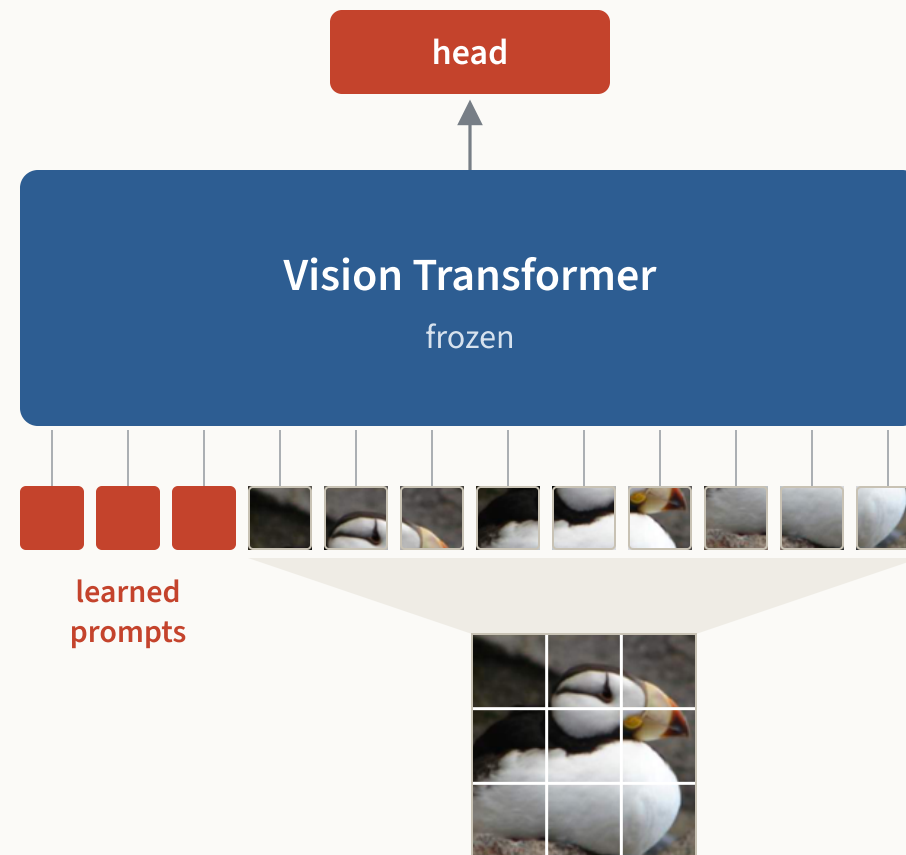
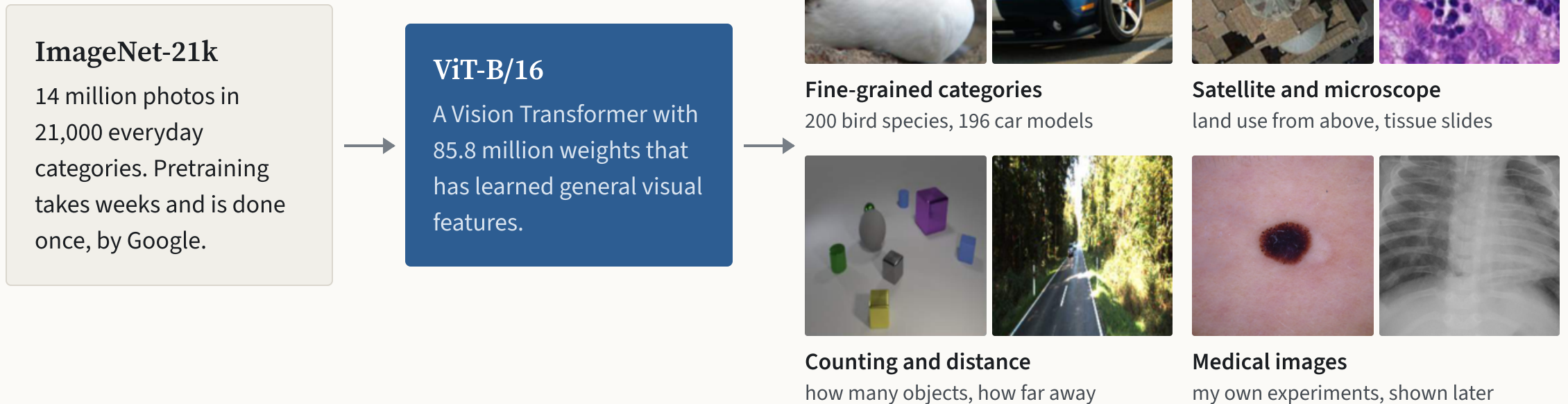


Visual Prompt Tuning

Teach a frozen Vision Transformer a new task by learning only a few extra input vectors.



Pretrain one big model once, then adapt it to each task



Each new task is small: most tasks in the paper have only **1,000 training images**, far too few to train from scratch. So we start from the pretrained model and **adapt** it.

The paper asks how to adapt it best: accurately, and cheaply enough to store for every task. Next, the standard answer and what it costs, then the cheaper methods tried before VPT.

Full fine-tuning stores a new copy of the model per task

Full fine-tuning retrains every weight of the pretrained model on the new task, and it is usually the most accurate option. The cost is that each task keeps its own retrained copy, so storage grows with every task and with the size of the model.

Shown for **1 task** on **ViT-B**, **86 million weights**. The next pages show 24 tasks, then the largest model.

Full fine-tuning: 1 retrained ViT-B copy



343 MB

1 copy at 343 MB each

VPT: one shared ViT-B plus 1 small file



each red speck is one task file, to scale

345 MB

343 MB once, plus 2.2 MB per task

Full fine-tuning stores a new copy of the model per task

Full fine-tuning retrains every weight of the pretrained model on the new task, and it is usually the most accurate option. The cost is that each task keeps its own retrained copy, so storage grows with every task and with the size of the model.

Shown for 24 tasks on ViT-B, 86 million weights.

Full fine-tuning: 24 retrained ViT-B copies



8.2 GB

24 copies at 343 MB each

VPT: one shared ViT-B plus 24 small files



395 MB

343 MB once, plus 2.2 MB per task

Full fine-tuning stores a new copy of the model per task

Full fine-tuning retrains every weight of the pretrained model on the new task, and it is usually the most accurate option. The cost is that each task keeps its own retrained copy, so storage grows with every task and with the size of the model.

Shown for 24 tasks on ViT-H, 632 million weights.

Full fine-tuning: 24 retrained ViT-H copies



61 GB

24 copies at 2.5 GB each

VPT: one shared ViT-H plus 24 small files



2.7 GB

2.5 GB once, plus 8.7 MB per task

Cheaper methods existed, but none matched full fine-tuning

		Mean accuracy (%)	Weights trained
Full fine-tuning	retrain every weight	73.9	100%
Only the top of the model is trained			
Partial-1	train the head and the last of 12 layers	66.2	8.35%
Linear probe	train only a new output layer	63.1	0.09%
MLP-3 head	train a bigger, three-layer head	62.8	1.45%
Small parts in or beside the backbone are trained			
Bias	train only the bias terms of every layer	71.0	0.21%
Adapter	insert small trainable blocks in every layer	66.7	0.36%
Sidetune	train a small network beside the model	57.0	10.09%

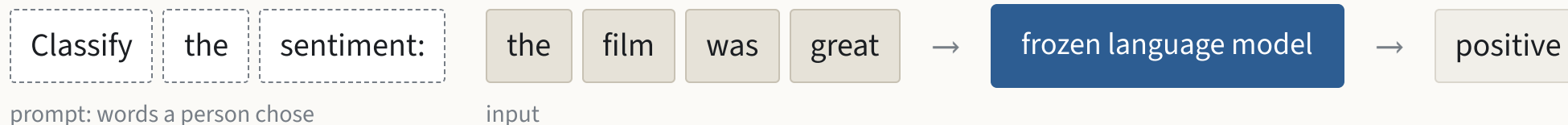
Bias came closest and even beat full fine-tuning on 8 of the 24 tasks, but on average no cheaper method reached it. VPT takes a third route: it leaves the model alone and changes the input.

Mean test accuracy over the paper's 24 tasks, ViT-B/16 (Jia et al. 2022, Table 1, mean of the four group averages). The right column is the share of weights trained for each task.

