] based on the same MIM pre-trained representation [22] using a more modest fine-tuning recipe while being 2.8$\times$ faster in terms of converge speed. We also observe a promising scaling trend of our approach.

Our study indicates that *designing* and *pre-training* specific feature extractors for visual recognition may no longer be *indispensable* given the strong representation inside vanilla ViT, as long as we find the right way to unleash it. We believe a trend in future machine vision research is to better leverage the pre-trained representation via delicately adapting it using simple task layers. These observations are also aligned with those witnessed in natural language processing (NLP) [13, 40–42], and we hope this work can encourage the vision community to explore a similar trajectory.

2 Method

The goal is to tame and unleash a MIM pre-trained vanilla ViT to achieve favorable performance in object-level recognition with feasible costs. To this end, we propose to (i) feed ViT with only the sampled partial inputs instead of the full input set, described in §2.1, and (ii) use a randomly initialized compact convolutional stem to replace the pre-trained patchify stem for a better pyramidal feature hierarchy, detailed in §2.2. Figure 1b illustrates our approach.

2.1 You Only Look at One *Partial* Sequence

Object-level recognition tasks usually benefit from higher resolution inputs, which are in general over one order of magnitude higher than the input size of image classification. While the vanilla ViT computes global attention for spatial features aggregation, and the computational and memory costs of global attention scale quadratically with the input resolutions. Hence the fine-tuning process will be largely slowed down if a vanilla ViT is directly fed with the full input set.

We notice that vanilla ViT treats the input as a sequence of individual elements, therefore it is possible for a vanilla ViT to receive and process only a *partial*, *discontinuous* input sub-sequence given positional information (via position embeddings), which is intrinsically different from ConvNets as well as hierarchical ViT counterparts that manipulating on 2D *continuous* gird inputs. This property of vanilla ViT encourages us to act bold: we *randomly sample* a subset of patch embeddings serving as the input set of a MAE [22] pre-trained vanilla ViT encoder, *i.e*., the encoder only looks at one *partial* input sequence. Surprisingly, we find that with only 25% of the input sequence for the ViT-Base encoder, our detector can already achieve a very competitive accuracy that outperforms an augmented Swin Transformer under the same fine-tuning procedure. Furthermore, with 50% of the input, MIMDET can yield 51.7 AP^{box}/ 46.1 AP^{mask}, outperforming Swin by 2.5 AP^{box}/ 2.6 AP^{mask}.

To our knowledge, there is little literature to demonstrate that the challenging object-level recognition can be successfully done with only randomly sampled partial inputs. Our motivation is:

(i) The visual signal is highly redundant and loosely spans the spatial dimension, *e.g*., a vanilla ViT is able to recover the missing contextual information based only on a small set of visible content during MIM pre-training [5, 22, 61], which implies it *understands* the *global* context before generation. Therefore it is possible to perform global visual understanding in complex scenes based on strong pre-trained representations given only partial observations.

(ii) Our approach introduces a smaller gap between MIM pre-training and fine-tuning, *i.e*. we fine-tune the ViT in a similar vein as pre-training. As during pre-training, ViT learns a

pretext task that requires global contextual reasoning with only a visible subset as inputs. Our fine-tuning process mimics the MIM pre-training that conducts global object-level understanding with only partial observations based on high-capacity representations.

This solution enables us to achieve a win-win scenario: it optimizes the accuracy-resource trade-off while introducing a smaller pre-training & fine-tuning gap as well as leveraging the pre-trained representations more judiciously. The output sequence fragments are then complemented with learnable tokens (special [MASK] tokens or feature embeddings, please refer to Table 6 for details), and processed by a lightweight MAE pre-trained ViT decoder (*e.g.*, only 4× Transformer layers with reduced embedding dimension) to recover the full image feature.

Another way to adapt vanilla ViT for high-resolution input is to re-introduce window attention or its variants during the fine-tuning stage [29], which can reduce the attention's compute. More importantly, this kind of adaptation still treats the input as 2D continuous regular grids as ConvNets and hierarchical ViTs therefore also introducing 2D inductive biases, which is in principle beneficial to object- / region-level understanding. We also agree that the fine-tuning process would benefit from task-specific prior knowledge, especially given relatively less data available than pre-training. However, using window attention causes a discrepancy between pre-training and fine-tuning to some extent, as the vanilla ViT is pre-trained to process 1D partial sequences and the window partition along with its inductive biases never appears in the pre-training stage. In the next section, we present a smarter way to inject 2D inductive biases into our detector.

2.2 You Only Pre-train the *Third* Stage

Introducing ConvStem for Hierarchical Features Construction. Well-established visual understanding task layers usually receive *multi-scale* features as inputs to deal with the large scale and size variations of objects, while the MIM pre-trained *single-scale* vanilla ViT demonstrates compelling scalability and transfer learning performances in the image- and video-level recognition tasks compared with its hierarchical counterparts [5, 15, 22, 55]. Therefore it is quite promising for an object detector to enjoy the best of two worlds. However, the multi-scale features do not naturally exist in a vanilla ViT. If we can find a sensible way to re-introduce the pyramidal feature hierarchy for it, then re-design specific task layers tailored for the single-scale ViT is no longer needed, and the heritage of visual recognition study can be largely inherited.

