HyperPatch: Hypercolumns for Transformer Backbones in Medical Segmentation

Amirhossein Sabour, Sahib Khokhar, and Mehdi Moradi*

McMaster University, Hamilton, ON, Canada
{saboua4, khokhs5, moradm4}@mcmaster.ca

Abstract. Dense medical image segmentation is usually learned end-to-end, making training expensive and data hungry, especially when strong Transformer foundation models are available but hard to adapt efficiently. Classic Hypercolumns concatenate intermediate CNN features at each pixel to obtain rich per-pixel representations, yet remain underexplored in medical imaging and are essentially absent for 3D volumes. We introduce *HyperPatch*, a patch-level Hypercolumn analogue for Transformer backbones. For each patch token, we extract representations from every Transformer block and concatenate them into a high dimensional vector, yielding a multi-level descriptor with local patch content and self-attention-induced global context. We couple HyperPatch with lightweight decoders that transform these vectors into dense 2D and 3D outputs, enabling pixel- and voxel-level predictions while keeping the backbone fixed. On 2D retinal vessel and endoscopy segmentation, HyperPatch consistently reaches a Dice score of 0.8 with only two to four training images, matching or surpassing state-of-the-art performance. We also provide feasibility results on 3D liver segmentation in abdominal CT. HyperPatch offers a simple and efficient strategy to repurpose pretrained Transformers for dense medical segmentation using a small trainable head and very limited training data. Code will be released upon acceptance.

Keywords: HyperPatch · Hypercolumn · Foundation Models · Patch-level · Dense Prediction · Medical Image Segmentation.

1 Introduction

One of the primary challenges in medical AI is the high cost of supervision. High-quality annotation is slow and often prohibitively expensive because medical images are complex and reliable labels require expert domain knowledge. This cost limits how quickly new tasks can be defined, especially for fine-grained segmentation, where strong performance often depends on large-scale data. Modern models, particularly transformers, also require substantial labeled data to generalize well [5]. This has driven work on data-efficient training [19], domain

* Corresponding author

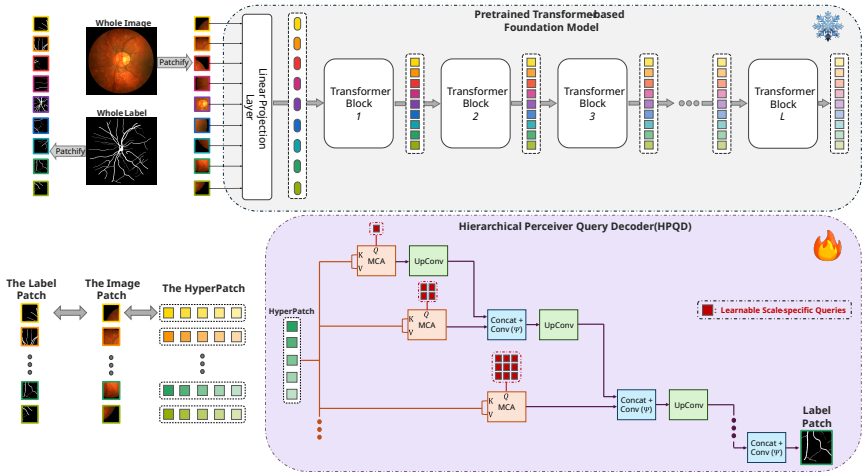

Fig. 1. HyperPatch overview. A frozen Transformer provides layer-wise patch tokens; HPQD queries each patch descriptor and decodes dense 2D or 3D predictions.

generalization [22] that transfers from data-rich domains via fine-tuning, multi-task learning [20], self-supervision [7, 3], and efficient adaptation [8]. In low-data regimes, the core issue is generalization: when an image is treated as one training instance, small datasets let models latch onto incidental cues that fail on unseen cases.

