Self-Supervised Pretraining for 3D Vision Tasks by Cross-View Completion

Anonymous Author(s)
Affiliation
Address
email

Abstract

Masked Image Modeling (MIM) has recently been established as a potent pretraining paradigm. A pretext task is constructed by masking patches in an input image, and this masked content is then predicted by a neural network using visible patches as sole input. This pretraining leads to state-of-the-art performance when finetuned for high-level semantic tasks, *e.g.* image classification and object detection. In this paper we instead seek to learn representations that transfer well to a wide variety of 3D vision and lower-level geometric downstream tasks, such as depth prediction or optical flow estimation. Inspired by MIM, we propose an unsupervised representation learning task trained from *pairs* of images showing the same scene from different viewpoints. More precisely, we propose the pretext task of *cross-view completion* where the first input image is partially masked, and this masked content has to be reconstructed from the visible content and the second image. In single-view MIM the masked content often cannot be inferred precisely from the visible portion only, so the model learns to act as a prior influenced by high-level semantics. In contrast, this ambiguity can be resolved with cross-view completion from the second unmasked image, on the condition that the model is able to understand the spatial relationship between the two images. Our experiments show that our pretext task leads to significantly improved performance for monocular 3D vision downstream tasks such as depth estimation. In addition, our model can be, by its design, directly applied to binocular downstream tasks such as optical flow or relative camera pose estimation, for which we obtain competitive results without bells and whistles, *i.e.* using a generic architecture without any task-specific design.

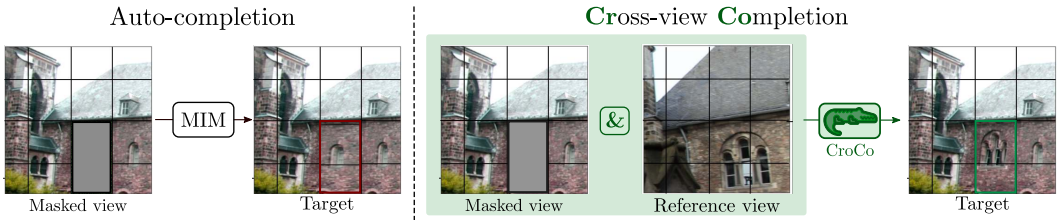

Figure 1: **Auto-completion *vs*. Cross-view completion tasks.** *Left:* given a masked image, a model trained for auto-completion can only leverage the visible context to fill-in the blanks and thus relies mainly on high-level semantic information. *Right:* given an additional view of the same scene, cross-view completion makes precise reconstruction possible, assuming that the model is able to understand both the scene geometry and the spatial relationship between the two images.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

1 Introduction

Self-supervised learning for model pretraining has allowed to achieve state-of-the-art performance in a number of high-level computer vision tasks, such as image classification or object detection. Contrary to traditional supervised learning, these models enable the use of unlabelled data via carefully designed pretext tasks. A popular method for self-supervision is instance discrimination [9, 10, 13, 22, 28, 30, 62] which constructs a pretext task by learning representations that are invariant to various data augmentations. More recently, Masked Image Modeling (MIM) [6, 12, 15, 25, 29, 43, 70] has emerged as a powerful alternative for self-supervision. Inspired by BERT models [18], these models are trained using an auto-completion pretext task. An encoder, usually a Vision Transformer (ViT) [20], takes a partial view of an image input, obtained by splitting the image into patches and masking some of them, and encodes it into a latent representation. The masked patches are then predicted by the decoder using the latent representation. To solve the task, the model must leverage the context given by the visible portion, and act as a prior for the ambiguous content that cannot be deduced from them. In practice these models are typically trained on object-centric datasets such as ImageNet and thus tend to learn high-level semantic information; that makes them well suited for tasks such as image classification or object detection [3, 29, 33].

In this paper, we propose instead a self-supervised training objective specially designed to learn 3D geometry from unlabeled data, named *Cross-view Completion*. Given two images depicting the same scene, random parts of the first input image are masked and then predicted by the model using both (1) the visible parts of this first image, as well as (2) a second image called *reference* image as it depicts the same scene from a different point of view (see Figure 1). While multi-view image completion has a long history in image editing [16, 69], we are the first to explore its potential as a self-supervised representation learning tool. In contrast to single-view completion, the proposed pretext task of cross-view completion allows to perform masked image modeling conditionally on a second view. In this case, most of the ambiguity can be resolved by reasoning about the scene geometry and spatial relationship between the two views. This is illustrated in the reconstruction examples of our model in Figure 2.

Figure 3 provides an overview of our self-supervised training model; we term it *CroCo*, which stands for Cross-view Completion. We divide both images into sets of non-overlapping patches, denoted as tokens. Most tokens from the first (target) image are randomly discarded, and the remaining ones are fed to an image encoder, which we implement using a Vision Transformer (ViT) backbone [20]. All patches from the second (reference) image are encoded in a *Siamese* manner, *i.e.*, using the same encoder with shared weights. The token latent representations output by the encoder from both images, including tokens to account for the masked patches of the target image, are then fed to a decoder whose goal is to predict the appearance of hidden patches. To this aim, we use a series of transformer decoder blocks comprising cross-attention layers. This allows non-masked tokens from the target image to attend tokens from the reference image, thus enabling cross-view comparison and reasoning. Our model is trained using a simple pixel reconstruction loss over all masked patches, similar to MAE [29]. To finetune the model on downstream tasks that process only a single image, *e.g.* monocular depth estimation, the decoder can be discarded and we use the pretrained encoder alone. In the case of binocular tasks such as optical flow estimation, the original CroCo architecture is used as is.