Trace to its source, the lack of the feature hierarchy in vanilla ViT is rooted in its early visual signal processing, which is too aggressive for a detector: ViT "patchifies" the input image into 16×16 non-overlapping patches and then embeds them to form the Transformer encoder's input. The downsampling rate is so high that a visual entity of interest in the input image smaller than 16×16 pixels will no longer exist in the spatial dimension of Transformer encoder's input embeddings. Artificially dividing vanilla ViT into multiple stages and upsampling the downsampled intermediate feature [1, 29] is kind of a compromise to construct a feature pyramid for the detector, since there is no explicit evidence that those disappeared visual entities as well as the low-level details in the spatial dimension will re-appear in their original location faithfully (please also refer to Figure 2), especially when typical region-based detectors extract and process object's features based on spatial feature alignment as well as translation & scale equivariance [21, 44].

To mitigate the aforementioned issues, our solution is to *throw away* the pre-trained large-stride patchify stem, and use a *randomly initialized neat convolutional* stem (ConvStem) as a replacement. We adopt a minimalist ConvStem design by simply stacking 3×3 convolutions with a stride of 2 and doubled feature dimensions. Each convolutional layer is followed by a layer normalization [2] and a GELU activation [23]. The detailed architectural configurations are given in the Appendix. Our ConvStem progressively reduces the spatial dimension as well as enriches the channel dimension. The output embedding, which has the same shape as the original patchified embedding, serves as the input (before the random sampling process described in §2.1) for the ViT encoder.

The newly introduced ConvStem is very compact, in terms of the model size, our ConvStem only has 4.1M parameters, which is less than 5% of the ViT-Base encoder's size. Small as it is, the pyramidal feature hierarchy naturally exists in our ConvStem's intermediate layers. We select the features with strides of {4, 8} pixels with respect to the input image as the input features of a standard FPN's first two stages (*i.e.*, the input of P_2 and P_3) [32], while the output of pre-trained ViT decoder (with a stride of 16, detailed in §2.1) serves as the input of FPN's P_4, and the input for P_5 is simply obtained

via a parameter-less mean pooling upon the output of ViT decoder. Now, we successfully obtain a feature pyramid network for object detection.

A ConvNet-ViT Hybrid Architecture. Previous attempts regard the pre-trained ViT as the *whole* feature extractor [1, 29], while we treat the ViT as only a *part* of it, *i.e.*, the 3^{rd}-stage. In essence, our feature extractor turns out to be a ConvNet-ViT *hybrid* architecture with a shallow & neat ConvNet / ConvStem as the earlier stage and a deep & strong ViT as the later stage. This is also consistent with current trends of the state-of-the-art visual encoder design methodology that tries to combine the strengths of two kinds of architectures [11, 53], *i.e.*, the ConvNet / ConvStem is more suitable for early visual signal processing, and introduces 2D inductive biases for the ViT encoder & detector, while the single-scale vanilla ViT is more scalable and tends to have a larger model capacity.

Notice that our ConvStem is used only during fine-tuning and does not need to be pre-trained, which is different from Xiao et al. [58]. In fact, the ConvStem cannot be pre-trained via MIM and also does not support MIM pre-training of any visual encoder, since dense-slid convolutions with kernel size larger than stride propagate information across tokens, causing information leakage and impeding the MIM. This work shows that the pre-trained early stage is not a *sine qua non* for a detector to achieve favorable performances during fine-tuning, *i.e.*, you only need to pre-train the 3^{rd}-stage.

3 Experiment

General Setting. We conduct our experiments on the COCO dataset [31] using the `Detectron2` library [57]. Models are trained on the `train2017` split and evaluated on the `val2017` split. For MIMDET, we perform experiments mainly on ViT-Base / MIMDET-Base model using 32× 32G V100 GPUs with a total batch size of 64 optimized by AdamW [36] with a learning rate of 8e-5. We initialize the vanilla ViT part via MAE [22] pre-trained weight on ImageNet-1K [45]. An augmented Mask R-CNN [21] with FPN [32] is chosen as the detection task layer following Li et al. [29]. Since the vanilla ViT encoder is already pre-trained while the task layer is trained from scratch, the learning rate of the ViT encoder part is divided by a factor of 2 and the learning rate for the task layer is multiplied by 2. We use a modest fine-tuning recipe following Swin Transformer [34], which is a 36-epoch schedule using multi-scale training (scale the shorter side in [480, 800] while the longer side is smaller than 1333) and random crop augmentation. "pt" & "ft" means pre-training & fine-tuning. The detailed settings and configurations are in the Appendix.