A common route to label-efficient learning is to increase the number of supervised instances extracted from each labeled image. *Hypercolumns* [6] do this by treating each spatial location as an instance described by activations from multiple CNN layers. In practice, they upsample intermediate feature maps to a common target grid, concatenate them across channels, and use the resulting per-location vector as the hypercolumn. With a pretrained backbone, this descriptor can encode local appearance and broader CNN context. Later data-factory pipelines use the same dense supervision idea with GAN-based synthesis or editing [13, 21]. Yet hypercolumns are high dimensional, require large per-location heads that can overfit, and are tied to CNN feature pyramids. Modern transformer foundation models instead use discrete patch tokens, where a direct pixel-level analogue is less natural.

We generalize hypercolumns to transformer backbones at the patch level. Each patch token becomes a training instance, and its intermediate transformer features form a *HyperPatch* descriptor for data-efficient learning. Figure 1 summarizes this pipeline.

Our contributions can be summarized as follows:

(1) **HyperPatch (patch-level hypercolumns for transformers):** We introduce *HyperPatch*, which treats transformer patches (not pixels) as training instances and builds a per-patch descriptor from intermediate backbone features.

(2) **H-PQD (lightweight Perceiver-style patch decoding):** We propose a *Hierarchical Perceiver Query Decoder (HPQD)* that aggregates variable-length intermediate patch tokens using learned, scale-specific queries, and produces $P \times P$ predictions via a coarse-to-fine hierarchy.
(3) **Backbone-agnostic low-label dense prediction:** Our framework is foundation model-agnostic: it reuses intermediate representations from arbitrary frozen transformer backbones to enable data-efficient dense segmentation in low-label regimes.
(4) **2D and 3D results:** Experiments on two 2D medical segmentation benchmarks and one 3D volumetric dataset show strong low-data performance while training only a compact decoder.

2 Methodology

2.1 Problem Definition

Let $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{N}$ denote a dataset of images and binary segmentation masks, where $\mathbf{X}_i \in \mathbb{R}^{H\times W\times C}$ (e.g., $C = 3$ for RGB images and $C = 1$ for grayscale images) and $\mathbf{Y}_i \in \{0,1\}^{H\times W}$. The value $\mathbf{Y}_i(u, v) \in \{0, 1\}$ indicates whether pixel (u, v) belongs to the background or foreground class.

Each pair $(\mathbf{X}_i, \mathbf{Y}_i)$ is mapped by a deterministic preprocessing operation $\mathcal{T}(\cdot)$ to $(\tilde{\mathbf{X}}_i, \tilde{\mathbf{Y}}_i)$, where $\tilde{\mathbf{X}}_i \in \mathbb{R}^{S\times S\times C}$ and $\tilde{\mathbf{Y}}_i \in \{0,1\}^{S\times S}$. Following the patchification used in [5], let $\mathbf{X}_i^p$ denote the patchified sequence representation of $\tilde{\mathbf{X}}_i$, obtained by splitting it into G^2 flattened, non-overlapping $P \times P$ patches,

$$\mathbf{X}_i^p \in \mathbb{R}^{G^2\times(P^2\cdot C)}, \qquad G = S/P. \tag{1}$$

Here, $G^2 = S^2/P^2$ is the number of patches (i.e., the sequence length). For the i-th image, patches are indexed by $j \in \{1, \ldots, G^2\}$, where $\mathbf{X}_{i,j} \in \mathbb{R}^{P\times P\times C}$ denotes the j-th image patch and $\mathbf{Y}_{i,j} \in \{0,1\}^{P\times P}$ the corresponding label patch.
HyperPatch - Patch Representation Extraction: Let $\mathcal{FM}$ denote a frozen pretrained transformer-based foundation model with L intermediate layers. Given the patchified input $\mathbf{X}_i^p$, $\mathcal{FM}$ outputs a sequence of patch tokens at each layer $\ell \in \{1, \ldots, L\}$. In particular, at layer ℓ, the output token sequence is