For pretraining, CroCo relies on pairs of images depicting the same scene; in this work the pairs were obtained from synthetic renderings of indoor scenes produced with the Habitat [50] simulator. We empirically show that high masking ratios, *e.g.* 90%, lead to the best pretraining performance. Our model is evaluated on monocular downstream tasks, in particular depth estimation on the NYUv2 dataset [51] and a diverse set of dense 2D and 3D regression tasks taken from Taskonomy [68]. We show that CroCo leads to significantly better performance compared to existing MIM models, whether they were pretrained on the same data or on ImageNet. When evaluated on monocular high-level semantic tasks, such as ImageNet classification, CroCo obtains lower performance compared to established MIM models, however. This essentially comes from our use of indoor scenes for pretraining, instead of highly semantic datasets like ImageNet [49], as empirically shown in our experiments. Lastly, we demonstrate that CroCo can be applied to binocular downstream tasks in a straightforward manner without bells and whistles. For instance, optical flow estimation can be performed by directly regressing 2 values per pixel using the decoder, and likewise, relative pose regression is achieved by simply appending a pose regression head to CroCo. In both cases, we show that CroCo pretraining leads to competitive results in these 3D vision downstream tasks.

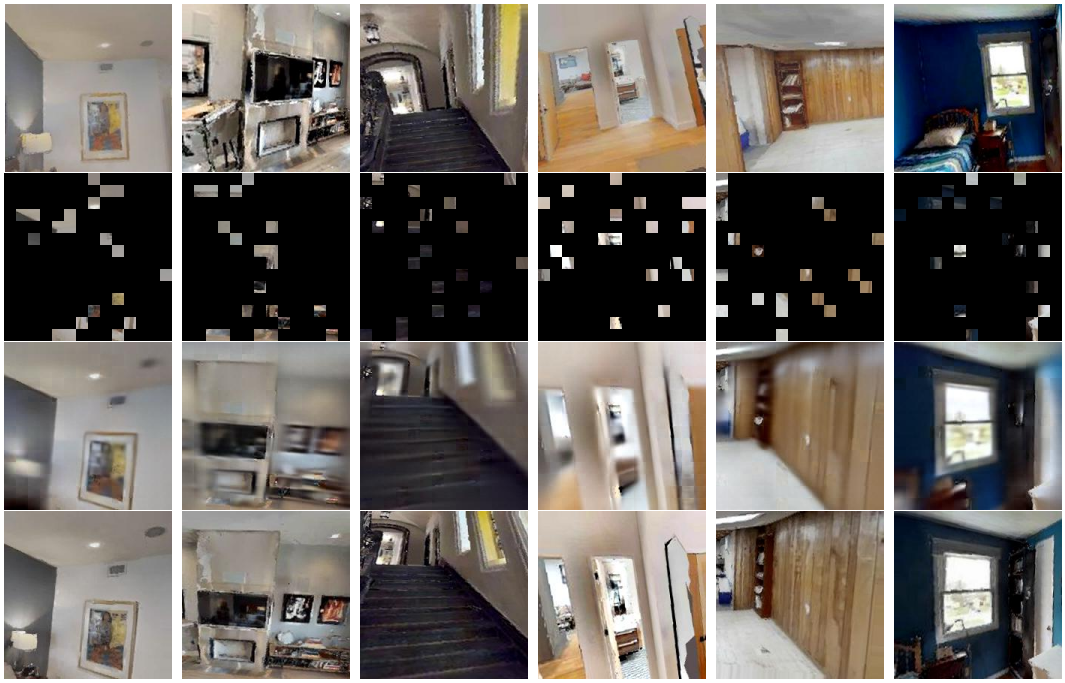

Figure 2: **Reconstruction examples from CroCo** on scenes unseen during training. From top to bottom: reference image (input), masked image (input), CroCo output, original target image.

2 Related work

Self-supervised representation learning is a paradigm developed to learn visual features from large-scale sets of unlabeled data [32], before finetuning the model on downstream tasks, *e.g*. classification or detection. To achieve that, a *pretext* task is designed and exploits inherent data attributes to automatically generate surrogate labels. The earliest principle behind most existing pretext tasks for computer vision revolves around purposefully removing *some* information from an image, *e.g*. the color, the orientation or the ordering of a sequence of patches obtained from the image that the model then learns to recover [22, 26, 40, 41]. It has been shown that despite the lack of any semantic supervision, networks trained to recover this artificially removed information learn useful representations and facilitate the training of various supervised downstream tasks [1, 24].

Recently, the paradigm of instance discrimination has received a lot of attention, achieving highly competitive results in self-supervised visual representation learning [11, 10, 13, 28, 30]. It seeks to learn outputs that are invariant to well designed classes of data augmentation. Positive image pairs, obtained from the same instance by data augmentation, are pulled closer by the model, while samples obtained from different instances are pushed apart in the embedding space.

More recently, motivated by the success of BERT [18] in NLP and by the introduction of Vision Transformers (ViT) [20], a variety of masked image prediction methods for self-supervised pretraining of vision models has been proposed. Reminiscent of denoising autoencoders [58] or context encoders [43], and aiming to reconstruct masked pixels [2, 12, 20, 23, 29, 64], discrete tokens [6, 72] or deep features [4, 61], these methods have demonstrated the ability to scale to large datasets and models and achieve state-of-the-art results on various downstream tasks. In particular, the masked autoencoder (MAE) [29] accelerates pretraining by using an asymmetric architecture that consists of a large encoder that operates only on unmasked patches followed by a lightweight decoder that reconstructs the masked patches from the latent representation and mask tokens. MultiMAE [3] leverages the efficiency of the MAE approach and extends it to multi-modal and multitask settings. Rather than masking input tokens randomly, the Masked Self-Supervised Transformer model (MST) [34] proposes to rely on the attention maps produced by a teacher network to dynamically mask low response regions of the input, and a student network is then trained to reconstruct it.