For the evaluation metrics, we report AP^{box} for object detection and AP^{mask} for instance segmentation, with a particular focus on the fine-tuning epochs / wall-clock time, as we care about *how efficiently* a set of *general upstream* visual representations can be adapted to a *specific downstream* task.

3.1 Study and Analysis

We study and analyze the main properties of MIMDET-Base via ablating the default configuration.

training sampling ratio (↓)	inference sampling ratio (↓) 12.5%	25%	50%	75%	100%	training time	training mem†
12.5%	43.0 / 38.7	46.1 / 41.7	47.5 / 42.9	47.9 / 43.2	47.6 / 43.0	16.0 h	14.2 G
25%		46.8 / 42.2	49.4 / 44.2	50.0 / 44.7	49.9 / 44.7	16.5 h	15.9 G
50%			49.5 / 44.3	51.0 / 45.2	51.5 / 46.0	20.3 h	19.9 G
75%				51.0 / 45.6	51.8 / 46.3	27.2 h	28.5 G
100%					51.8 / 46.2	31.0 h‡	43.2 G‡

Table 1: **Random sampling ratio for training and inference.** Numbers of cells in the upper triangular represent "AP^{box}/ AP^{mask}". †: measured with batch size of 1. ‡: measured on A100 GPUs.

Sampling Ratio and Type for ViT Encoder. We study different random sampling ratio combinations for training & inference in Table 1. In general, our MIMDET can achieve *better* performance if the inference sampling ratio is *higher* than the training sampling ratio.

Specifically, under the scenario of inference with full input, we find with only 25% of the input for the ViT encoder during training, MIMDET can already achieve a very compelling performance that outperforms the upgraded Swin Transformer (detailed in §3.3 and Table 9).

1st-stage ft	2nd-stage ft	AP^{box}	AP^{mask}
50% rand	✗	51.5	46.0
50% rand	full set	51.4	46.0

Table 2: **Study of additional fine-tuning with full input set.** Fine-tuning once with a sampling ratio of 50% is sufficient.

training	inference	AP^{box}	AP^{mask}
50% rand	full set	51.5	46.0
50% grid	full set	48.7	44.0

Table 3: **Random sampling *vs*. grid sampling.** Random sampling generalizes well in the full input set inference scenario.

Furthermore, with 50% of the input for the ViT encoder during training, MIMDET obtains very competitive results which are similar to the 100% input training & inference setting, striking a good trade-off between accuracy and resources. From another perspective, in Table 2 we show that adding an additional short fine-tuning stage (6-epoch) with full inputs after the fine-tuning procedure (36-epoch) with sampled inputs does *not* help further improve the accuracy. These results indicate that training with 50% randomly sampled inputs is sufficient to achieve a satisfactory performance.

The compelling results of using only 50% input for training are not a coincidence, as we believe it is closely related to the use of ConvStem. Specifically, the receptive field size of our ConvStem is 31, approximately 2× of the stem's output (also the ViT encoder's input) feature's stride. That means if we sample with a stride of 2 on the input feature of ViT encoder (*i.e*., grid sampling with a ratio of 50%), the sampled feature map can almost cover all locations of the original input image. In practice, we find uniform random sampling generalizes much better than grid sampling in the full input set inference scenario, as shown in Table 3.

# eval × ratio	1×100%	1×50% rand	2×50% rand	4×50% rand	8×50% rand
AP^{box}/ AP^{mask}	51.5 / 46.0	49.5 / 44.3	50.3 / 44.8	50.9 / 45.3	51.0 / 45.2

Table 4: **Study of different inference strategies.** "# eval × ratio": the ensemble result of several independent evaluations with a specific sampling ratio, *e.g*., "2×50% rand" means the ensemble accuracy of inference twice with a random sampling ratio of 50%.

Inference Strategy. We study the appropriate inference strategy for MIMDET in Table 4. Under the scenario of inference with sampled inputs, we find the ensemble of more trials generally improves the performance compared with inference only once with sampling. Meanwhile, inference with the full input set, which can be also regarded as *an ensemble of input features* for ViT encoder since it is pre-trained with only partial observations, is more performant. These results imply that the ensemble of input features works better than the ensemble of output results for our approach.

To summarize, results in Table 1, 2, 3 and 4 indicate that our default training and inference strategies are nearly optimal.

ConvStem Type. In Table 5, we study whether increasing the capacity of the convolutional earlier stage is beneficial to the detection performance, and the answer is negative. We choose the recent state-of-the-art ConvNeXt's [35] earlier stage design[2] as a replacement of our default / naïve design, and we observe no further improvement. This implies that for the earlier stage, its convolutional property matters more than its strength. As the FPN can help fuse different-level features to have strong semantics at all scales [32], the expressiveness of high-resolution features for object-level recognition shouldn't be a worry.

Decoder Input Feature. Since we only encode partial inputs, to obtain the full set of features, we need to fill in all unsampled locations for the decoder as well as the task layer, as studied in Table 6. One straightforward solution is to fill the blank with [MASK] tokens, as the decoder is pre-trained to process them. While we find using the ConvStem output feature is slightly better.