Where the word "prompt" comes from

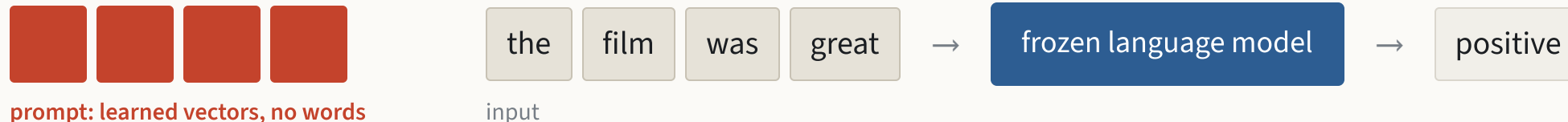
2020

GPT-3



2021

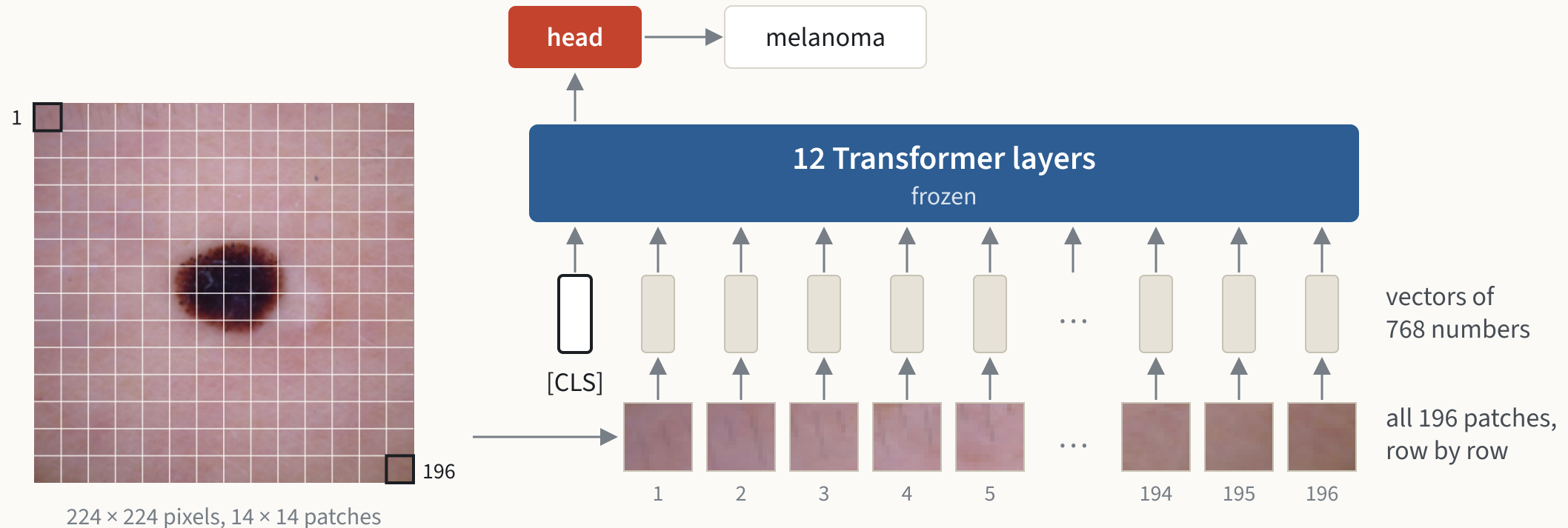
Prompt tuning



In 2020, a prompt was text. GPT-3, a frozen model with 175 billion weights, was only trained to continue text, yet it does a new task when you type a description of the task in front of the input. Not a single weight changes. But words are discrete, so the only way to improve a prompt is trial and error.

In 2021, prompt tuning (Lester et al.) replaced the words with **learned vectors** that correspond to no word. They are trained by gradient descent on the task's examples while the model stays frozen, and they are all you store per task. Prefix-tuning (Li & Liang) does the same at every layer.

How a Vision Transformer reads an image



A fixed grid cuts the image into 196 patches of 16×16 pixels, and each patch, read row by row, becomes a vector of 768 numbers. One extra vector, [CLS], goes in front to collect a summary of the whole image: the 12 layers mix all 197 vectors, and the head reads only [CLS] to name the class.

The layers only see a sequence of vectors, so we can add a few more.

A visual prompt is the same idea applied to image patches

Language

Lester et al. 2021



learned prompt: random at first, trained for the task, same for every input

the film was great



"the film was great"



frozen language model



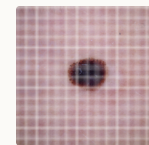
positive

Vision

Jia et al. 2022



learned prompt: random at first, trained for the task, same for every image



all 196 patches of the image



frozen Vision Transformer

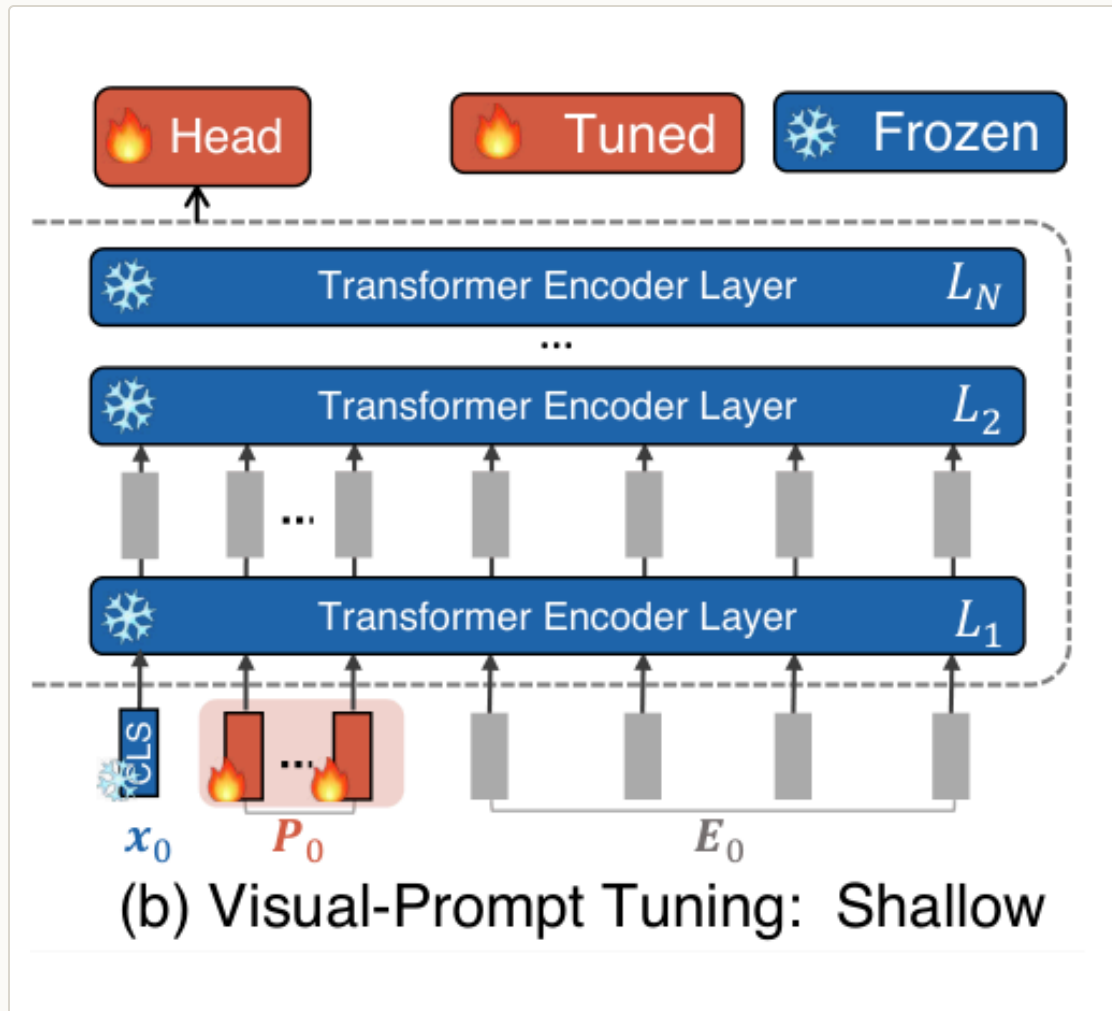


melanoma

The grey vectors come from the input, as before. The red ones do not: they start as random numbers, are trained on the task's examples while the model stays frozen, and are then put in front of every input of that task. A visual prompt is therefore not text and not a pattern painted on the pixels, only learned numbers in the model's input space.

In language, prompt tuning matched full fine-tuning only once the model had about 10 billion weights. VPT beats full fine-tuning on 20 of 24 tasks with an 86-million-weight ViT-B.

VPT trains a few extra input vectors and the head



VPT-shallow

VPT-deep

The p prompt vectors P are added once, before the first layer. After that they pass through the 12 frozen layers like any other token.

$$[x_1, Z_1, E_1] = L_1([x_0, P, E_0]) \quad \text{prompts enter once}$$

$$[x_i, Z_i, E_i] = L_i([x_{i-1}, Z_{i-1}, E_{i-1}]) \quad i = 2 \dots 12$$

$$y = \text{Head}(x_{12})$$

Per task, only the red parts are trained and saved: 50 prompt vectors and the head, **0.14%** of the model.

VPT trains a few extra input vectors and the head

VPT-shallow

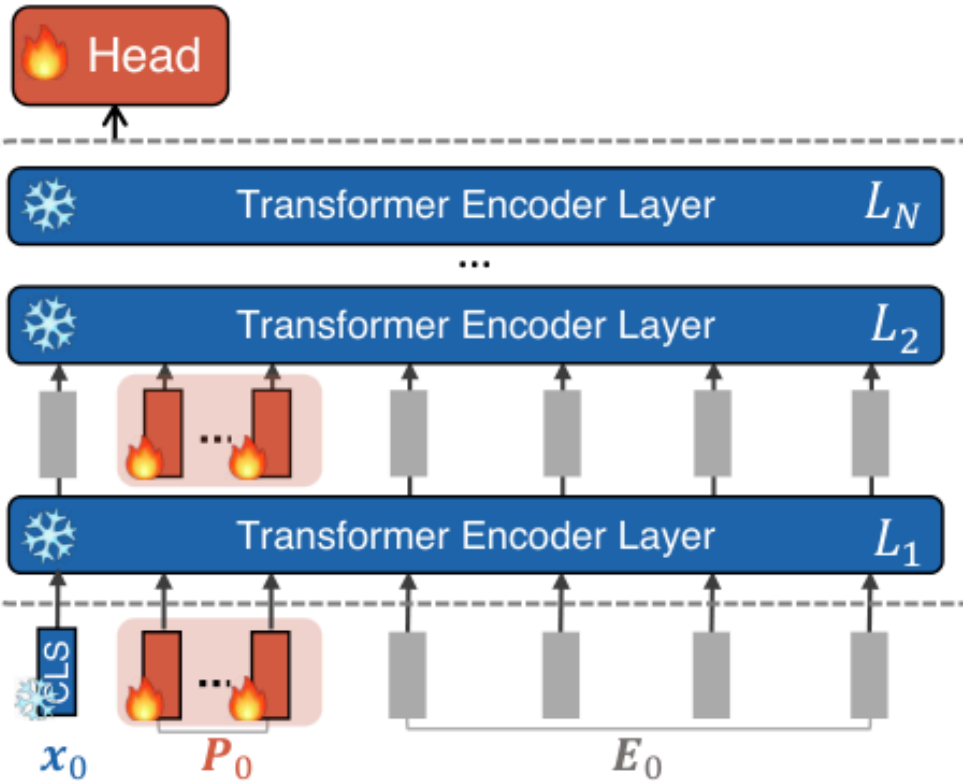
VPT-deep

Every layer gets its own fresh set of p prompt vectors. What the previous layer produced at those positions is thrown away: the blank in the equation.