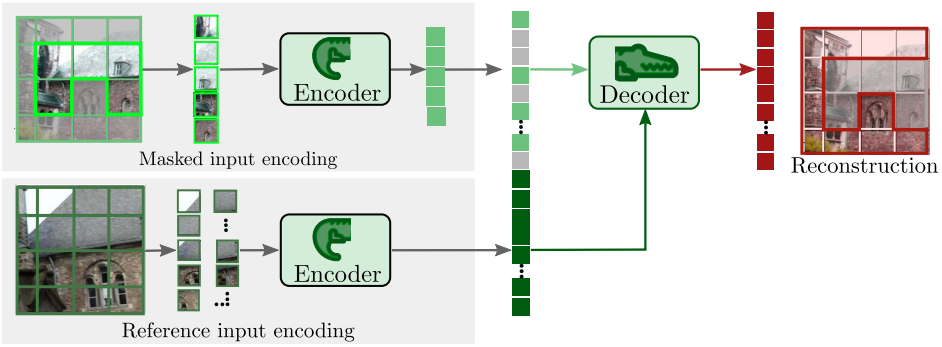

Figure 3: **Overview of our CroCo model during pretraining.** Patches from the first input image are masked; visible ones are encoded into their latent representations, then padded with masked tokens to account for hidden patches. The same encoder is used to encode the patches of the second image. The decoder receives the encoded tokens from both images and use them to reconstruct the masked parts of the target image.

Self-supervised pretraining for dense downstream tasks. Seminal self-supervised methods [14, 28] were designed to output global image representations, thus encoding limited local information, and thereby hindering transferability of the learned models to downstream tasks involving dense per-pixel predictions, such as semantic segmentation. To alleviate this issue, several self-supervised methods have been proposed to perform contrastive learning on dense local representations rather than global ones. For instance, PixPro [63] applies contrastive learning directly at the pixel level; DenseCL [60] optimizes a pairwise contrastive at the pixel level between two views of input images; while Self-EMD [36] relies on the Earth Mover's Distance (EMD) to compute a similarity score between two dense embeddings. In contrast, InsLoc [66] proposes to paste image instances at various locations and scales onto background images, then predicts instance categories and foreground bounding boxes in the composed images. In VADeR [44] and FlowE [65], dense image representations are learned for several dense semantic downstream tasks such as object detection and semantic segmentation, using multi-hierarchy features [59], while [67] combines multi-resolution parallel modules with local-window self-attention to improve the memory and computation efficiency of dense prediction. These self-supervision methods lead to improved performance on dense semantic downstream tasks such as object detection or semantic/instance segmentation. However, their design and evaluation protocols are not focused on 3D vision and lower-level geometric tasks.

3 Cross-view Completion pretraining

In this section we present CroCo, our self-supervised pretraining method based on cross-view completion and tailored to 3D vision tasks.

Overview. Figure 3 gives an overview of the CroCo architecture. Let $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ be two images of the same scene taken from different viewpoints. Both images are divided into N non-overlapping patches $\boldsymbol{p_1} = \{\boldsymbol{p_1^1}, \ldots, \boldsymbol{p_1^N}\}$, $\boldsymbol{p_2} = \{\boldsymbol{p_2^1}, \ldots, \boldsymbol{p_2^N}\}$, and a number $n = \lfloor rN \rfloor$ of tokens from the first set $\boldsymbol{p_1}$ is randomly masked, with $r \in [0, 1]$ being a hyper-parameter. We typically use a masking ratio $r = 0.9$, *i.e.*, 90% of patches from $\boldsymbol{p}_1$ are discarded. We denote the remaining set of visible patches from the first image by $\tilde{\boldsymbol{p}}_\mathbf{1} = \{\boldsymbol{p_1^i} | m_i = 0\}$, with $m_i = 0$ indicating that patch $\boldsymbol{p_1^i}$ is not masked ($m_i = 1$ otherwise).

An encoder $\mathcal{E}_\theta$ processes $\tilde{\boldsymbol{p}}_\mathbf{1}$ and $\boldsymbol{p_2}$ independently, and a decoder $\mathcal{D}_\phi$ takes the encoding $\mathcal{E}_\theta(\tilde{\boldsymbol{p}}_\mathbf{1})$, conditioned on encoding $\mathcal{E}_\theta(\boldsymbol{p}_2)$, to reconstruct $\boldsymbol{p}_1$:

$$\hat{\boldsymbol{p}}_\mathbf{1} = \mathcal{D}_\phi\left(\mathcal{E}_\theta(\tilde{\boldsymbol{p}}_\mathbf{1}); \mathcal{E}_\theta(\boldsymbol{p}_2)\right). \tag{1}$$

Details on the encoder $\mathcal{E}_\theta$. A Siamese network denoted by $\mathcal{E}_\theta$ with shared weights θ, implemented as a ViT [20], serves to encode the two input images previously split into independent patch sets $\tilde{\boldsymbol{p}}_\mathbf{1}$ and $\boldsymbol{p_2}$. In this study, we consider images of resolution 224×224, with a patch size of 16×16 pixels. Following ViT, the encoder consists in a linear projection of the flattened input RGB patches, to which sinusoidal positional embeddings [57] are added, followed by a series of transformer blocks, each composed of a multi-head self-attention block and an MLP (Multi-Layer Perceptron).

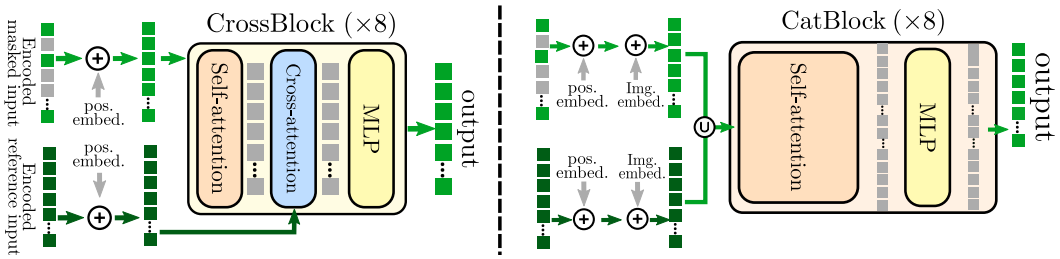

Figure 4: **Attention blocks in the decoder.** CrossBlock (left) combines information from the input sets by alternating self-attention and cross-attention, while CatBlock (right) concatenates the two sets before applying self-attention.