$$\mathbf{Z}_i^{(\ell)} \in \mathbb{R}^{G^2\times d_\ell}. \tag{2}$$

where d_ℓ is the token dimension, and the j-th token corresponds to the j-th image patch. The token from layer ℓ associated with patch index j of image i is written as $\mathbf{z}_{i,j}^{(\ell)} \in \mathbb{R}^{d_\ell}$, where $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, G^2\}$, and $\ell \in \{1, \ldots, L\}$. The HyperPatch representation for the j-th patch of the i-th image is defined as the ordered collection of these tokens across depth,

$$\mathcal{H}_{i,j} = \big(\mathbf{z}_{i,j}^{(1)}, \mathbf{z}_{i,j}^{(2)}, \ldots, \mathbf{z}_{i,j}^{(L)}\big). \tag{3}$$

This layer-wise patch token collection is hypothesized to capture complementary cues (patch-local, neighborhood, and global) for the j-th patch of the i-th image.

The objective is to learn a general-purpose decoder g_θ that operates at the patch level and maps each HyperPatch representation $\mathcal{H}_{i,j}$ to per-pixel logits over the associated patch:

$$\hat{\mathbf{S}}_{i,j} = g_\theta(\mathcal{H}_{i,j}), \qquad \hat{\mathbf{S}}_{i,j} \in \mathbb{R}^{2\times P\times P}. \tag{4}$$

These logits induce a predicted binary label patch $\hat{\mathbf{Y}}_{i,j} \in \{0,1\}^{P\times P}$ (e.g., via an arg max over the two channels). This patch-local formulation enables treating each pair $(\mathcal{H}_{i,j}, \mathbf{Y}_{i,j})$ as an independent training instance for the decoder g_θ. During inference, an image-level prediction on the $S \times S$ image grid is obtained by tiling the set of patch-level predictions $\{\hat{\mathbf{Y}}_{i,j}\}_{j=1}^{G^2}$.

2.2 Hierarchical Perceiver Query Decoder (HPQD)

As defined in Sec. 2.1, the decoder g_θ maps each HyperPatch representation $\mathcal{H}_{i,j}$ to per-pixel logits $\hat{\mathbf{S}}_{i,j}$ over the associated $P \times P$ image patch (Eq. (4)). We recall that $\mathcal{H}_{i,j}$ is the layer-wise token collection extracted for patch index j of image i (Eq. (3)), and thus serves as a patch-specific representation used to predict the labels of the pixels within that patch. A primary challenge is that the resulting token sequence has length L, where L (and, in general, the token dimensionalities across layers) depends on the chosen foundation model, which complicates designing a decoder with consistent intermediate shapes.

Inspired by works such as [9, 12], we propose a hierarchical architecture that decouples decoding from backbone depth by treating the layer-wise tokens within each HyperPatch as a latent set of semantic descriptors and learning to query them at multiple scales.

Specifically, let $\mathcal{R} = \{r_0, r_1, \ldots, r_M\}$ denote a schedule for increasing spatial resolutions of the decoder g_θ, where $r_0 = 1$ is the bottleneck seed size (e.g., 1×1) and $r_M = P$ is the target patch size ($P \times P$). For each resolution $r_k \in \mathcal{R}$, we introduce a set of r_k^2 learnable scale-specific queries $\mathbf{Q}_k \in \mathbb{R}^{r_k^2\times d}$, where d is the hidden dimension, that learn to extract the most relevant information from the HyperPatch, $\mathcal{H}_{i,j} = (\mathbf{z}_{i,j}^{(\ell)})_{\ell=1}^{L}$, for scale k.