Using ConvStem features at unsampled locations is an analogy of *stochastic depth* [25], as we "randomly drop a subset of layers (*i.e*., all unsampled embeddings of the ViT encoder) and bypass them with the identity function (*i.e*., ConvStem features) [25]." Therefore, this training and inference strategy also aligns with the motivation of "train *short* networks and use *deep* networks at test time." as well as "an implicit *ensemble* of network of different depths [25]." Since using the ConvStem feature brings nearly cost-free improvement, we use it for comparisons with other leading approaches in Table 9.

[2]We directly adopt the configuration / pre-trained weights of the first two stages of ConvNeXt-Large, since this design has matched model size as well as output channels.

ConvStem	pt?	AP^{box}	AP^{mask}
naïve	✗	51.5	46.0
ConvNeXt [35]	✗	51.3	45.8
ConvNeXt [35]	✓	51.4	45.8

Table 5: **Study of the ConvStem type.** The naïve design (§2.2) with random initialization is sufficient.

dec input feat	AP^{box}	AP^{mask}
[MASK] token	51.5	46.0
ConvStem feat	51.7	46.1

Table 6: **Study of the decoder input feature type of all unsampled positions.** ConvStem feature is slightly better.

# dec layers	AP^{box}	AP^{mask}	# params	training mem†	training time
1	49.9	44.6	1.00× (118 M)	1.00× (12.2 G)	1.00× (15.5 h)
2	50.6	45.2	1.03× (121 M)	1.18× (14.4 G)	1.12× (17.3 h)
4	51.5	46.0	1.08× (127 M)	1.63× (19.9 G)	1.16× (20.3 h)
8	51.6	46.1	1.18× (140 M)	2.39× (29.2 G)	1.74× (27.0 h)

Table 7: **Study of the number of decoder layers.** †: measured with batch size of 1.

Number of Decoder Layers. We study the impact of the number of MAE pre-trained decoder layers in Table 7. The lightweight decoder is used to recover full features given encoded partial observations. We find a decoder with 4 layers achieves the best trade-off. If we randomly initialize the decoder, the training diverges using our default configuration.

It is usually believed that the MAE encoder learns general representations that transfer well, while by analogy with BERT [13] from NLP, we believe the MAE decoder is actually trained to be a BERT encoder, and the MAE encoder is more like a BERT tokenizer that maps the raw input signal to BERT encoder's input embeddings. To our knowledge, MIMDET is the first work to leverage the MAE pre-trained decoder in downstream tasks. What the MAE decoder learns during pre-training is still unclear, and the property of it deserves more attention in future research.

	hybrid FPN?	rand sample?	# dec layers	AP^{box}	AP^{mask}	# params	time
row1	✓	✓	4	51.5	46.0	127 M	20.3 h
row2	✓	✓	1	49.9	44.6	118 M	15.5 h
row3	✓	✗	0	49.5	44.5	113 M	17.5 h
row4	✗	✗	0	48.9	43.9	120 M	18.4 h

Table 8: **Study of the ViT's & FPN's input form.** "hybrid FPN?": the FPN's P_2 & P_3 features come from ConvStem's intermediate features (✓) or ViT's intermediate features [1, 29] (✗). "rand sample": the ViT encoder's input comes from the 50% randomly sampled ConvStem's output (✓) or window partitioned ConvStem's output with a window size of 7×7 [29, 34] (✗).

ViT's & FPN's Input Form. In Table 8, we conduct ablation studies on MIMDET-Base with only one decoder layer (row2) to align with the budgets of row3 and row4.

row2 & row3 show that randomly sampled inputs can achieve better performance than window partitioned inputs [29, 34] in MIMDET given similar budgets. Also notice that row3 is a counterpart of Swin Transformer[3]. Compared with the well-established Swin+ in Table 9 (49.2 AP^{box}/ 43.5 AP^{mask}), row3 with a ConvNet (rand.init.) & ViT (pre-trained) hybrid architecture can obtain higher accuracy, indicating that pre-training specific feature extractor for visual recognition may no longer be indispensable given the strong representation inside vanilla ViT.

row3 → row4 demonstrates that the performance will suffer if the higher resolution inputs of FPN are from the intermediate features of ViT encoder [1, 29] instead of ConvStem, which implies the convolutional feature is more beneficial to object-level understanding tasks. We show some qualitative results in the next section that can help us gain an intuitive sense.

3.2 Visualization

Figure 2 visualizes some backbone & FPN feature maps with a stride of 4 for both Li et al. [29] and our MIMDET. The stride-4 backbone feature of Li et al. [29] is obtained from a stride-16 ViT encoder feature via upsampling using two stride-2 transposed convolutions with 2×2 kernel. The resulting features suffer from very strong "checkerboard artifacts [38]". If we look closer, the evidence of ViT

[3]The #Transformer layers of different stages in Swin-Base is {2, 2, 18, 2}. Therefore both Swin-Base and MIMDET-Base in row3 of Table 8 have a deep & strong 3^{rd}-stage with window attention, as well as a shallow & compact early stage.