Details on the decoder $\mathcal{D}_\phi$. The decoder $\mathcal{D}_\phi$ with weights ϕ takes as input the set of encoded tokens from the first image $\mathcal{E}_\theta(\tilde{\boldsymbol{p}}_1)$ concatenated with a learned representation $\boldsymbol{e}_{\text{mask}}$ that is repeated n times to account for the masked patches. These tokens are iteratively processed by an attention-based block that also takes into account the encoded tokens $\mathcal{E}_\theta(\boldsymbol{p}_2)$ from the reference image $\mathbf{x}_2$. A sinusoidal positional encoding is added to all tokens before being decoded by $\mathcal{D}_\phi$. We experiment with two different architectures for the decoder block that are illustrated in Figure 4. The first one, termed *CrossBlock*, consists of (a) multi-head self-attention on the tokens representing the first image; (b) multi-head cross-attention between these tokens and the tokens in $\mathcal{E}_\theta(\boldsymbol{p}_2)$ from the reference image, and (c) an MLP. The second architecture, termed *CatBlock*, proceeds by concatenating the tokens from the two images, with an added learnable embedding for each of the two images, and then feeding these tokens to a series of standard transformer blocks, *i.e.*, each consisting of (a) multi-head self-attention and (b) an MLP layer. Only the subset of tokens corresponding to the first image is taken into account for the final prediction. These two attention blocks represent a trade-off between model size and computational cost. Indeed, *CrossBlock* has more learnable parameters, due to the cross-attention module, but a lower computational cost as it avoids the quadratic complexity of computing self-attention over the joint token set of size $2N$.

Training the CroCo model. The decoder produces one feature vector per patch token, which is processed by a fully-connected layer that outputs 3 values (R, G and B) per pixel for each patch, *e.g.*, $16 \times 16 \times 3 = 768$ values for squared patches with side 16. These outputs serve as predicted reconstruction $\hat{\boldsymbol{p}}_1$, evaluated using a reconstruction error such as the Mean Squared Error (MSE) loss between predictions and ground-truth values of the pixels, averaged over the set of all masked tokens, denoted $\boldsymbol{p}_1 \backslash \tilde{\boldsymbol{p}}_1$. Thus to perform self-supervised pretraining, the network is trained to minimize:

$$\mathcal{L}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{1}{|\boldsymbol{p}_1 \backslash \tilde{\boldsymbol{p}}_1|} \sum_{\boldsymbol{p}_1^i \in \boldsymbol{p}_1 \backslash \tilde{\boldsymbol{p}}_1} \|(\hat{\boldsymbol{p}}_1^i - \boldsymbol{p}_1^i)\|^2. \tag{2}$$

We additionally try a variant of the loss where the targets are normalized within a patch: each patch is normalized using the mean and standard deviation of all pixel values within this patch.

Pretraining details. We implement our CroCo model in PyTorch [42] and train the network for 400 epochs using the AdamW optimizer [38]. We use a cosine learning rate schedule with a base learning rate of 1.5×10^{-4} for an effective batch size of 256; with a linear warmup in the first 40 epochs. As encoder, we use a ViT-Base/16 backbone, *i.e.*, a series of 12 transformer blocks with 768 dimensions and 12 heads for self-attention with patches of size 16×16. For both *CrossBlock* and *CatBlock* decoders, we use a series of 8 blocks with 512 dimensions and 16 attention heads. To account for the different dimensions of the encoder and the decoder, we apply a fully-connected layer between them.

Pretraining data. We train our model on a set of synthetic image pairs of 3D indoor scenes derived from the HM3D [46], ScanNet [17], Replica [52] and ReplicaCAD [55] datasets as follows. In each 3D scene, we randomly sample up to 1000 pairs of camera viewpoints with a co-visibility greater than 50%, and render these pairs of images using the Habitat simulator [50]. In total, we generated a dataset of 1,821,391 pairs, that we refer to as *Habitat* in this paper.

Finetuning CroCo. Our CroCo model can be used as pretraining to solve either monocular or binocular tasks, dense or not. For monocular task, the ViT-Base encoder is used alone, while binocular tasks benefit from the pretraining of both the encoder and the decoder.

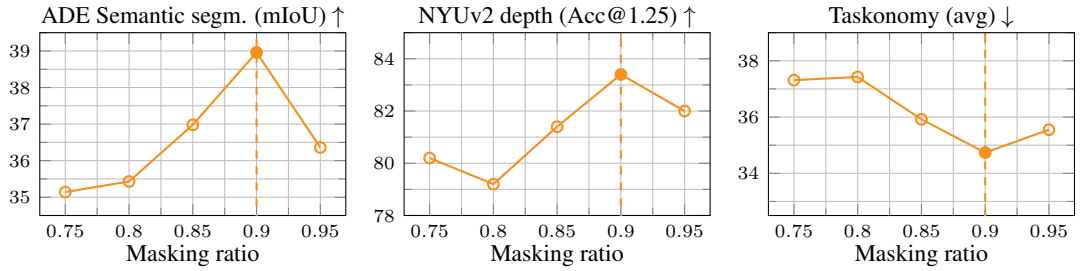

Figure 5: **Impact of the masking ratio** using the CrossBlock decoder and a loss on RGB values.

4 Experimental results

We evaluate CroCo on both downstream monocular tasks in Section 4.1, for which we provide ablations (Section 4.1.1) and a comparison to the state of the art (Section 4.1.2), and on binocular tasks in Section 4.2. Training details for each task are available in the supplementary material.

4.1 Monocular transfer tasks

High-level semantic tasks. While targeting 3D vision tasks, we still evaluate some high-level semantic tasks, namely *image classification with linear probing* on ImageNet-1K [49] and report top-1 accuracy. We follow the exact same protocol as MAE [29] for that, with global average pooling for CroCo as we did not include a [CLS] token in our model. Additionally, we report results on *semantic segmentation* on ADE20k [71] with 150 classes and 20,210 training images. We follow the protocol of [3] and report mean Intersection-over-Union (mIoU) on the validation set, using the same ConvNext [37] prediction head on top of the encoder.