$$[x_i, _, E_i] = L_i([x_{i-1}, P_{i-1}, E_{i-1}]) \quad i = 1 \dots 12$$

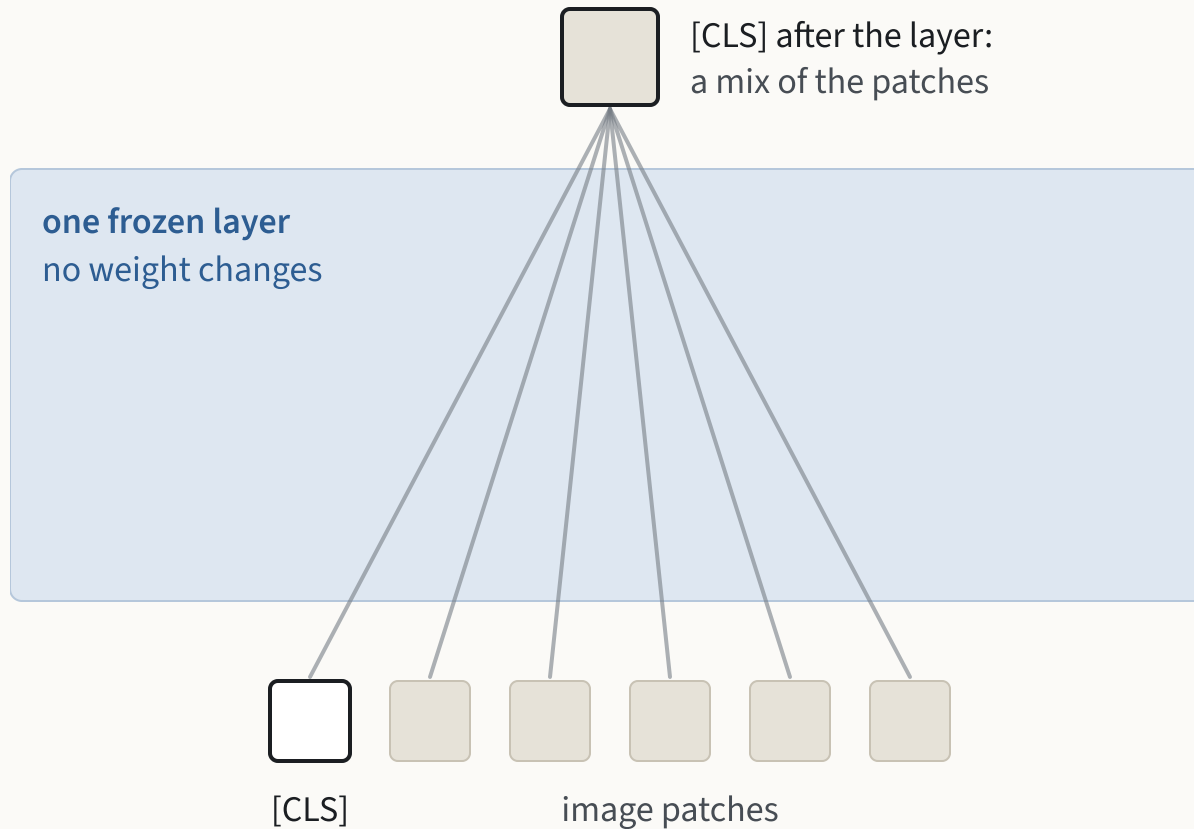
$$y = \text{Head}(x_{12})$$

Per task, only the red parts are trained and saved: 50 prompt vectors for each layer and the head, **0.63%** of the model.



(a) Visual-Prompt Tuning: Deep

Why a few extra tokens can steer a frozen network



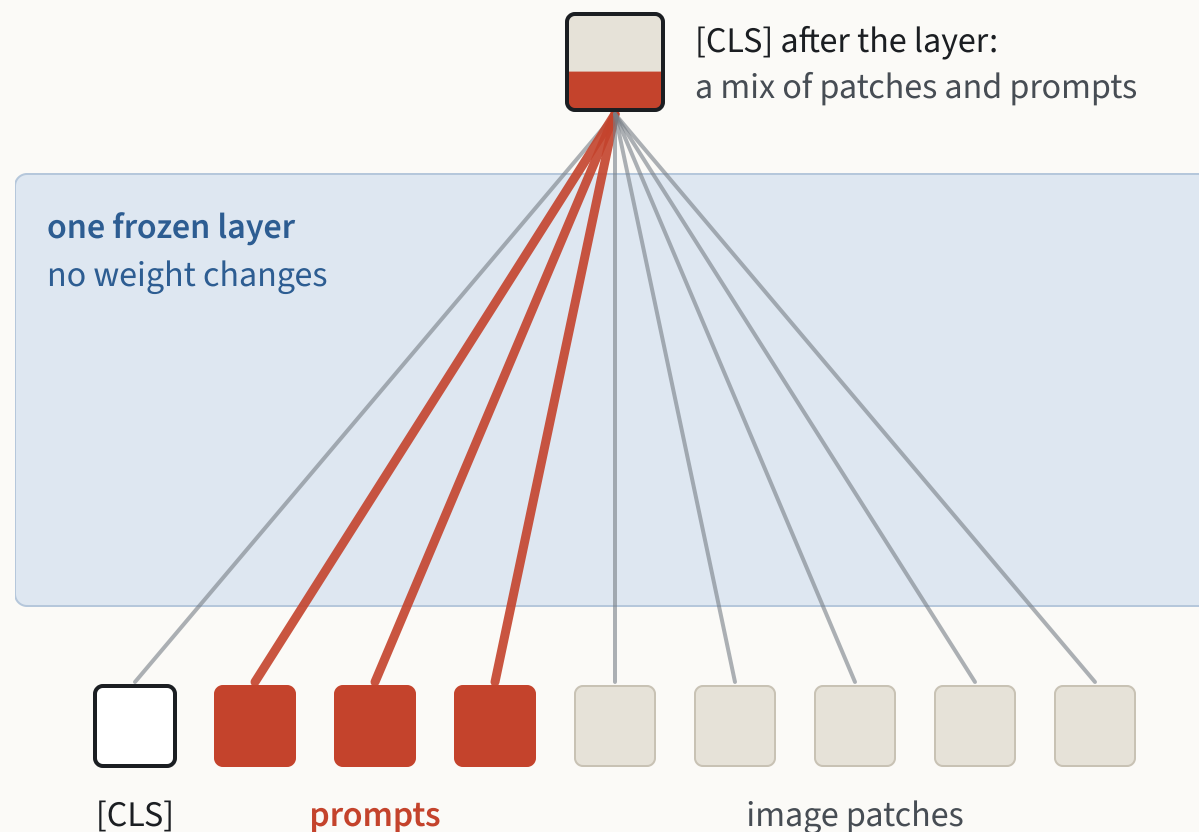
Without prompts

With prompts

In every layer, each token takes in information from all the other tokens; this is attention. With prompts in the sequence, [CLS] and every patch also take in information from the prompts.

So training the prompts changes what the frozen network computes, without changing a single weight.

Why a few extra tokens can steer a frozen network



Without prompts

With prompts

In every layer, each token takes in information from all the other tokens; this is attention. With prompts in the sequence, [CLS] and every patch also take in information from the prompts.

So training the prompts changes what the frozen network computes, without changing a single weight.

How much VPT trains and stores per task

$N \cdot p \cdot d = 12 \times 50 \times 768 = 460,800$ prompt numbers, plus 78,438 for a 102-class head

Numbers trained and stored per task

Full fine-tuning	all 85.8 million weights, retrained for this task	343 MB 100%
VPT-deep	the same 85.8 million weights, frozen and shared by every task	2.2 MB 0.63%

Compute per image

Plain ViT	197 tokens	1.00×
VPT-deep	197 tokens	1.27× +50 prompt tokens

Shown for $p = 50$; at $p = 200$ the file is 7.7 MB and compute is about 2.1×. More prompts keep the file small, but every image carries them through all 12 layers.

ViT-B/16, $N = 12$ layers, $d = 768$, 32-bit numbers. Compute counts multiply-adds per layer for n tokens ($12nd^2 + 2n^2d$). The paper measures about 2× test latency at $p = 200$ (Jia et al. 2022, Table 12).

VPT in code: a few lines and one new hyperparameter

```
# VPT-deep on a frozen ViT-B/16, simplified from demo/vpt/vpt_model.py
vit = timm.create_model(BACKBONE, pretrained=True)           # ImageNet-21k ViT-B/16
vit.requires_grad_(False)                                    # backbone frozen
prompts = nn.Parameter(torch.empty(12, p, 768).uniform_(-b, b)) # trained: 12 layers × p × 768
head = nn.Linear(768, num_classes)                           # trained: the classifier

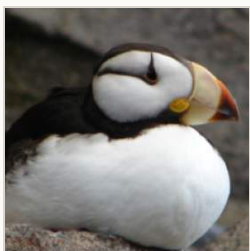
def forward(img):
    x = vit._pos_embed(vit.patch_embed(img))                  # [CLS] + 196 patch vectors
    for i, layer in enumerate(vit.blocks):
        rest = x[:, 1:] if i == 0 else x[:, 1 + p:]           # drop last layer's prompts
        P = prompts[i].expand(len(x), -1, -1)                 # this layer's prompts
        x = layer(torch.cat([x[:, :1], P, rest], dim=1))       # frozen layer on [CLS, P, rest]
    return head(vit.norm(x)[:, 0])                             # classify from [CLS]
```

The prompts start random, with Xavier-uniform values as in the official code. The paper tried smarter starts, such as class prototypes, and random did as well or better.

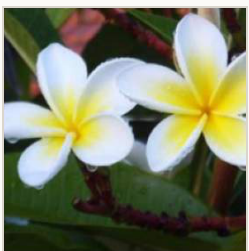
Training is plain SGD with momentum for 100 epochs on a cosine schedule. Learning rate and weight decay are picked on a validation split, for every method alike.

The one new hyperparameter is the prompt length p , chosen per task from {1, 5, 10, 50, 100, 200} on the validation split.