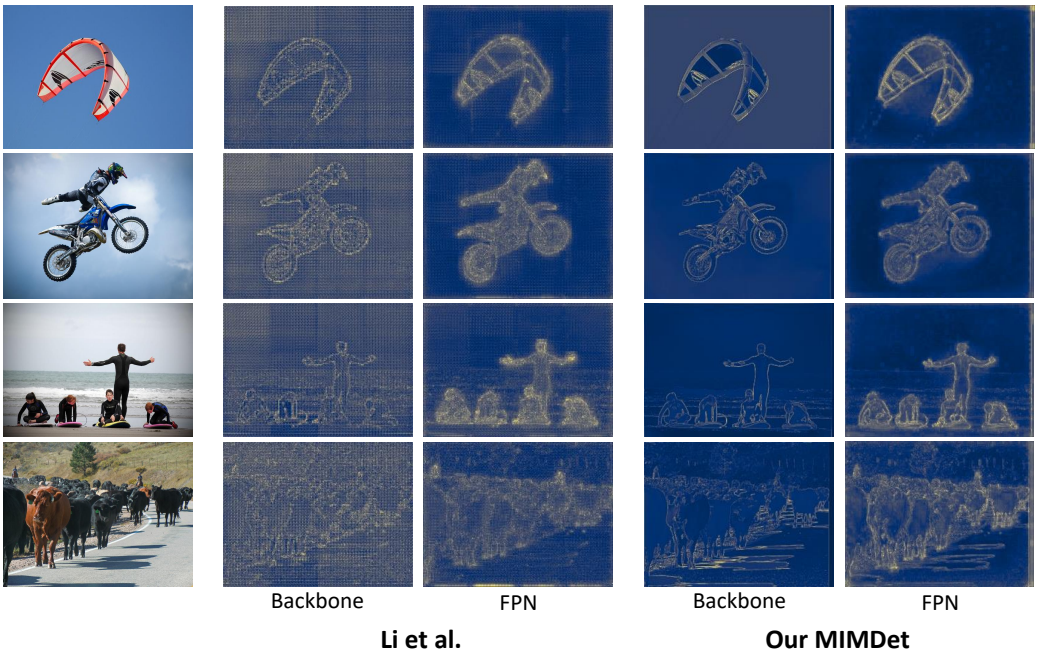


Figure 2: **Visualizations and comparisons of some stride-4 backbone and FPN feature maps.** The feature maps of Li et al. [29] is obtained from our re-implementation which successfully reproduces its reported results.

attention's window partition emerges. Thanks to FPN, the noise can be mitigated to some extent. However, many low-level details are still fuzzy. On the other hand, our ConvStem in MIMDET can always produce clear and tidy features, which is beneficial to both the ViT encoder as well as the Mask R-CNN detector. More visualizations are available in the Appendix.

3.3 Comparisons with Previous Approaches

Table 9 shows the comparisons. Our MIMDET can successfully adapt a MAE pre-trained representation to achieve strong AP^{box} and AP^{mask}, better than some representative well-established hierarchical ViT architectures [30, 34] under the same fine-tuning procedure. MIMDET-Base (Table 9a) without relative position biases can outperform the previous best adapted vanilla ViT [29] given the same initial representation [22] with a more modest data augmentation strategy (resize & crop *v.s.* LSJ), while only requiring 2.8× less epochs (36 ep *v.s.* 100 ep) to coverage and being 2.7× faster (20.3 h *v.s.* 54.0 h) in training[4]. Moreover, we observe a promising scaling trend of our approach, *i.e.*, our MIMDET-Large (Table 9b) without relative position biases can achieve 54.3 AP^{box}/ 48.2 AP^{mask}, outperforming the best large model in Li et al. [29] as well as strong hierarchical competitors [30].

3.4 Limitation and Discussion

A potential limitation is we only study one representative framework of MIM, *i.e.*, the MAE family [22] based on an asymmetric encoder-decoder design that allows the encoder to process only partial observations without [MASK] tokens during pre-training. Nevertheless, we believe MAE is a simple, strong and scalable MIM framework that would dominate visual pre-training over the next few years as BERT [13] in NLP. The MAE framework is also compatible with other advances in MIM pre-training [4, 14, 55], and there is a lot of evidence that MAE generalizes well to 2D visual multi-modal & multi-task [3], medical [64], video [51], 3D [39], RL [59] and even language [56] & audio [37] pre-training. Therefore we believe this work can contribute to a board research related to MIM / MAE study.

[4]Measurements and comparisons of the training time is based on our re-implementation of Li et al. [29], which successfully reproduces its reported results (ViT-Base: Our reproduced results: 50.4 AP^{box}/ 44.9 AP^{mask} *v.s.* Li et al. [29]'s: 50.3 AP^{box}/ 44.9 AP^{mask}).