3D vision tasks. We evaluate *monocular depth prediction* on NYUv2 [51] (795 training and 655 test images) by reporting δ_1, *i.e.*, the percentage of pixels that have an error ratio (max over prediction divided by ground-truth and its inverse) below 1.25. We additionally report results on the Taskonomy dataset [68] (800 training images) and report L1-loss on the 'tiny' test set (54,514 images) for the same 8 tasks as [3]: *curvature, depth, edges, 2D keypoints, 3D keypoints, normal, occlusion and reshading*. For better clarity with decimal number, we report the L1-losses multiplied by 1000. We also report the average ranking (rank.) over the 8 tasks in tables. For these tasks, we use a DPT head [47] on top of the encoder. We additionally experiment on the task of absolute camera pose regression in the supplementary material.

4.1.1 Ablations

Masking ratio. We first report in Figure 5 the performance on semantic segmentation (ADE), depth estimation (NYUv2) and Taskonomy when varying the masking ratio r using a *CrossBlock* decoder and a loss without normalization. We can observe that the best performance is obtained with a masking ratio of $r = 90\%$. This optimal ratio is higher than what was found for instance in MAE [29] for the auto-completion task. We attribute this to the help provided by the reference image in cross-view completion. We show some reconstruction examples in Figure 2. Despite the small number of visible patches, this amount, combined with the reference view, seems to provide sufficient information to reconstruct the first image. Note that the reconstructions tend to be blurry. This is due to the MSE loss, but as noted in MAE [29], beyond a certain point, sharper reconstructions do not necessarily lead to better pretraining.

Normalized targets. Regressing RGB values normalized by the mean and standard deviation within each patch has proven to be effective for MAE [29], and we measure the impact of this normalization on our CroCo model pretraining in the first two rows of Table 1. We observe that it indeed consistently improves the performance on all tasks, and we therefore use it in the remainder of the experiments.

Decoder architecture. In Table 1, we compare decoders built with the two proposed attention blocks: CrossBlock which uses cross-attention to exchange information between the two images, and CatBlock which concatenates the tokens from both images. Note that CatBlock has a smaller number of learnable parameters but a higher number of FLOPs. The performances obtained with the two

Table 1: **Impact of normalizing targets and decoder block** with a masking ratio of 90%. We also indicate the number of FLOPs and parameters for the full network (encoder and decoder).

Normalized Target	Decoder Block	ADE ↑ segm.	NYUv2 ↑ depth	Taskonomy ↓ avg.	Taskonomy ↓ rank.	FLOPs	Params
	CrossBlock	39.0	83.4	34.74	2.50	50.2G	120M
✓	CrossBlock	40.6	85.6	**33.00**	**1.75**	50.2G	120M
✓	CatBlock	**42.1**	**86.0**	33.33	**1.75**	58.5G	111M

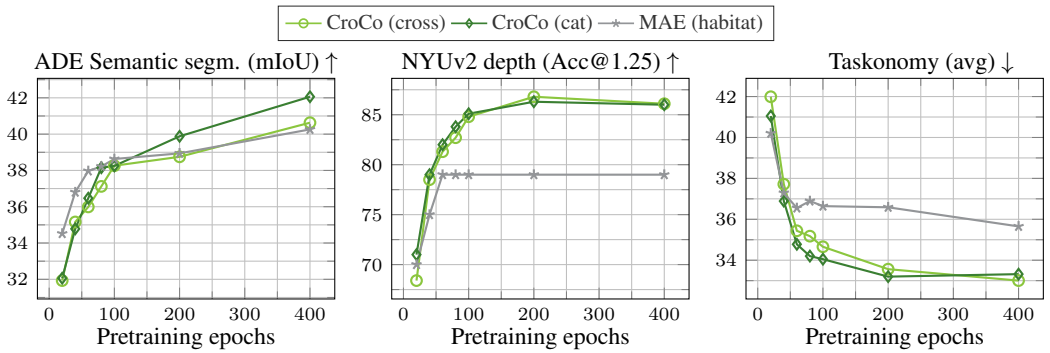

Figure 6: **Performance as a function the number of pretraining epochs** for various models.

proposed architectures are quite similar; CatBlock yielding slightly better performance for semantic segmentation or depth estimation while CrossBlock performs better on Taskonomy. We favor the use of CrossBlock in what follows. In the supplementary material, we provide an ablation study on the number of blocks used in the decoder.

Leveraging the decoder. Note that the pretrained decoder can also be used when finetuning the model on monocular tasks: it suffices to feed the input image twice to the whole CroCo network, which in practice is done by duplicating the encoder output before passing it to the decoder. While this significantly increases the computational cost of the model, it yields a consistent gain of performance from 40.6 to 41.0 on ADE, 86.1 to 88.1 on NYUv2, and an improvement on 6 out of 8 tasks on Taskonomy, see further details in the supplementary material.

Training profile. In Figure 6 we show the performance on the downstream tasks as the pretraining progresses up to 400 epochs. We observe that the performance on 3D vision tasks – depth estimation on NYUv2 and Taskonomy – reaches a plateau and thus we evaluate our model pretrained for 400 epochs in what follows. We additionally train a MAE model[1] on the Habitat dataset, using all available images (*i.e.* twice the number of pairs). When pretraining on Habitat, both models achieve similar performance on the task of semantic segmentation, while CroCo significantly outperforms MAE on depth prediction and Taskonomy. This shows the benefit of using cross-view completion pretraining, rather than pure auto-completion, for 3D vision downstream tasks.

4.1.2 Comparison to the state of the art

In Table 2, we compare the proposed CroCo pretraining strategy to the state of the art on monocular tasks. In particular, we compare to DINO [11], a state-of-the-art self-supervised method based on instance discrimination, and to MIM methods with MAE [29] and MultiMAE [3]. The latter uses extra data modalities, namely depth and semantic segmentation generated by off-the-shelf supervised models for pretraining. All three methods rely on the ImageNet-1K (IN1K) dataset [49] for pretraining. This leads to high performance on high-level semantic tasks compared to our approach. Note that DINO, which outputs a global representation, performs best on ImageNet-1K, while MIM-based approaches, which already output dense patch-level predictions during pretraining, obtain better results on dense tasks such as semantic segmentation or depth.