Perceiver-Based Spatial Projection in HPQD: At any scale k, the decoder extracts scale-specific information from the HyperPatch by querying the HyperPatch token sequence (i.e. $\mathcal{H}_{i,j} = (\mathbf{z}_{i,j}^{(\ell)})_{\ell=1}^{L}$). We model this interaction with a Multi Head Cross Attention (MCA) block, where the r_k^2 learnable scale-specific queries $\mathbf{Q}_k = (\mathbf{q}_{k,1}, \mathbf{q}_{k,2}, \ldots, \mathbf{q}_{k,r_k^2})^\top \in \mathbb{R}^{r_k^2\times d}$ attend to the HyperPatch tokens $\mathcal{H}_{i,j} = (\mathbf{z}_{i,j}^{(\ell)})_{\ell=1}^{L}$, which serve as the Key-Value set:

$$\mathbf{V}_k = \mathrm{Reshape}_{r_k\times r_k}(\mathrm{MCA}(\mathbf{Q}_k, \mathcal{H}_{i,j})), \qquad \mathbf{V}_k \in \mathbb{R}^{C_k\times r_k\times r_k}. \tag{5}$$

This produces a feature map with a fixed spatial layout at scale r_k that is invariant to the sequence length L (the number of transformer blocks in the backbone), enabling consistent feature shapes across foundation models.

Hierarchical Reconstruction in HPQD: Given the set of scale-specific query feature maps $\{\mathbf{V}_k\}_{k=0}^{M}$ obtained via Eq. (5), HPQD reconstructs a progressively

refined representation through a coarse-to-fine hierarchy. Let $\mathbf{Z}_k$ denote the decoder's internal feature map at resolution r_k (with $\mathbf{Z}_0$ at the seed scale $r_0 = 1$). We initialize the reconstruction at the seed scale by setting

$$\mathbf{Z}_0 = \mathbf{V}_0, \qquad \mathbf{Z}_0 \in \mathbb{R}^{C_0 \times r_0 \times r_0}, \tag{6}$$

and for each subsequent resolution r_k (with $k > 0$), we upsample the previous representation and fuse it with the newly queried features at the current scale:

$$\mathbf{Z}_k = \Psi(\text{Concat}[\text{UpConv}(\mathbf{Z}_{k-1}),\ \mathbf{V}_k]), \qquad \mathbf{Z}_k \in \mathbb{R}^{C_k \times r_k \times r_k}. \tag{7}$$

Here, UpConv$(\cdot)$ denotes a transposed convolution that maps $\mathbf{Z}_{k-1}$ to resolution r_k, and Ψ is a lightweight convolutional fusion block. This design injects global context at the seed scale ($r_0 = 1$) and progressively refines patch-local structure by incorporating scale-specific queries at finer resolutions $(r_1, \ldots, r_M)$. The final feature map $\mathbf{Z}_M$ is projected to pixel-level logits $\hat{\mathbf{S}}_{i,j}$.

Applicability to 3D Volumes: The same HyperPatch construction and patch-level decoding paradigm extends directly to volumetric segmentation by replacing 2D image patches with 3D voxel patches. A 3D transformer backbone yields one token per 3D patch at each intermediate layer, so $\mathcal{H}_{i,j}$ is defined identically as the layer-wise token collection associated with a fixed patch index. Accordingly, HPQD is applied in the same manner, with 2D reshaping and upsampling operators replaced by their 3D counterparts (e.g., $\hat{\mathbf{S}}_{i,j} \in \mathbb{R}^{2 \times P \times P \times P}$).

3 Experiments and Results

We report extensive results on two benchmark 2D datasets and initial feasibility results on a 3D dataset.

3.1 Results on CHASE-DB1

Dataset: CHASE-DB1 [15] is a small, publicly available retinal vessel dataset from Kingston University London and St. George's, University of London. It is derived from the Child Heart and Health Study in England (CHASE) and contains 28 fundus images from both eyes of 14 children in a multi-ethnic cohort. Each image has two ground truth annotations from independent human observers. The dataset is split at the patient level, with a fixed hold-out test set of six images from three patients and four images from two patients for validation.

Implementation details: For HyperPatch, all backbones are frozen and only HPQD is trained for 200 epochs with BCE+Dice loss, AdamW (lr = $5{\times}10^{-4}$, weight decay = 10^{-4}), cosine annealing, and patch-level batch size $1,024$. Augmentations include random flips, gamma, RGB shift, brightness/contrast, and random resized crops (scale $0.5 - 1.0$) each epoch. Checkpoints use validation Dice; all baselines share the same splits and budgets.