[5]In previous literature [28, 30], Swin-Base with standard Mask R-CNN obtains 48.5 AP^{box}/ 43.4 AP^{mask}.

backbone	pt config	ft epochs	data aug	rel pos?	AP^{box}	AP^{mask}
● *representative hierarchical architecture*						
Swin+ [34]	sup-1K	36	resize & crop	✓	49.2	43.5
MViTv2 [30]	sup-1K	36	resize & crop	✓	51.0	45.7
● *adapted vanilla Vision Transformer*						
Li et al. [29]	MAE-1K	100	LSJ_{1024}	✓	50.3	44.9
MIMDET (Ours)	MAE-1K	36	resize & crop	✗	**51.7**	**46.1**

(a) Results of *base-sized* models.

backbone	pt config	ft epochs	data aug	rel pos?	AP^{box}	AP^{mask}
● *representative hierarchical architecture*						
MViTv2 [30]	sup-1K	36	resize & crop	✓	51.8	46.2
MViTv2 [30]	sup-21K	36	resize & crop	✓	52.7	46.8
● *adapted vanilla Vision Transformer*						
Li et al. [29]	MAE-1K	100	LSJ_{1024}	✓	53.3	47.2
MIMDET (Ours)	MAE-1K	36	resize & crop	✗	**54.3**	**48.2**

(b) Results of *large-sized* models.

Table 9: **COCO object detection and instance segmentation results.** We use Mask R-CNN as the task layer. "rel pos": using relative position biases [43, 46], which usually improves ∼1 AP but severely adds training time and memory. "Swin+": its Mask R-CNN is augmented following Li et al. [29][5]. "LSJ_{1024}": large scale jittering [19] on a 1024×1024 canvas. "sup-21K": pre-training using ImageNet-21K [12] with supervision.

Reproducibility. The code needed to reproduce the experimental results is in the supplemental material. We observe ∼0.2 AP fluctuation with respect to different random seeds, which is very common for the COCO dataset. We report the key results using the median of 3 independent runs.

4 Related Work

Hierarchical Backbone for Object Detection. Well-established object detectors [21, 32, 33, 44, 49, 50] usually take advantage of multi-scale features as inputs for better performance. The multi-scale inputs naturally exist in hierarchical ConvNet [20, 26, 47, 48, 60] / ViT [16, 30, 34, 54, 62] backbones. This work aims to adapt and unleash MIM pre-trained vanilla ViT's representations [5, 22, 61, 63] for object-level recognition without modifying its pre-training process and its architectural nature.

Taming Vanilla ViT for Object Detection. Since the introduction of Transformer [52] to computer vision [15], the effort of taming pre-trained vanilla ViT for object detection has never stopped. Beal et al. [7] is the first to adapt a supervised pre-trained ViT for object detection with a Faster R-CNN detector [44]. YOLOS [17] proposes to perform object detection in a pure sequence-to-sequence manner with a pre-trained ViT encoder only. Similar to YOLOS, we also treat the inputs for ViT encoder as 1D sequences instead of 2D grids. Li et al. [29] is the first work to conduct a large-scale study of vanilla ViT on object detection with powerful MIM pre-trained representations [5, 22], demonstrating the promising capability and capacity of vanilla ViT in object-level recognition. UViT [9] is a recent single-scale ViT detector with a *detection-oriented* design. Different from UViT, we aim to leverage the *general* representations from MIM for high-performance object detection. Another series of work [6, 10] explore the pre-training of DETR framework [8, 65].

5 Conclusion

In this paper, we explore how to unlock the potential of MIM pre-trained vanilla ViT for high-performance object detection and instance segmentation. The satisfactory results imply designing and pre-training specific feature extractors for visual recognition may no longer be a *sine qua non* process for computer vision research. As vanilla ViT demonstrates extremely strong model capacity, future state-of-the-art visual recognition systems shall learn to tame it and unleash it. These trends have already been witnessed in NLP [13, 40–42], and we hope our work can encourage the vision community to explore the powerful general visual representations hidden in the vanilla ViT.

References

[1] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *NeurIPS*, 2021.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Roman Bachmann, David Mizrahi, Andrei Atanov, and Amir Zamir. Multimae: Multi-modal multi-task masked autoencoders. *arXiv preprint arXiv:2204.01678*, 2022.

[4] 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.

[5] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022.

[6] Amir Bar, Xin Wang, Vadim Kantorov, Colorado J Reed, Roei Herzig, Gal Chechik, Anna Rohrbach, Trevor Darrell, and Amir Globerson. Detreg: Unsupervised pretraining with region priors for object detection. In *CVPR*, 2022.

[7] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[9] Wuyang Chen, Xianzhi Du, Fan Yang, Lucas Beyer, Xiaohua Zhai, Tsung-Yi Lin, Huizhong Chen, Jing Li, Xiaodan Song, Zhangyang Wang, et al. A simple single-scale vision transformer for object localization and instance segmentation. *arXiv preprint arXiv:2112.09747*, 2021.

[10] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *CVPR*, 2021.