The paper tests VPT on 24 very different tasks



CUB-200



Oxford Flowers



Caltech101



DTD



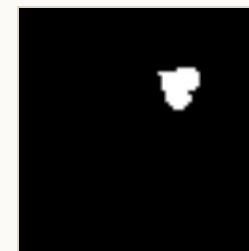
EuroSAT



Resisc45



CLEVR count



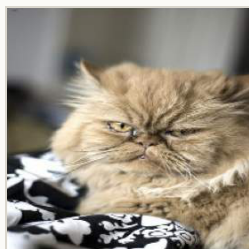
dSprites location



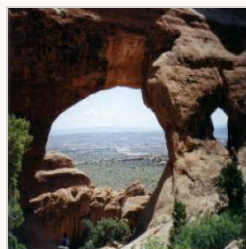
Stanford Cars



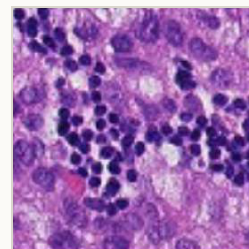
Stanford Dogs



Oxford Pets



SUN397



Patch Camelyon



Retinopathy



KITTI distance



SmallNORB azimuth

FGVC, 5 tasks

Fine-grained categories, such as 200 bird species, with thousands of training images.

VTAB Natural, 7 tasks

Ordinary photos of objects, textures, animals and scenes, close to the pretraining data.

VTAB Specialized, 4 tasks

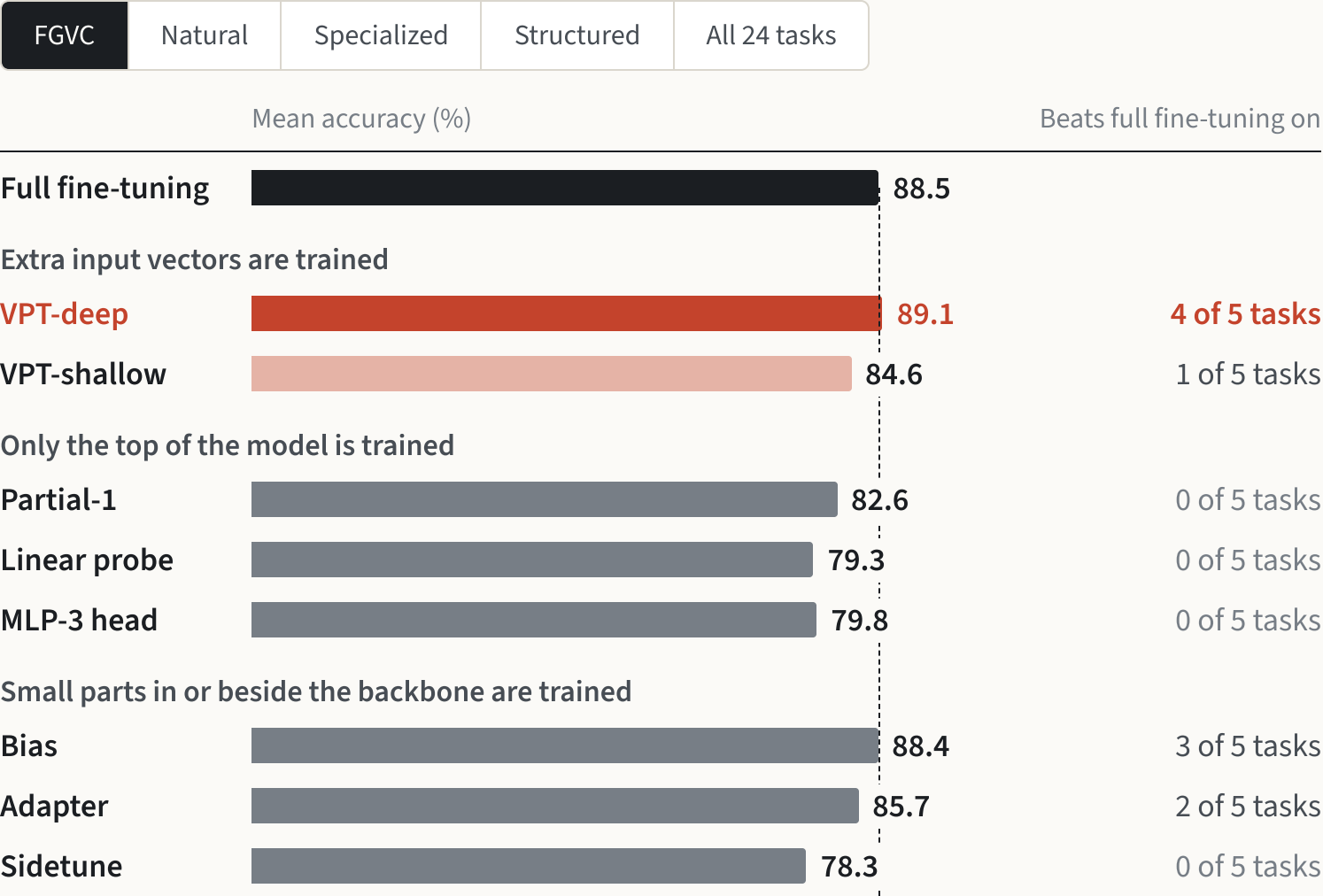
Images from special sensors: satellites, tissue slides under a microscope, and retina cameras.

VTAB Structured, 8 tasks

Geometry instead of categories, such as counting objects or judging distance.

All 24 tasks use the same frozen ViT-B/16, pretrained on ImageNet-21k. The 19 VTAB tasks have only 1,000 training images each.

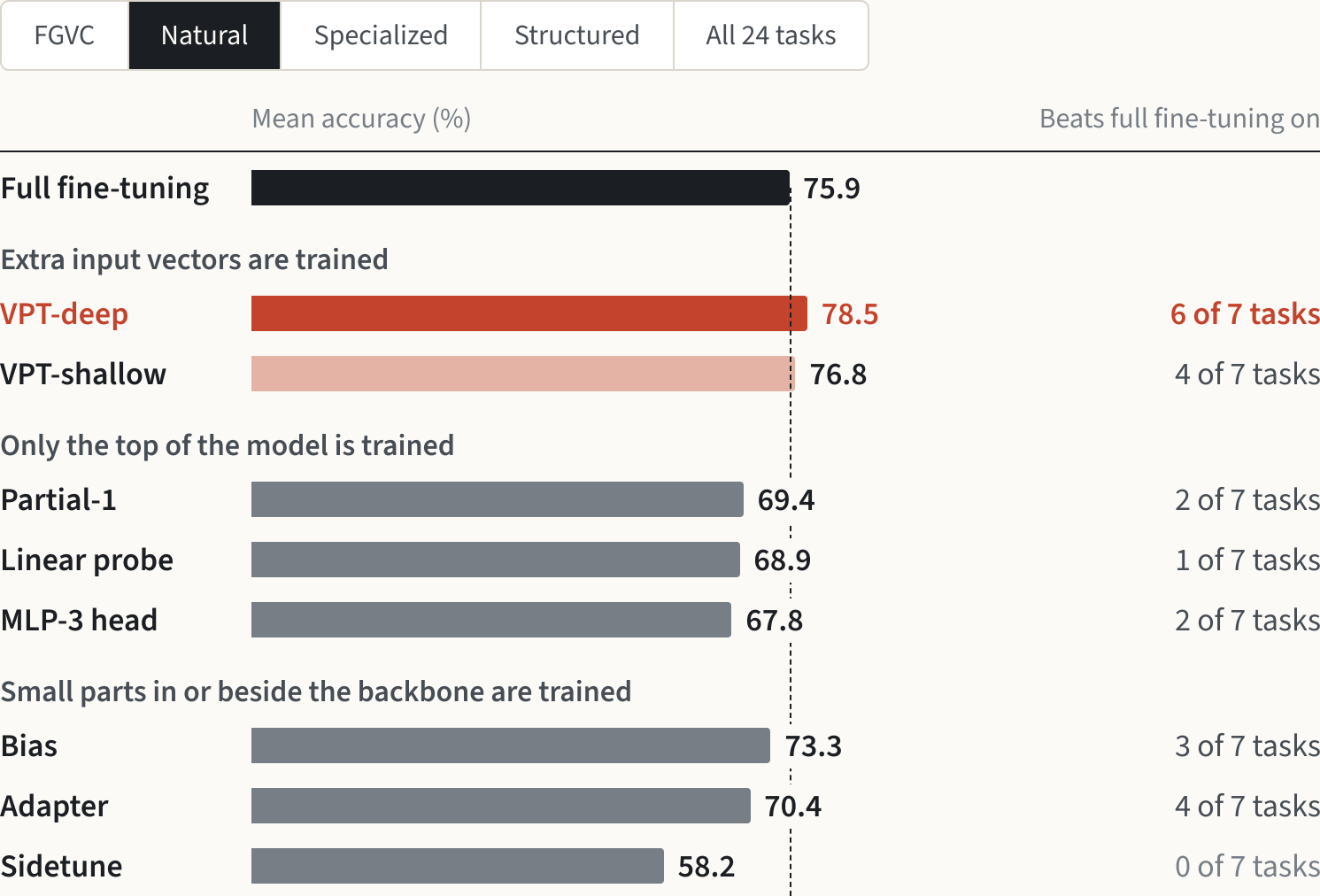
VPT-deep beats full fine-tuning on 20 of 24 tasks



These five tasks ask the model to tell apart 200 bird species, 196 car models, 120 dog breeds or 102 kinds of flower, with thousands of training images per task.

Every method does well on such photos, and VPT-deep only edges out full fine-tuning, 89.1 against 88.5.