[1] `https://github.com/facebookresearch/mae`

Table 2: **Comparison to the state-of-the-art pretraining methods** on semantic tasks (image classification on ImageNet-1K with linear probing and semantic segmentation on ADE) and 3D vision tasks (NYUv2, Taskonomy), with a ViT-Base/16 backbone. We indicate the pretraining data in parenthesis. Best and second best results are **bold** and underlined.

Pretraining method (data)	IN1K ↑	ADE ↑	NYUv2 ↑	Taskonomy ↓									
	lin.	segm.	depth	curv.	depth	edges	kpts2d	kpts3d	normal	occl.	reshad.	avg.	rank.
DINO [11] (IN1K)	**78.2**	44.7	66.8	43.04	38.42	3.80	0.16	45.85	65.71	0.57	115.02	39.07	5
MAE [29] (IN1K)	75.1	46.1	79.6	41.59	35.83	**1.19**	0.08	44.18	59.20	**0.55**	106.08	36.09	2.13
MutliMAE [3] (IN1K)	60.2	**46.4**	83.0	41.42	35.38	2.17	**0.07**	44.03	60.35	0.56	105.25	36.17	2.75
MAE (Habitat)	32.5	40.3	79.0	42.06	33.63	1.79	0.08	44.81	59.76	0.56	102.54	35.65	2.88
CroCo (Habitat)	37.0	40.6	**85.6**	**40.91**	**31.34**	1.74	0.08	**41.69**	**54.13**	**0.55**	**93.58**	**33.00**	**1.25**

Table 3: **Optical flow results** on the MPI-Sintel dataset for various pretraining methods when finetuned on AutoFlow.

encoder init.	decoder init.	MPI-Sintel clean	MPI-Sintel final
random	random	18.81	18.97
MAE (IN1K)	random	4.68	5.16
MAE (Habitat)	random	4.63	5.24
CroCo (Habitat)	**CroCo** (Habitat)	**2.80**	**3.54**

To measure the impact of the pretraining dataset, we report the performance of a MAE model trained on Habitat, *i.e.*, with exactly the same data and ViT-Base architecture as used for CroCo. In this case, the performance on semantic tasks largely drops compared to pretraining on ImageNet (*e.g.* 75.1% to 32.5% in classification on IN1K, see Table 2). This confirms that the high performance on semantic tasks are mostly due to the use of IN1K for pretraining. When evaluating 3D vision downstream tasks, such as depth estimation on NYUv2 or Taskonomy, CroCo significantly outperforms all other methods. Interestingly, it even outperforms MultiMAE for depth estimation on NYUv2 with 85.6% Acc@1.25 *vs.* 83.0%, while other approaches are below 80%. On Taskonomy, CroCo performs best on 6 tasks out of 8 and is second best on the two other tasks, with an average rank of 1.25/5.

4.2 Applications to binocular tasks

We now empirically demonstrate that our CroCo model achieves competitive performance in two binocular tasks, namely optical flow and relative pose estimation, without any task-specific design.

Optical Flow. We treat optical flow as a straightforward regression task and do not change the pretraining architecture except for modifying the regression head to output two flow channels instead of 3 RGB color channels. We finetune our network on the public 40,000 synthetic pairs from AutoFlow [53], using a simple MSE loss on 224×224 crops, for 100 epochs, without any data augmentation besides color jittering. We evaluate on the MPI-Sintel [8] dataset (1041 image pairs in the train split), on the clean and final renderings, using the average endpoint error (AEPE) metric. To test on high resolution 1024×536 images, we regularly sample overlapping tiles of size 224×224 at the same position in both input images. We regress the flow between each pair of corresponding tiles. To recombine flow values, at each pixel location in the final prediction we use the flow value predicted by the nearest tile (*i.e.* based on the tile center).

In Table 3 we compare performance for various pretraining strategies and datasets on the MPI-Sintel training set. Without pretraining, the network is unable to learn anything useful, as shown by the large AEPE values, despite thorough but unsuccessful hyper-parameter tuning. This may be due to the high difficulty of the task coupled with the limited amount of training data. The performance significantly improves when we initialize the encoder with an MAE-pretrained model. With a CroCo initialization of both encoder *and* decoder weights, however, we again observe a large improvement with an average decrease of the error by almost 2 pixels *w.r.t.* the best MAE model on both clean and final renderings. Interestingly, we find that even with just 2 blocks in the decoder, a model pretrained with CroCo still outperforms an MAE model having 8 decoder blocks (3.59/4.35 AEPE on clean/final, resp., compared to 4.63/5.24 AEPE for MAE; see the supplementary material for a full ablation). These results clearly demonstrate the benefits from pretraining with the cross-view completion task.

Table 4: **Relative pose estimation results** with the median camera position and orientation errors on 7-scenes. Finetuning a model pretrained with CroCo achieves competitive results compared to existing methods directly regressing relative camera pose, without applying any fusion technique.