[11] Zihang Dai, Hanxiao Liu, Quoc Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *NeurIPS*, 2021.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019.

[14] Xiaoyi Dong, Jianmin Bao, Ting Zhang, Dongdong Chen, Weiming Zhang, Lu Yuan, Dong Chen, Fang Wen, and Nenghai Yu. Peco: Perceptual codebook for bert pre-training of vision transformers. *arXiv preprint arXiv:2111.12710*, 2021.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[16] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *ICCV*, 2021.

[17] Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. In *NeurIPS*, 2021.

[18] Yuxin Fang, Li Dong, Hangbo Bao, Xinggang Wang, and Furu Wei. Corrupted image modeling for self-supervised visual pre-training. *arXiv preprint arXiv:2202.03382*, 2022.

[19] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *CVPR*, 2021.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[21] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. In *ICCV*, 2017.

[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[23] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[24] Byeongho Heo, Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Junsuk Choe, and Seong Joon Oh. Rethinking spatial dimensions of vision transformers. In *ICCV*, 2021.

[25] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[27] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989.

[28] Youngwan Lee, Jonghee Kim, Jeff Willette, and Sung Ju Hwang. Mpvit: Multi-path vision transformer for dense prediction. In *CVPR*, 2022.

[29] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick. Benchmarking detection transfer learning with vision transformers. *arXiv preprint arXiv:2111.11429*, 2021.

[30] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Improved multiscale vision transformers for classification and detection. In *CVPR*, 2022.

[31] 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 *ECCV*, 2014.

[32] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[33] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[35] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022.

[36] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[37] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino. Masked spectrogram modeling using masked autoencoders for learning general-purpose audio representation. *arXiv preprint arXiv:2204.12260*, 2022.

[38] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. URL http://distill.pub/2016/deconv-checkerboard.

[39] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. *arXiv preprint arXiv:2203.06604*, 2022.

[40] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[41] 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 *ICML*, 2021.

[42] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 2020.

[44] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[45] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[46] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

[47] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[48] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[49] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020.

[50] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019.

[51] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv preprint arXiv:2203.12602*, 2022.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[53] Ashish Vaswani, Prajit Ramachandran, Aravind Srinivas, Niki Parmar, Blake Hechtman, and Jonathon Shlens. Scaling local self-attention for parameter efficient visual backbones. In *CVPR*, 2021.

[54] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[55] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. *arXiv preprint arXiv:2112.09133*, 2021.

[56] Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*, 2022.

[57] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[58] Tete Xiao, Piotr Dollar, Mannat Singh, Eric Mintun, Trevor Darrell, and Ross Girshick. Early convolutions help transformers see better. In *NeurIPS*, 2021.

[59] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[60] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[61] 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 *CVPR*, 2022.

[62] Weijian Xu, Yifan Xu, Tyler Chang, and Zhuowen Tu. Co-scale conv-attentional image transformers. In *ICCV*, 2021.

[63] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022.

[64] Lei Zhou, Huidong Liu, Joseph Bae, Junjun He, Dimitris Samaras, and Prateek Prasanna. Self pre-training with masked autoencoders for medical image analysis. *arXiv preprint arXiv:2203.05573*, 2022.

[65] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.

A Appendix

A.1 Implementation Details

Architecture 1 ConvStem for ViT-Base (PyTorch Style), which can help preserve low-level details, produce higher resolution hierarchical features for FPN, and introduce 2D inductive biases for the ViT encoder & detector.

```
# Number of Parameters: 4.1M.
ConvStem(
  ModuleList(
    (0): Sequential(
      (0): Conv2d(3, 96, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (1): LayerNorm2d(96, eps=1e-06, affine=True) & GELU()
    )
    (1): Sequential(
      (0): Conv2d(96, 192, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (1): LayerNorm2d(192, eps=1e-06, affine=True) & GELU() # Input for FPN P2.
    )
    (2): Sequential(
      (0): Conv2d(192, 384, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (1): LayerNorm2d(384, eps=1e-06, affine=True) & GELU() # Input for FPN P3.
    )
    (3): Sequential(
      (0): Conv2d(384, 768, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (1): LayerNorm2d(768, eps=1e-06, affine=True) & GELU()
      (2): Conv2d(768, 768, kernel_size=(1, 1), stride=(1, 1)) # Input for ViT-Base Enc.
    )))
```

Architecture of ConvStem. We adopt a minimalist ConvStem design, *i.e.*, by simply stacking 3×3 regular convolutions with a stride of 2 and doubled feature dimensions. Each convolutional layer is followed by a layer normalization [2] and a GELU activation [23]. The detailed configurations are given in Architecture 1.