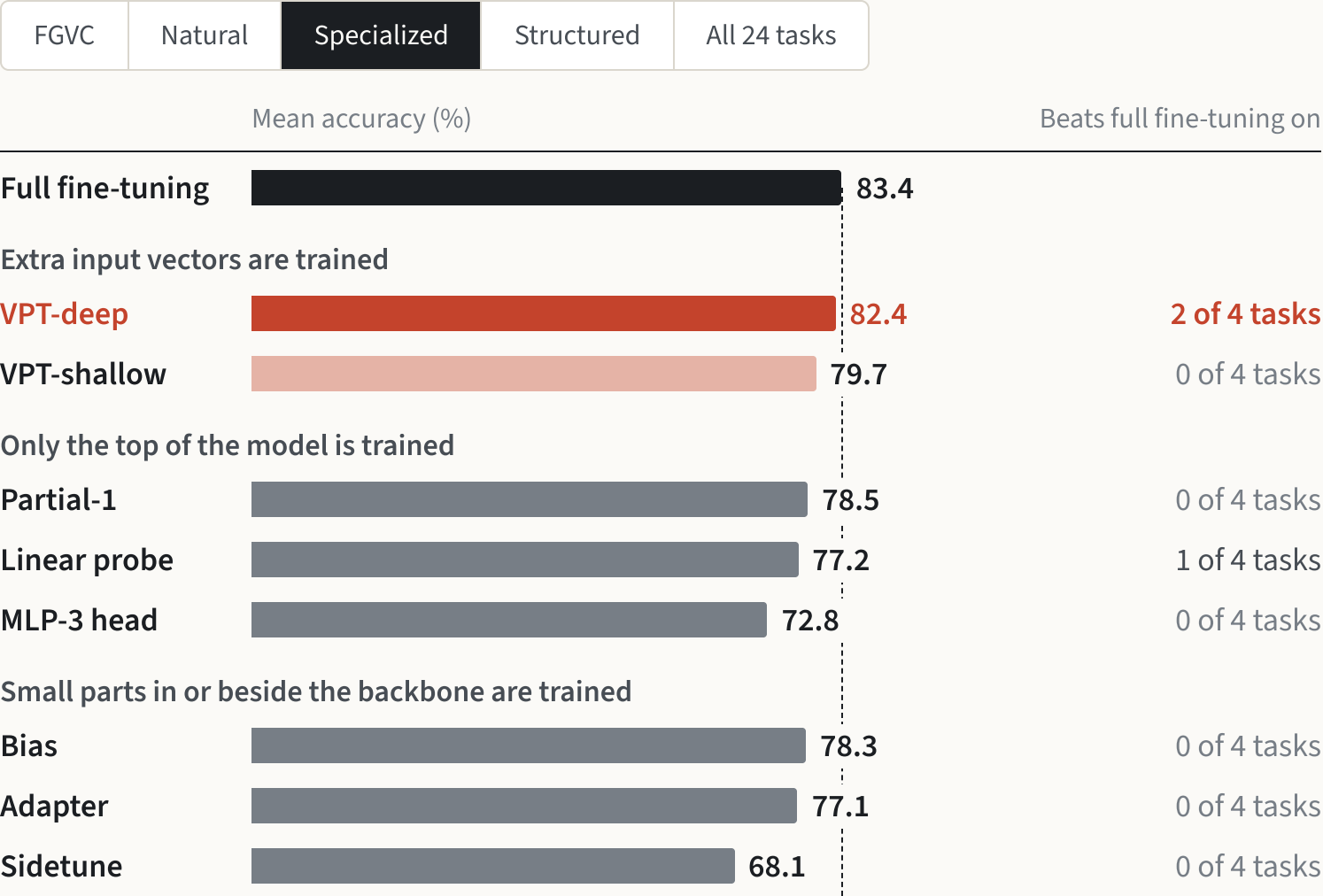
VPT-deep beats full fine-tuning on 20 of 24 tasks



These seven tasks use ordinary photos of objects, pets, textures, scenes and house numbers, close to what the model saw in pretraining, but with only 1,000 training images each.

With so little data, retraining all 86 million weights overfits, so even VPT-shallow beats full fine-tuning here.

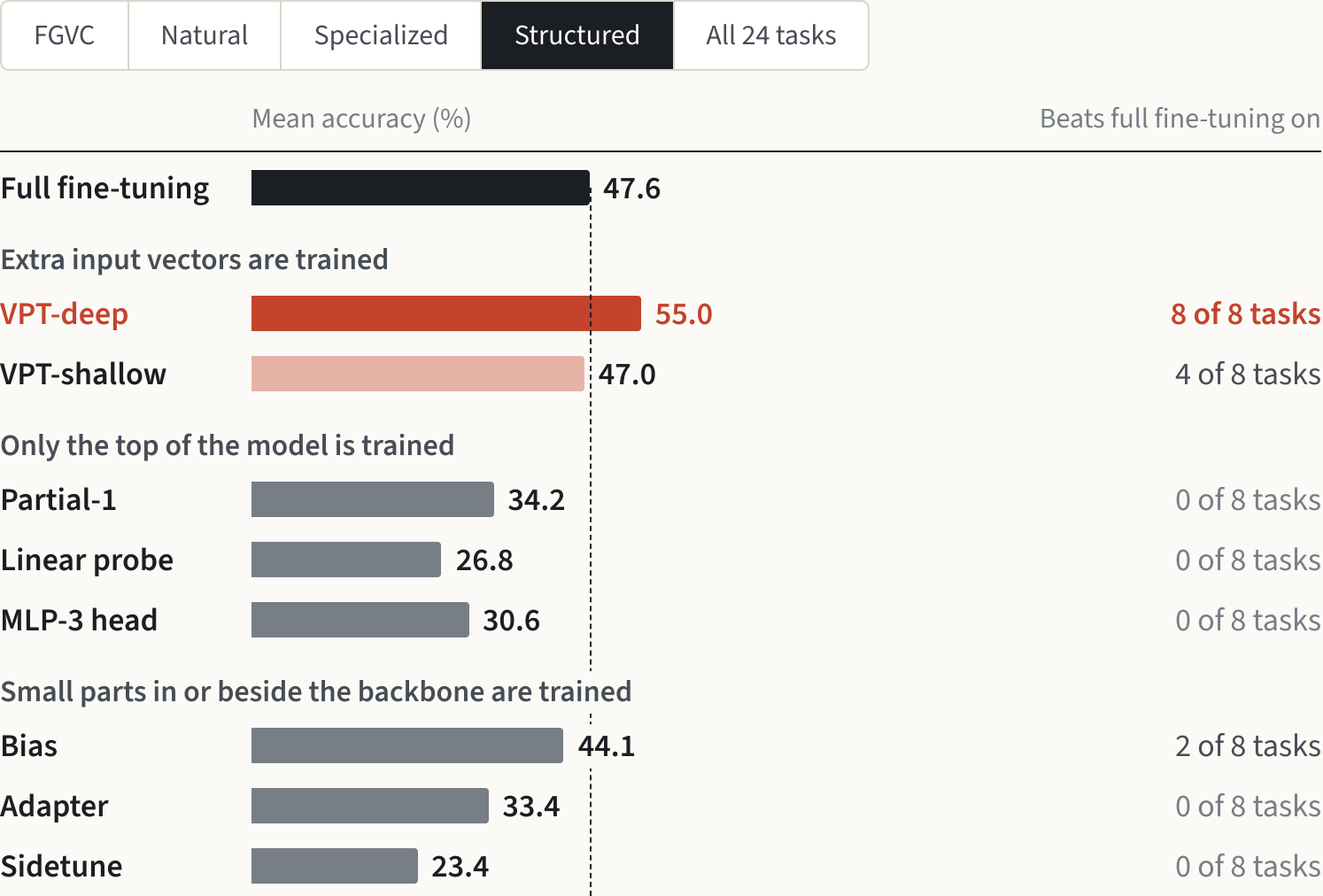
VPT-deep beats full fine-tuning on 20 of 24 tasks



These four tasks come from special cameras: satellite images of land use, microscope slides of tumour tissue and photos of the retina, with 1,000 training images each.

This is the one group where full fine-tuning stays ahead, by one point, but VPT-deep is still the best of the cheap methods.

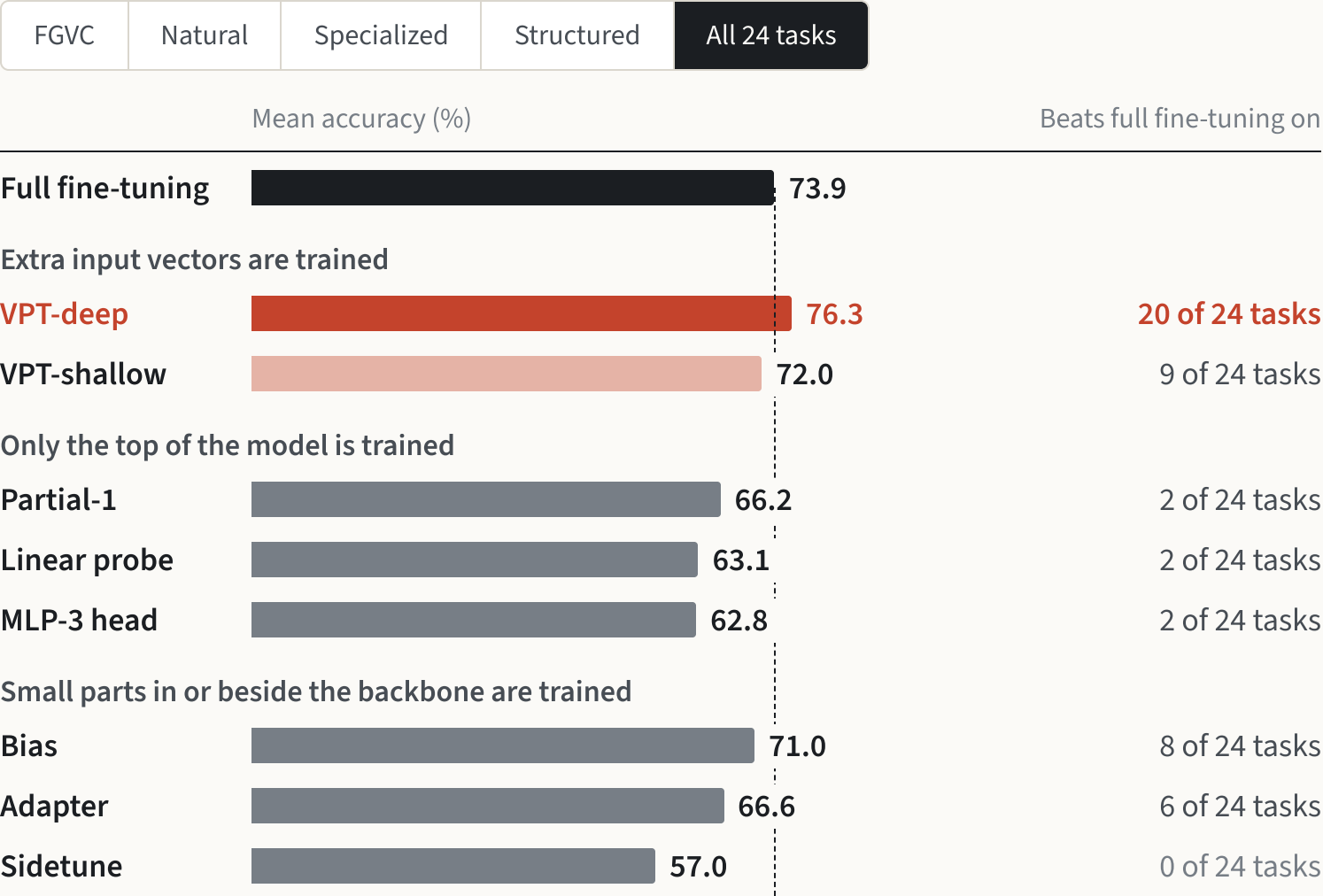
VPT-deep beats full fine-tuning on 20 of 24 tasks



These eight tasks ask about geometry rather than objects: how many things there are, how far away, at what angle. Most are synthetic scenes, far from ImageNet.