Method / Pretraining	chess	fire	heads	office	pumpkin	redkitchen	stairs	Average
RelocNet* [5]	12cm, 4.14°	26cm, 10.44°	14cm, 10.5°	18cm, 5.32°	26cm, 4.17°	23cm, 5.08°	28cm, 7.53°	21cm, 6.74°
NC-EssNet* [73]	12cm, 5.63°	26cm, 9.64°	14cm, 10.66°	20cm, 6.68°	22cm, 5.72°	22cm, 6.31°	31cm, 7.88°	21cm, 7.50°
CamNet*† [19]	4cm, **1.73°**	**3cm, 1.74°**	5cm, **1.98°**	4cm, **1.62°**	**4cm**, **1.64°**	**4cm**, **1.63°**	**4cm**, **1.51°**	**4cm**, **1.69°**
top1 AP-GeM-18	27.9cm, 12.81°	40.4cm, 16.06°	21.6cm, 16.46°	37.5cm, 12.79°	44.4cm, 12.58°	46.7cm, 13.92°	32.2cm, 14.59°	36cm, 14.2°
MAE (Habitat)	13.2cm, 9.44°	32.0cm, 15.10°	16.0cm, 16.75°	24.8cm, 11.54°	25.4cm, 10.62°	29.4cm, 13.32°	32.8cm, 14.88°	24.8cm, 13.09°
CroCo (Habitat)	**2.4cm**, 2.81°	4.0cm, 3.86°	**3.1cm**, 4.00°	**3.4cm**, 2.53°	4.9cm, 2.79°	5.5cm, 3.72°	11.7cm, 4.53°	5.0cm, 3.46°

*: fuse multiple pose predictions. †: exploit temporal information and multi-step retrieval.

Our results (2.80/3.54 on clean/final, resp.) are slightly behind recent state-of-the-art approaches like PWC-Net[54] (2.55/3.93), LiteFlowNet2 [31] (2.24/3.78) and RAFT [56] (1.43/2.71), all trained on FlyingChairs [21] and FlyingThings [39] combined. However, our CroCo model remains competitive for this task considering the simplicity of our approach. In particular, note that i) our model is trained from limited data without any sophisticated data augmentation, ii) our architecture is generic and not specially designed for optical flow prediction, unlike [54, 31, 56] which all rely on 4D cost volumes, iii) regression at test time is simple and straightforward, iv) the small 224×224 resolution used at test time hinders our performance on large displacements, a problem which we leave to future work. We refer to the supplementary material for additional ablative experiments and comparisons.

Relative pose estimation. We experiment on the relative pose regression (RPR) downstream task. Given a pair of images, the goal is to predict the relative pose of the camera with respect to a reference view. To do so, we feed to the CroCo model two images corresponding to normalized camera views, and we regress the rigid transformation $(\boldsymbol{R}, \boldsymbol{t}) \in SO(3) \times \mathbb{R}^3$ between these two views using a differentiable Procrustes layer [7] on top of the prediction head to ensure that $\boldsymbol{R}$ is a rotation matrix. We finetune the RPR model with an MSE loss between the predicted relative pose $(\boldsymbol{R}, \boldsymbol{t}) \in SO(3) \times \mathbb{R}^3$ and a ground truth $(\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}})$: $\|\boldsymbol{R} - \hat{\boldsymbol{R}}\|_F^2 + \lambda\|\boldsymbol{t} - \hat{\boldsymbol{t}}\|^2$, with λ set to 100 in practice. During training, we augment the training image pairs with random permutations, color jittering and virtual camera rotations. We evaluate on the 7-scenes dataset [27], with a single model finetuned on the union of the official training sets of all scenes, and tested on the corresponding test splits. Images are rescaled and cropped to a 224×224 resolution for simplicity. At test time, for each image of the test split we retrieve the nearest image from the training set according to their AP-GeM-18 descriptors [48], and we predict the pose relative to this retrieved image.

For each scene, we report the standard median position and median orientation error in Table 4. We compare results obtained with the model initialized with CroCo to that of a model with an encoder initialized with MAE (Habitat) and a decoder trained from scratch (last two rows). Unsurprisingly, the former leads to significantly better results than the latter. We furthermore compare our model pretrained with CroCo to other state-of-the-art RPR methods which directly regress relative camera poses [5, 19] or essential matrices [73] as well as the the retrieval baseline consisting in predicting the identity as relative pose (see Table 4 upper part). Our model outperforms all methods except CamNet [19]. Note that the latter fuses multiple pose predictions, exploit temporal information and multi-step retrieval; in contrast our model directly predicts the relative pose from a (query, map) image pair, without any further processing, and is thus significantly simpler.

5 Discussion

We have introduced the novel task of cross-view completion for pretraining computer vision networks tailored to 3D vision downstream tasks. Our experiments show that cross-view completion allows to learn representations better suited to 3D vision tasks than classical MIM with auto-completion, and straightforwardly transfer to monocular and binocular tasks. A limitation of our model pretrained with cross-view completion is that it seems less tailored to high-level semantic tasks. This is arguably more related to the choice of the dataset than to the pretraining task itself, and future work could explore the use of cross-view completion in conjunction with more object-centric datasets like ImageNet. Our approach requires pairs of images depicting the same scene, and in this work, we leverage synthetic renderings only. Future work could also extend it to real-world image pairs, which can be obtained without any supervision, for instance with Structure-from-Motion [35, 45] or geo-referencing.

References

[1] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. A Critical Analysis of Self-supervision, or what We Can Learn from a Single Image. In *ICLR*, 2020.

[2] Sara Atito, Muhammad Awais, and Josef Kittler. SiT: Self-supervised vIsion Transformer. *arXiv preprint arXiv:2104.03602*, 2021.

[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] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation Using Neural Nets. In *ECCV*, 2018.

[6] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT Pre-Training of Image Transformers. In *ICLR*, 2022.

[7] Romain Brégier. Deep regression on manifolds: a 3D rotation case study. In *3DV*, 2021.

[8] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.

[9] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

[10] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. In *NeurIPS*, 2020.

[11] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV*, 2021.

[12] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative Pretraining From Pixels. In *ICML*, 2020.

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, 2020.

[14] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E. Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020.

[15] Xiaokang Chen, Mingyu Ding, Xiaodi Wang, Ying Xin, Shentong Mo, Yunhao Wang, Shumin Han, Ping Luo, Gang Zeng, and Jingdong Wang. Context Autoencoder for Self-Supervised Representation Learning. *arXiv preprint arXiv:2202.03026*, 2022.

[16] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region Filling and Object Removal by Exemplar-Based Image Inpainting. *IEEE Trans. Image Processing*, 2004.

[17] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL HLT*, 2019.

[19] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization. In *ICCV*, 2019.

[20] 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.