Experiments: Table 1 reports Dice for $N \in \{2, 4, 8, 16, 18\}$ training images. Baselines include DeepLabV3 [4], U-Net [17], and a state-of-the-art ViT-based

DPT decoder [16] with DINOv2 ViT-B [14] or SAM ViT-H [11]; HyperPatch is evaluated with the same two transformer backbones. Across all N, HyperPatch outperforms conventional CNN models, whose fine-tuned versions reach only 0.63–0.65 Dice at very low N. Freezing U-Net or DeepLabV3 backbones lowers trainable parameters but does not improve these low-data results.

Compared with DPT, HyperPatch is stronger for very small N and converges as N grows; with SAM ViT-H, it reaches 0.80 Dice using only two images.

Table 1. CHASE-DB1 retinal vessel segmentation (test Dice on 6 held-out images). All frozen-backbone methods train only the decoder for 200 epochs with identical augmentation.

Method	Backbone	Params	Training images (N)				
			2	**4**	**8**	**16**	**18**
Fully fine-tuned (all parameters trainable)							
U-Net	ResNet-34	24 M	0.65	0.67	0.73	0.76	0.76
DeepLabV3	ResNet-50	42 M	0.63	0.63	0.66	0.63	0.61
Frozen backbone							
U-Net	ResNet-34	3.2 M	0.58	0.58	0.61	0.71	0.71
DeepLabV3	ResNet-50	18.5 M	0.23	0.22	0.23	0.24	0.24
DPT Decoder	DINOv2 ViT-B	20.0 M	0.76	0.77	0.78	0.79	0.79
DPT Decoder	SAM ViT-H	20.8 M	0.78	0.80	**0.82**	**0.83**	**0.83**
HyperPatch	DINOv2 ViT-B	14.1 M	0.78	0.79	0.79	0.80	0.80
HyperPatch	SAM ViT-H	17.2 M	**0.80**	**0.81**	**0.82**	0.81	0.82

3.2 Results on Kvasir-SEG

Dataset: Kvasir-SEG [10] is an open-access gastrointestinal polyp dataset with 1,000 RGB colonoscopy images, each paired with a manually annotated polyp mask created and verified by an expert gastroenterologist.

Implementation details: Implementation largely mirrors CHASE-DB1, with cross-entropy loss rather than BCE+Dice, a fixed learning rate, crop scale 0.1–1.0, and patch-level batch size 512.

Experiments: We reserve 400 images for testing and 88 for validation, then vary training size as $N \in \{2, 4, 8, 16, 32, 64, 128, 256, 512\}$. Table 2 reports mean Dice. Fully fine-tuned CNNs become competitive near 512 images, but DPT and HyperPatch lead at smaller N. Below 256 images, DPT and HyperPatch with DINOv2 ViT-B are statistically indistinguishable, while HyperPatch uses 14.1M trainable parameters versus 20M for DPT.

Figure 2 presents example outcomes for different methods and numbers of training images on both the CHASE-DB1 and Kvasir-SEG datasets.

Table 2. Kvasir-SEG polyp segmentation (test Dice on 400 held-out images; 512 train / 88 val / 400 test). All frozen-backbone methods train only the decoder for 200 epochs with identical augmentation.