Here VPT-deep wins all eight, 55.0 against 47.6, while the linear probe falls to 26.8, so the frozen features alone do not hold this information.

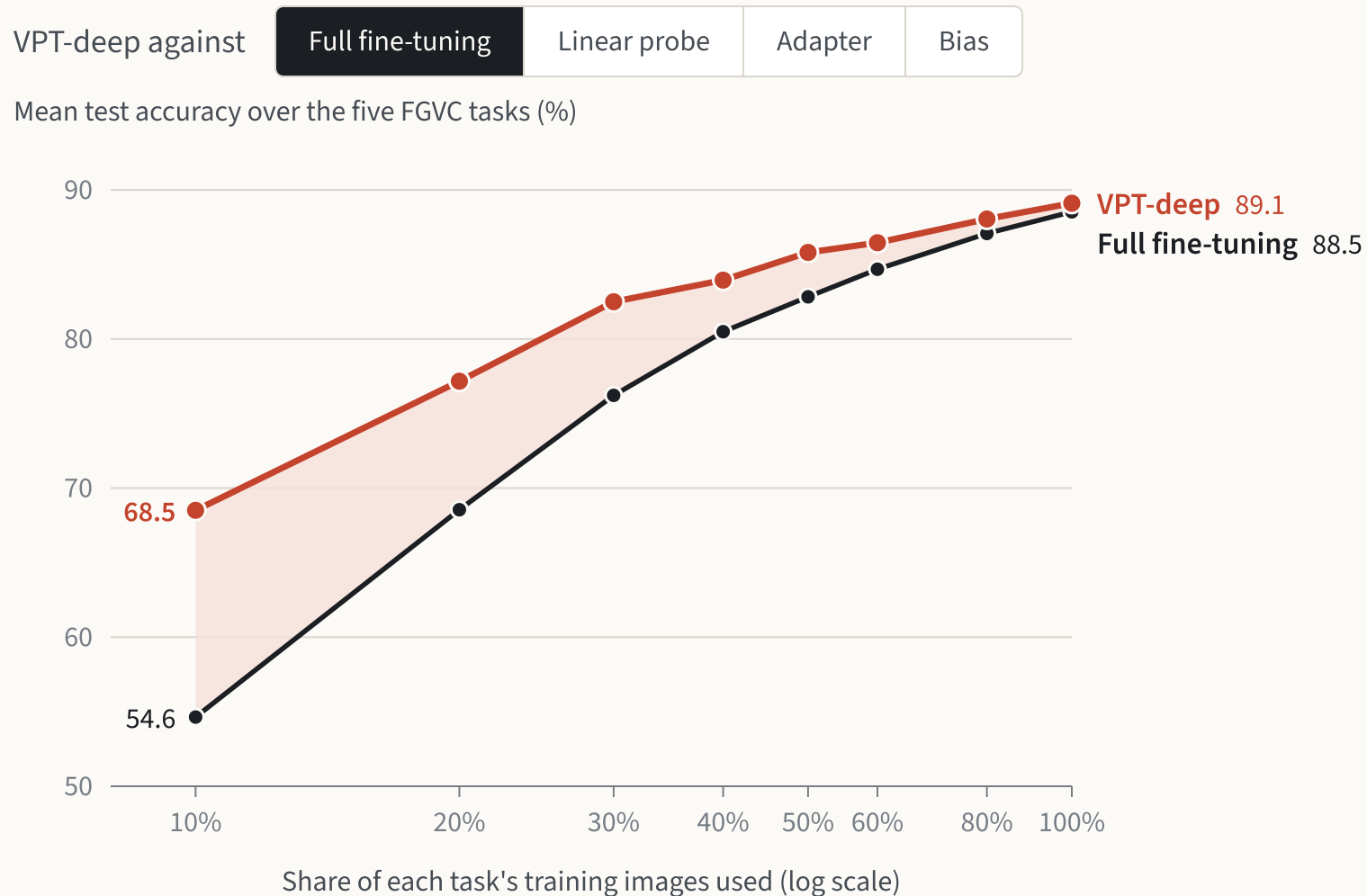
VPT-deep beats full fine-tuning on 20 of 24 tasks



Averaged over all 24 tasks, VPT-deep is the only cheap method above full fine-tuning, 76.3 against 73.9.

It gets there while storing 1.18 backbones' worth of weights for the 24 tasks, where full fine-tuning stores 24.

VPT leads with little data and stays ahead with more

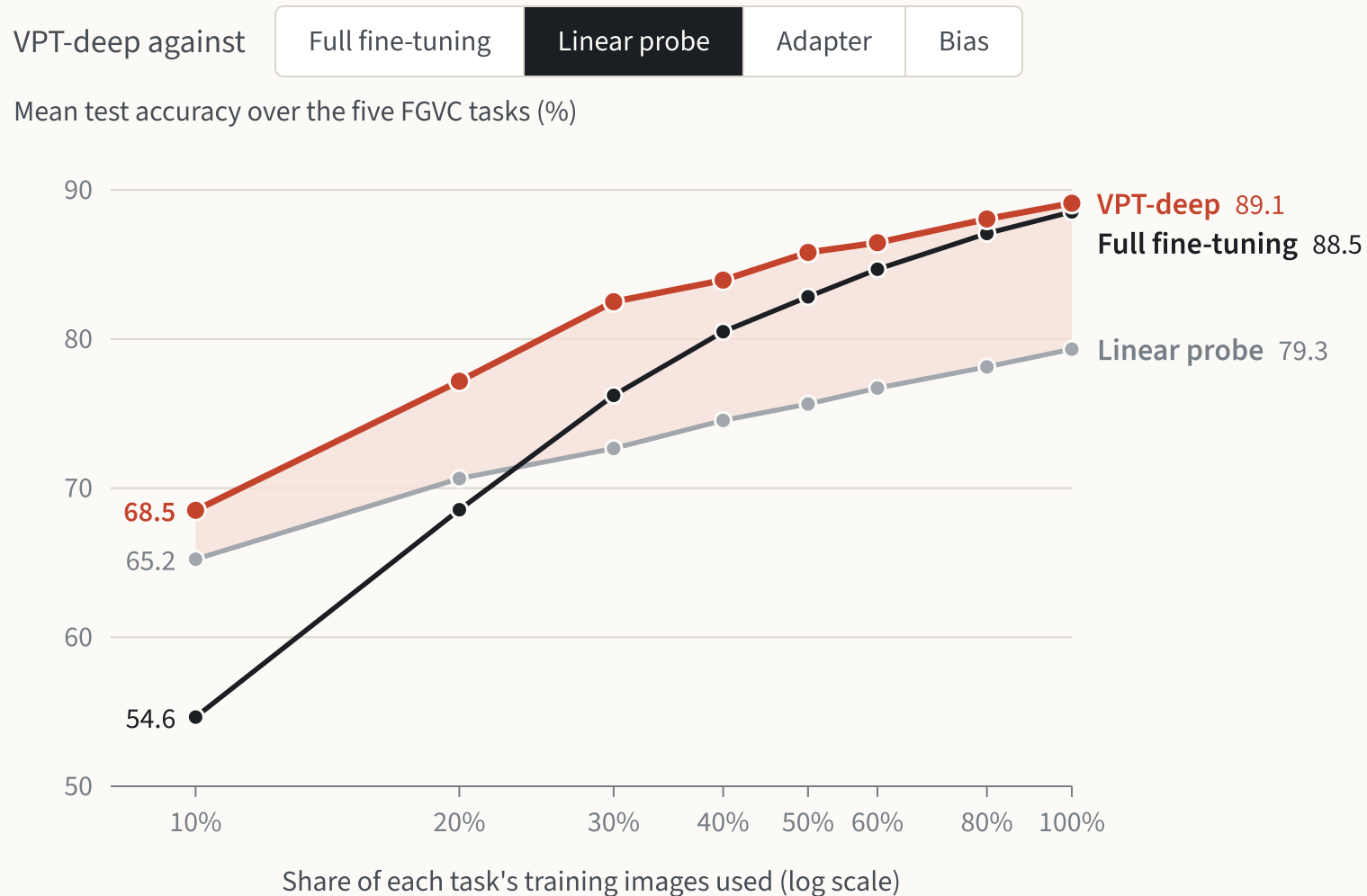


With a tenth of the training images, full fine-tuning has too many weights to fit and falls 14 points behind VPT-deep.

The gap shrinks as the data grows, but VPT-deep stays ahead all the way to the full training set.

Jia et al. 2022, Fig. 3: ViT-B/16, mean test accuracy over the five FGVC tasks, each method trained on a share of every task's training images. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the 100% points match Table 1.