[21] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks. In *ICCV*, 2015.

[22] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative Unsupervised Feature Learning with Convolutional Neural Networks. In *NeurIPS*, 2014.

[23] Alaaeldin El-Nouby, Gautier Izacard, Hugo Touvron, Ivan Laptev, Hervé Jegou, and Edouard Grave. Are Large-scale Datasets Necessary for Self-Supervised Pre-training? *arXiv preprint arXiv:2112.10740*, 2021.

[24] Linus Ericsson, Henry Gouk, and Timothy M. Hospedales. How Well Do Self-Supervised Models Transfer? In *CVPR*, 2021.

[25] 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.

[26] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. In *ICLR*, 2018.

[27] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time RGB-D camera relocalization. In *ISMAR*, 2013.

[28] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *NeurIPS*, 2020.

[29] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders are Scalable Vision Learners. *arXiv preprint arXiv:2111.06377*, 2021.

[30] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*, 2020.

[31] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow CNN - revisiting data fidelity and regularization. *IEEE Trans. PAMI*, 2021.

[32] Longlong Jing and Yingli Tian. Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey. *IEEE Trans. PAMI*, 2021.

[33] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring Plain Vision Transformer Backbones for Object Detection. *arXiv preprint arXiv:2203.16527*, 2022.

[34] Zhaowen Li, Zhiyang Chen, Fan Yang, Wei Li, Yousong Zhu, Chaoyang Zhao, Rui Deng, Liwei Wu, Rui Zhao, Ming Tang, and Jinqiao Wang. MST: Masked Self-Supervised Transformer for Visual Representation. In *NeurIPS*, 2021.

[35] Zhengqi Li and Noah Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In *CVPR*, 2018.

[36] Songtao Liu, Zeming Li, and Jian Sun. Self-EMD: Self-Supervised Object Detection without ImageNet. *arXiv preprint arXiv:2011.13677*, 2020.

[37] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022.

[38] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.

[39] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[40] Mehdi Noroozi and Paolo Favaro. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *ECCV*, 2016.

[41] Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting Self-Supervised Learning via Knowledge Transfer. In *CVPR*, 2018.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*, 2019.

[43] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context Encoders: Feature Learning by Inpainting. In *CVPR*, 2016.

[44] Pedro O. Pinheiro, Amjad Almahairi, Ryan Y. Benmalek, Florian Golemo, and Aaron Courville. Unsupervised Learning of Dense Visual Representations. In *NeurIPS*, 2020.

[45] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. PAMI*, 2018.

[46] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-Matterport 3D Dataset (HM3D): 1000 Large-scale 3D Environments for Embodied AI. In *NeurIPS datasets and benchmarks*, 2021.

[47] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

[48] Jerome Revaud, Jon Almazan, Rafael Sampaio de Rezende, and Cesar Roberto de Souza. Learning with Average Precision: Training Image Retrieval with a Listwise Loss. In *ICCV*, 2019.

[49] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

[50] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019.

[51] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[52] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[53] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. AutoFlow: Learning a better training set for optical flow. In *CVPR*, 2021.

[54] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *CVPR*, 2018.

[55] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *NeurIPS*, 2021.

[56] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020.

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

[58] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *JMLR*, 2010.

[59] Luya Wang, Feng Liang, Yangguang Li, Honggang Zhang, Wanli Ouyang, and Jing Shao. RePre: Improving Self-Supervised Vision Transformer with Reconstructive Pre-training. *arXiv preprint arXiv:2201.06857*, 2022.

[60] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense Contrastive Learning for Self-Supervised Visual Pre-Training. In *CVPR*, 2021.

[61] 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.

[62] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination. In *CVPR*, 2018.

[63] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate Yourself: Exploring Pixel-Level Consistency for Unsupervised Visual Representation Learning. In *CVPR*, 2021.

[64] 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.

[65] Yuwen Xiong, Mengye Ren, Wenyuan Zeng, and Raquel Urtasun. Self-Supervised Representation Learning from Flow Equivariancen. In *ICCV*, 2021.

[66] Ceyuan Yang, Zhirong Wu, Bolei Zhou, and Stephen Lin. Instance Localization for Self-supervised Detection Pretraining. In *CVPR*, 2021.

[67] Yuhui Yuan, Rao Fu, Lang Huang, Weihong Lin, Chao Zhang, Xilin Chen, and Jingdong Wang. HRFormer: High-Resolution Transformer for Dense Prediction. In *NeurIPS*, 2021.

[68] Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, 2018.

[69] Sameh Zarif, Ibrahima Faye, and Dayang Rohaya. Image Completion: Survey and Comparative Study. *International Journal of Pattern Recognition and Artificial Intelligence*, 2015.

[70] Y. Zhao, G. Wang, C. Luo, W. Zeng, and Z. Zha. Self-supervised visual representations learning by contrastive mask prediction. In *ICCV*, 2021.

[71] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K Dataset. In *CVPR*, 2017.

[72] 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.

[73] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To Learn or Not to Learn: Visual Localization from Essential Matrices. In *ICRA*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Yes.
   (b) Did you describe the limitations of your work?
   Yes, in the Discussion section (Section 5).
   (c) Did you discuss any potential negative societal impacts of your work?
   No, we do not envision any potential negative societal impact beyond those that are generic to machine learning and data bias.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them?
   Yes.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results?
   N/A.
   (b) Did you include complete proofs of all theoretical results?
   N/A.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)?
   Not with the submission but we are planning to release code if accepted.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)?
   All details are available in the supplementary material.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)?
No, because pretraining is costly (about a week on 8 gpus), but our ablations with for instance the different decoder blocks concur towards the same conclusion that CroCo outperforms other pretraining strategies.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)?
Yes, in the supplementary material.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators?
Yes.

(b) Did you mention the license of the assets?
Yes, in the supplementary material.

(c) Did you include any new assets either in the supplemental material or as a URL?
No.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating?
N/A.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content?
N/A.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable?
N/A.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable?
N/A.

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation?
N/A.