Method	Backbone	Params	Training images (N)							
			4	**8**	**16**	**32**	**64**	**128**	**256**	**512**
Fully fine-tuned (all parameters trainable)										
U-Net	ResNet-34	24 M	0.49	0.55	0.72	0.82	0.86	0.87	0.89	**0.90**
DeepLabV3	ResNet-50	42 M	0.46	0.57	0.73	0.83	0.86	**0.88**	**0.90**	**0.90**
Frozen backbone										
U-Net	ResNet-34	7 M	0.23	0.42	0.64	0.72	0.76	0.78	0.81	0.82
DeepLabV3	ResNet-50	2 M	0.52	0.55	0.63	0.69	0.74	0.75	0.79	0.80
DPT Decoder	DINOv2 ViT-B	20.0 M	**0.76**	**0.80**	**0.84**	**0.84**	**0.87**	**0.88**	**0.90**	**0.90**
HyperPatch	DINOv2 ViT-B	14.1 M	**0.76**	0.79	0.83	**0.84**	0.86	0.86	0.88	0.87

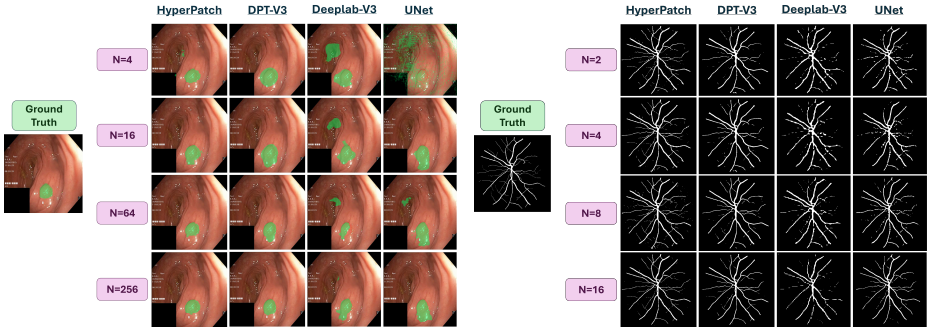

Fig. 2. Qualitative segmentation results on Kvasir-SEG (left) and CHASE-DB1 (right). Each row corresponds to increasing training set size N.

3.3 Results on MSD-Liver

Dataset: MSD Liver [18, 1, 2] is a publicly available volumetric dataset for liver and liver tumour segmentation released as part of the Medical Segmentation Decathlon, with 131 contrast-enhanced abdominal CT volumes and voxel-level liver annotations.

Implementation details: HyperPatch3D uses a pretrained 3DINO ViT backbone to extract 1024 dimensional hypercolumn features from 96^3-voxel patches at 1 mm isotropic spacing; only HPQD is trained. Training uses 100 epochs with weighted cross-entropy (inverse square-root class weighting), AdamW (lr $= 5\times10^{-4}$, weight decay $= 10^{-4}$), cosine annealing ($\eta_{\min} = 10^{-6}$), and patch-level batch size 256 (reduced to 128/64/32 for the smallest subsets to keep comparable weight updates). Augmentations include flips, 90° rotations, gamma adjustment, histogram shift, Gaussian noise, and a stochastic choice of smoothing, sharpening, or Gibbs noise via 4 random crops per volume. Checkpointing uses

validation loss; inference uses 50%-overlap sliding windows, with mean Dice over 31 held-out volumes. All experiments share the same validation/test splits and epoch count.

Experiments: We use 31 test and 10 validation volumes, vary training size as $N \in \{2, 4, 8, 16, 32, 64\}$, and compare HyperPatch3D with 3DINO-UNETR. Fig. 3 shows the expected drop below a small-N threshold, but stronger low-data robustness than the baseline. This is consistent with treating each volume as many patch-level training instances, reducing overfitting and supporting liver-specific learning when annotations are scarce. At higher N, HyperPatch3D is on par with 3DINO-UNETR, motivating further study for data-scarce 3D medical segmentation, where annotation cost is a central bottleneck.

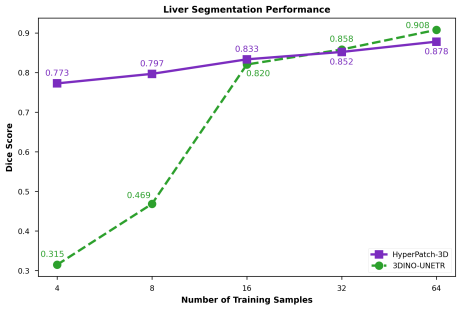

Fig. 3. Results showing the performance of HyperPatch3D in comparison to the 3DINO-UNETR baseline, for increasing training image counts.