VPT leads with little data and stays ahead with more

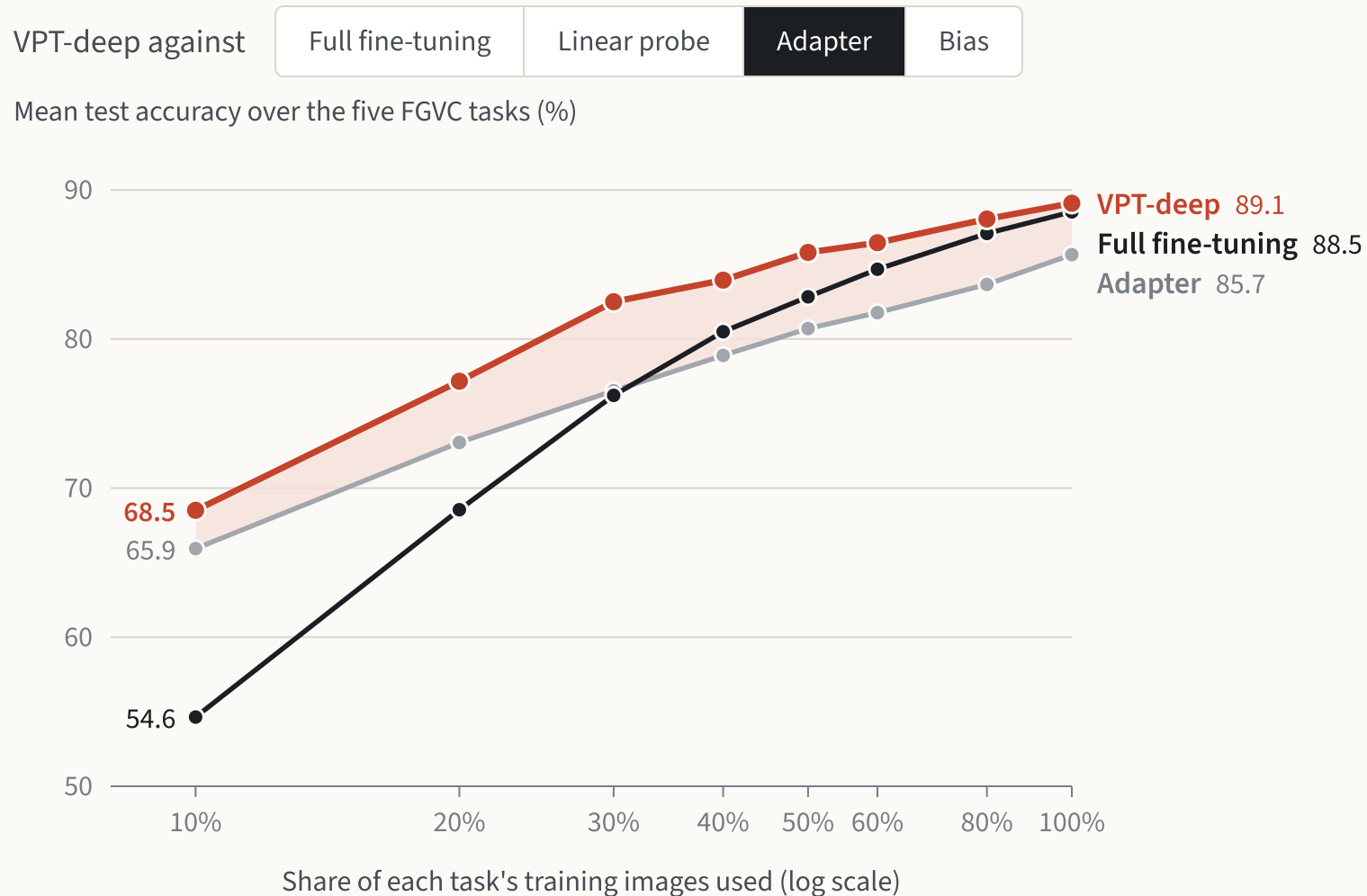


The linear probe trains only a new output layer, so it does well when data is scarce.

From 30% of the data on, full fine-tuning overtakes it, and with all the data it trails VPT-deep by almost 10 points.

Jia et al. 2022, Fig. 3: ViT-B/16, mean test accuracy over the five FGVC tasks, each method trained on a share of every task's training images. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the 100% points match Table 1.

VPT leads with little data and stays ahead with more

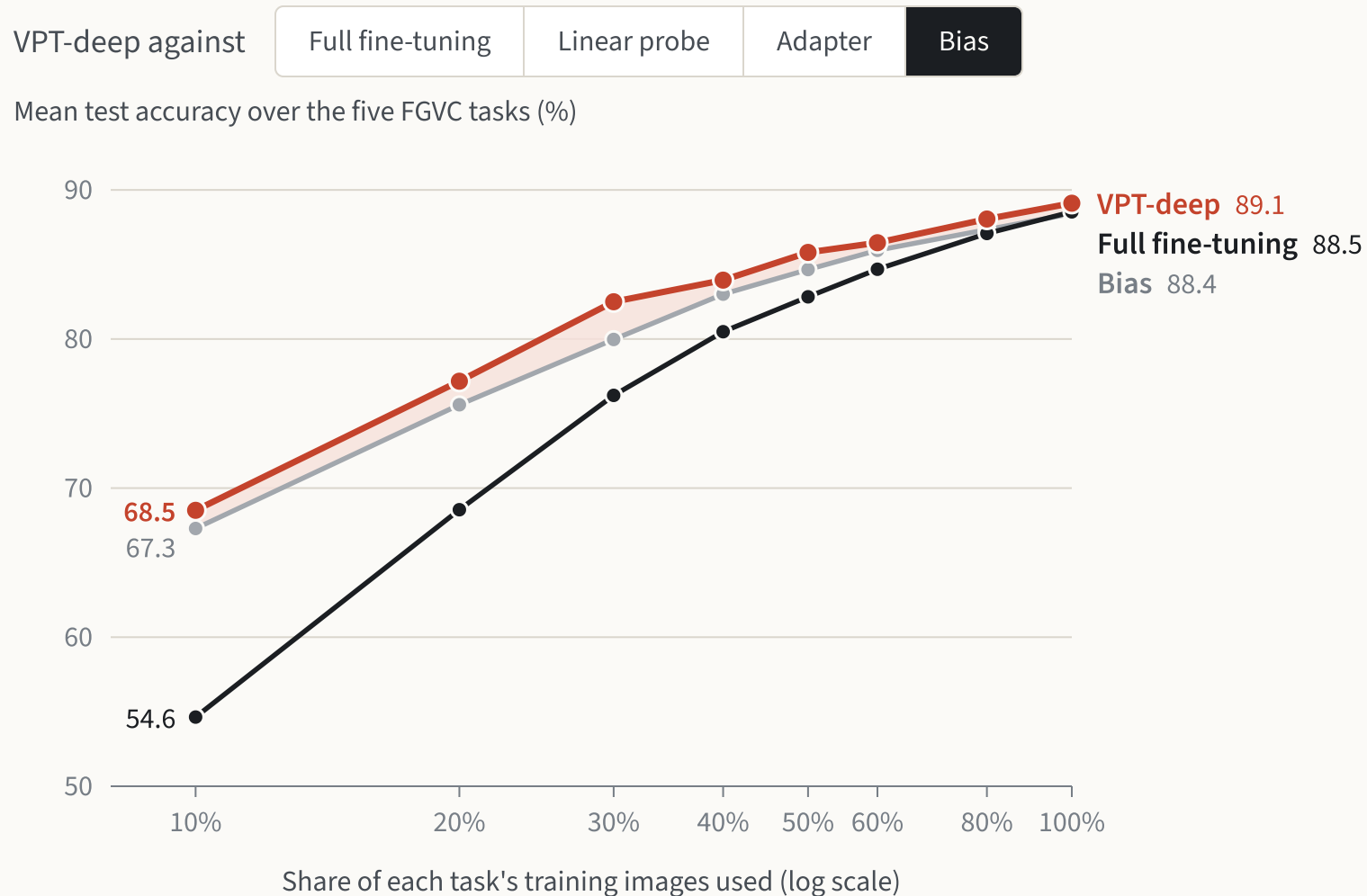


The adapter inserts small trained blocks into every layer, and it too beats full fine-tuning on little data.

Full fine-tuning passes it at 40% of the data, while VPT-deep stays 3 to 6 points above it throughout.

Jia et al. 2022, Fig. 3: ViT-B/16, mean test accuracy over the five FGVC tasks, each method trained on a share of every task's training images. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the 100% points match Table 1.

VPT leads with little data and stays ahead with more

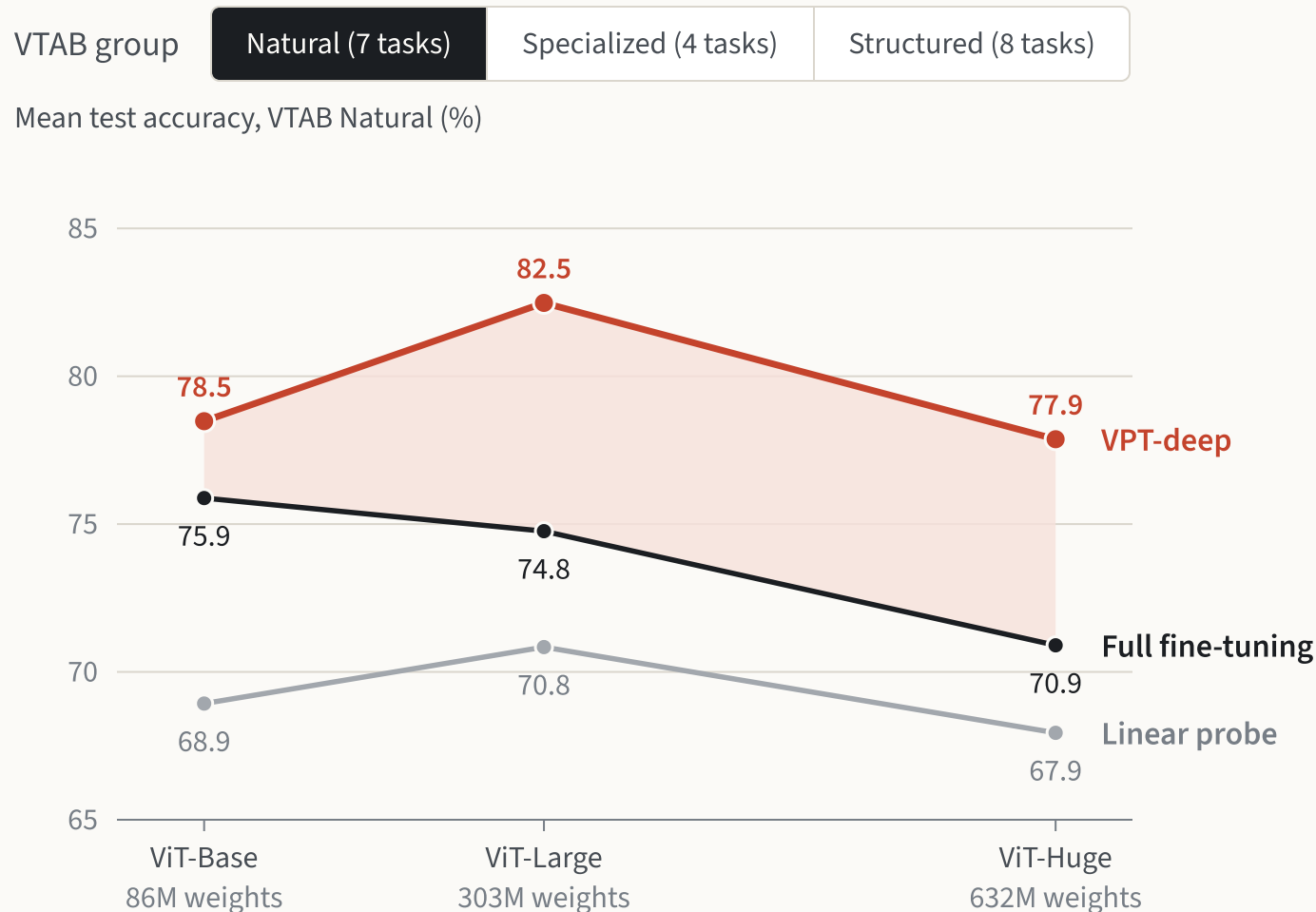


Bias trains only the bias terms and is the closest rival: it beats full fine-tuning up to 80% of the data.