Hyper-parameters and Model Configurations. Hyper-parameters and model configurations for fine-tuning on the COCO dataset are shown in Table 10. Since the vanilla ViT encoder is already pre-trained while the task layer is trained from scratch, the learning rate of the ViT encoder part is divided by a "lr multiplier" and the learning rate for the task layer is multiplied by a "lr multiplier".

backbone	hyper-parameters					model configs		
	lr	lr multiplier	weight decay	drop path	ft epochs	params (M)	FLOPs (G)	inf. time (s)
MIMDET-Base	$8e^{-5}$	2	0.1	0.1	36	128	933	0.29
MIMDET-Large	$8e^{-5}$	3.5	0.1	0.1	36	349	2082	0.58

Table 10: Hyper-parameters and and model configurations for COCO fine-tuning. We report the average number of FLOPs and inference time for the first 100 images in the COCO `val` set following [8] on a V100 GPU.

Optimization. The loss function of MIMDET keeps the *same* as the canonical Mask R-CNN [21, 29], *i.e.*, explicit reconstruction loss for ViT encoder is *not* needed during the fine-tuning, even though the encoder only receive partial observations. The implicit reconstruction process of ViT encoder is driven by the supervision from the Mask R-CNN detector.

A.2 More Visualizations

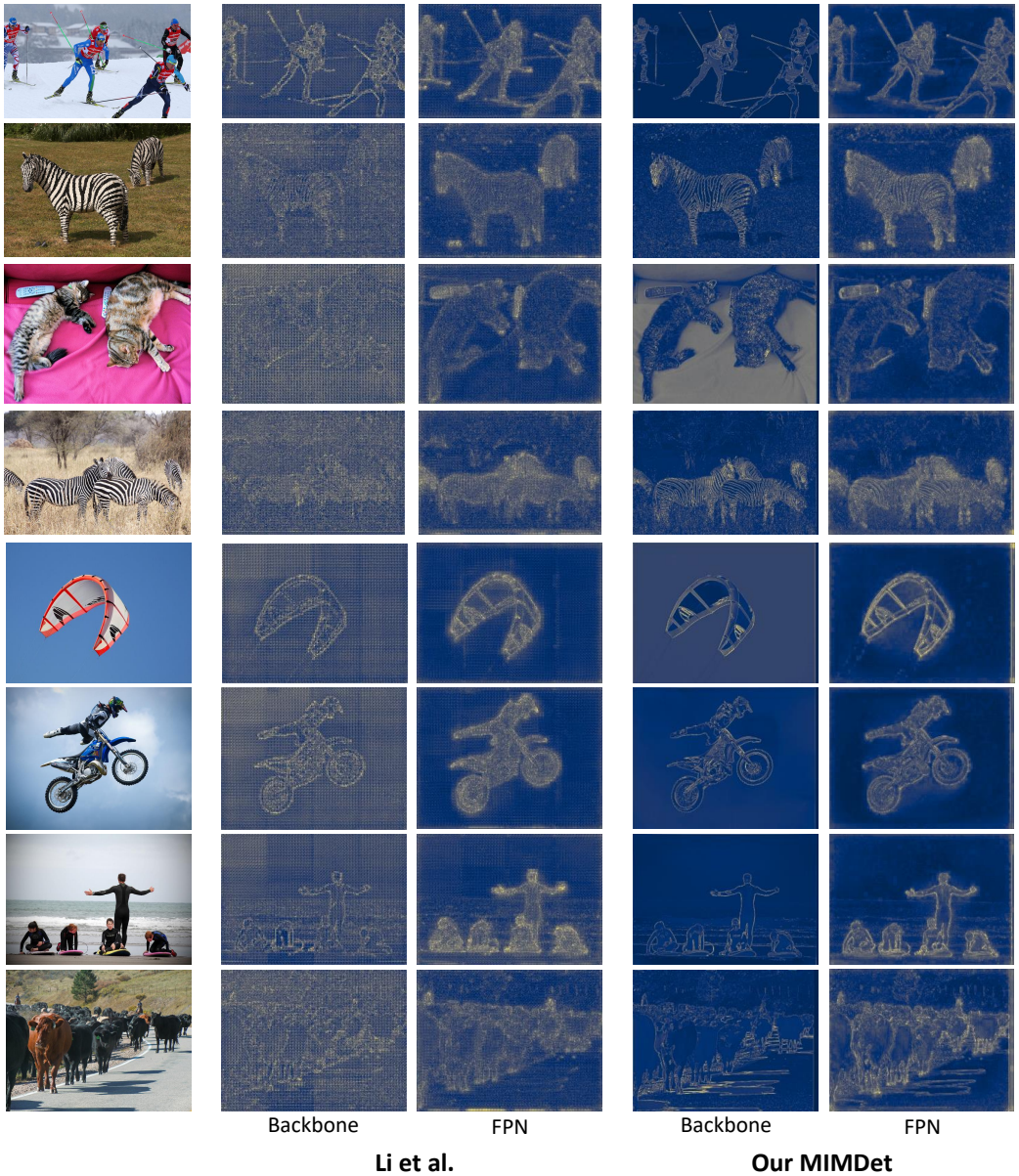


Figure 3: **More visualizations and comparisons of some stride-4 backbone and FPN feature maps.** The feature maps of Li et al. [29] is obtained from our re-implementation which successfully reproduces its reported results.