Conclusion HyperPatch extends Hypercolumns to Transformer foundation models through patch-level descriptors for dense medical segmentation. It achieves competitive 2D and 3D results with limited training data; future work will benchmark broader 3D tasks.

References

1. Antonelli, M., Reinke, A., Bakas, S., Farahani, K., Kopp-Schneider, A., Landman, B.A., Litjens, G., Menze, B., Ronneberger, O., Summers, R.M., et al.: The medical segmentation decathlon. Nature communications **13**(1), 4128 (2022)
2. Bilic, P., Christ, P., Li, H.B., Vorontsov, E., Ben-Cohen, A., Kaissis, G., Szeskin, A., Jacobs, C., Mamani, G.E.H., Chartrand, G., et al.: The liver tumor segmentation benchmark (lits). Medical image analysis **84**, 102680 (2023)
3. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)
4. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation (2017), https://arxiv.org/abs/1706.05587

5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale (2021), https://arxiv.org/abs/2010.11929
6. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization (2015), https://arxiv.org/abs/1411.5752
7. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022)
8. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al.: Lora: Low-rank adaptation of large language models. Iclr **1**(2), 3 (2022)
9. Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., Carreira, J.: Perceiver: General perception with iterative attention. In: International conference on machine learning. pp. 4651–4664. PMLR (2021)
10. Jha, D., Smedsrud, P.H., Riegler, M.A., Halvorsen, P., de Lange, T., Johansen, D., Johansen, H.D.: Kvasir-seg: A segmented polyp dataset (2019), https://arxiv.org/abs/1911.07069
11. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything (2023), https://arxiv.org/abs/2304.02643
12. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In: International conference on machine learning. pp. 19730–19742. PMLR (2023)
13. Ling, H., Kreis, K., Li, D., Kim, S.W., Torralba, A., Fidler, S.: Editgan: High-precision semantic image editing (2021), https://arxiv.org/abs/2111.03186
14. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision (2024), https://arxiv.org/abs/2304.07193
15. Owen, C.G., Rudnicka, A.R., Mullen, R., Barman, S.A., Monekosso, D., Whincup, P.H., Ng, J., Paterson, C.: Measuring retinal vessel tortuosity in 10-year-old children: Validation of the computer-assisted image analysis of the retina (caiar) program. INVESTIGATIVE OPHTHALMOLOGY & VISUAL SCIENCE **50**, 2004–2010 (2009). https://doi.org/10.1167/iovs.08-3018, https://durham-repository.worktribe.com/output/1226234
16. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction (2021), https://arxiv.org/abs/2103.13413
17. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation (2015), https://arxiv.org/abs/1505.04597
18. Simpson, A.L., Antonelli, M., Bakas, S., Bilello, M., Farahani, K., Van Ginneken, B., Kopp-Schneider, A., Landman, B.A., Litjens, G., Menze, B., et al.: A large annotated medical image dataset for the development and evaluation of segmentation algorithms. arXiv preprint arXiv:1902.09063 (2019)
19. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International conference on machine learning. pp. 10347–10357. PMLR (2021)
20. Vandenhende, S., Georgoulis, S., Proesmans, M., Dai, D., Gool, L.V.: Revisiting multi-task learning in the deep learning era. IEEE transactions on pattern analysis and machine intelligence (2020)

21. Zhang, Y., Ling, H., Gao, J., Yin, K., Lafleche, J.F., Barriuso, A., Torralba, A., Fidler, S.: Datasetgan: Efficient labeled data factory with minimal human effort (2021), https://arxiv.org/abs/2104.06490
22. Zhou, K., Liu, Z., Qiao, Y., Xiang, T., Loy, C.C.: Domain generalization: A survey. IEEE transactions on pattern analysis and machine intelligence **45**(4), 4396–4415 (2022)