VPT-deep is still ahead at every point, by half a point to 2.5 points.

Jia et al. 2022, Fig. 3: ViT-B/16, mean test accuracy over the five FGVC tasks, each method trained on a share of every task's training images. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the 100% points match Table 1.

The result holds for larger backbones too

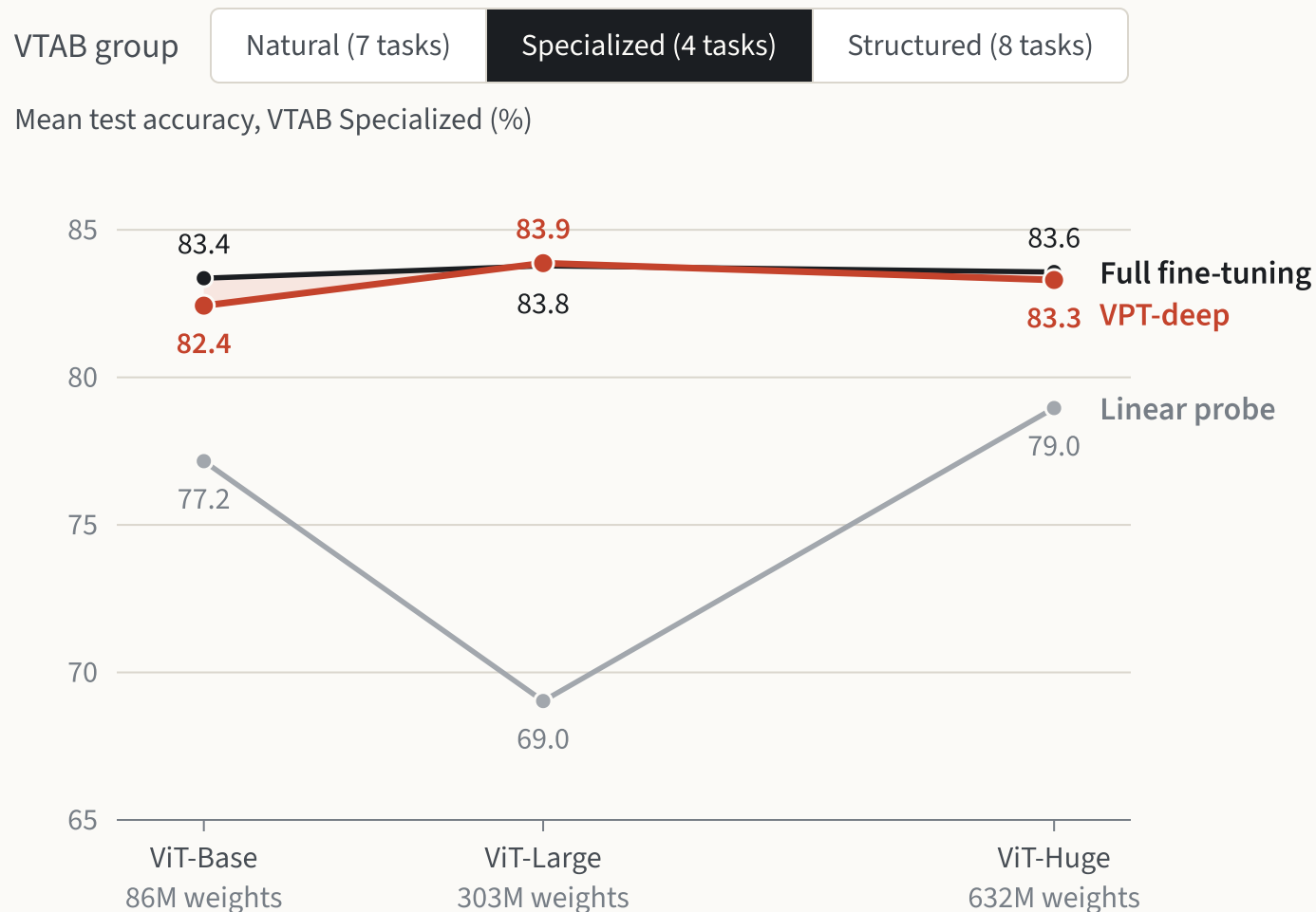


On ordinary photos, VPT-deep beats full fine-tuning with all three backbones: by 2.6 points on ViT-Base and by 7 to 8 points on the larger two.

Full fine-tuning even gets worse as the model grows, from 75.9 to 70.9.

Jia et al. 2022, Fig. 4: VTAB-1k, mean test accuracy per group, backbones pretrained on ImageNet-21k. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the ViT-B points match Table 1.

The result holds for larger backbones too

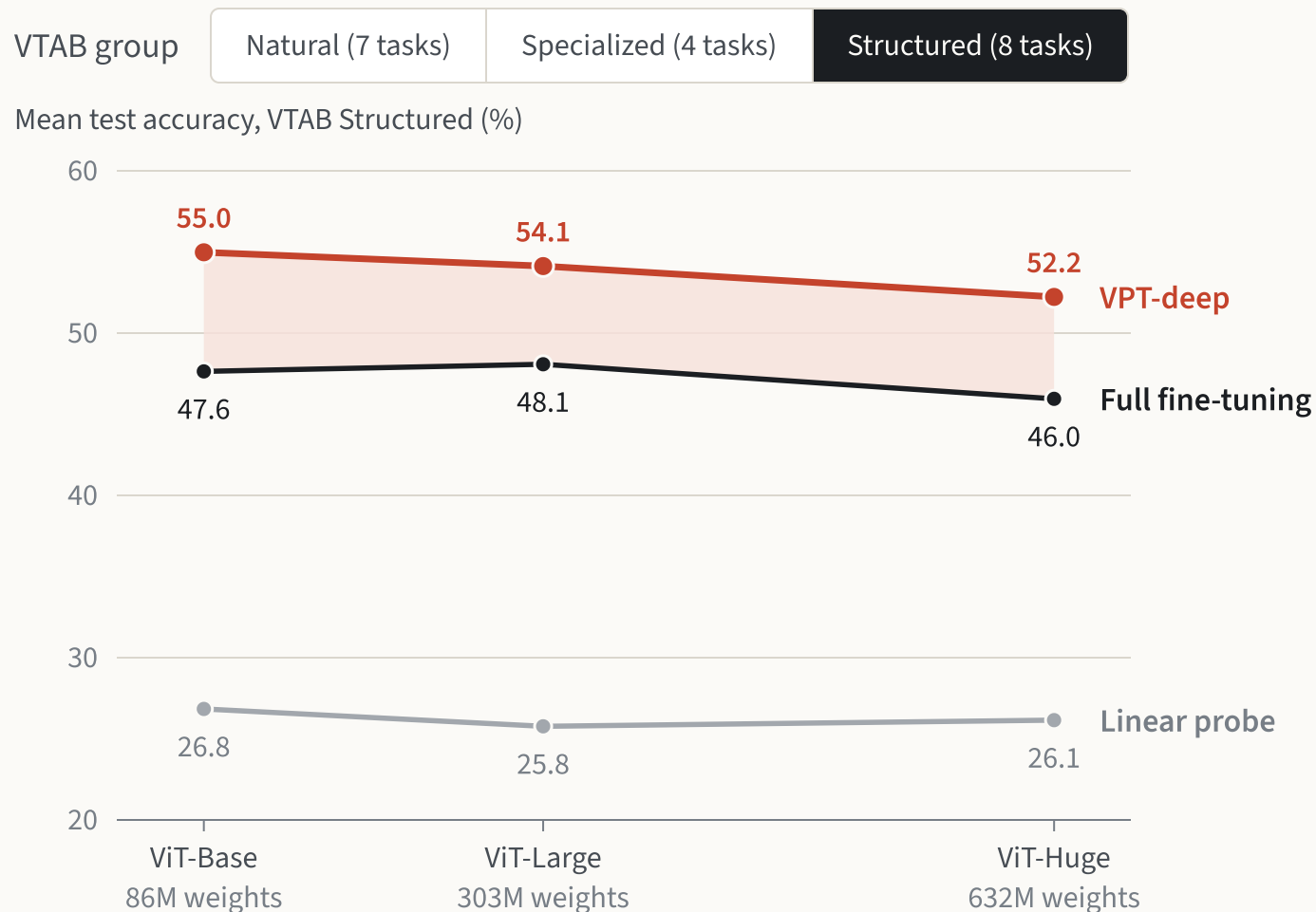


On medical and satellite images, the two are level with every backbone and never differ by more than one point.

This was already the one group where full fine-tuning kept a small lead on ViT-Base, and larger backbones do not change that.

Jia et al. 2022, Fig. 4: VTAB-1k, mean test accuracy per group, backbones pretrained on ImageNet-21k. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the ViT-B points match Table 1.

The result holds for larger backbones too

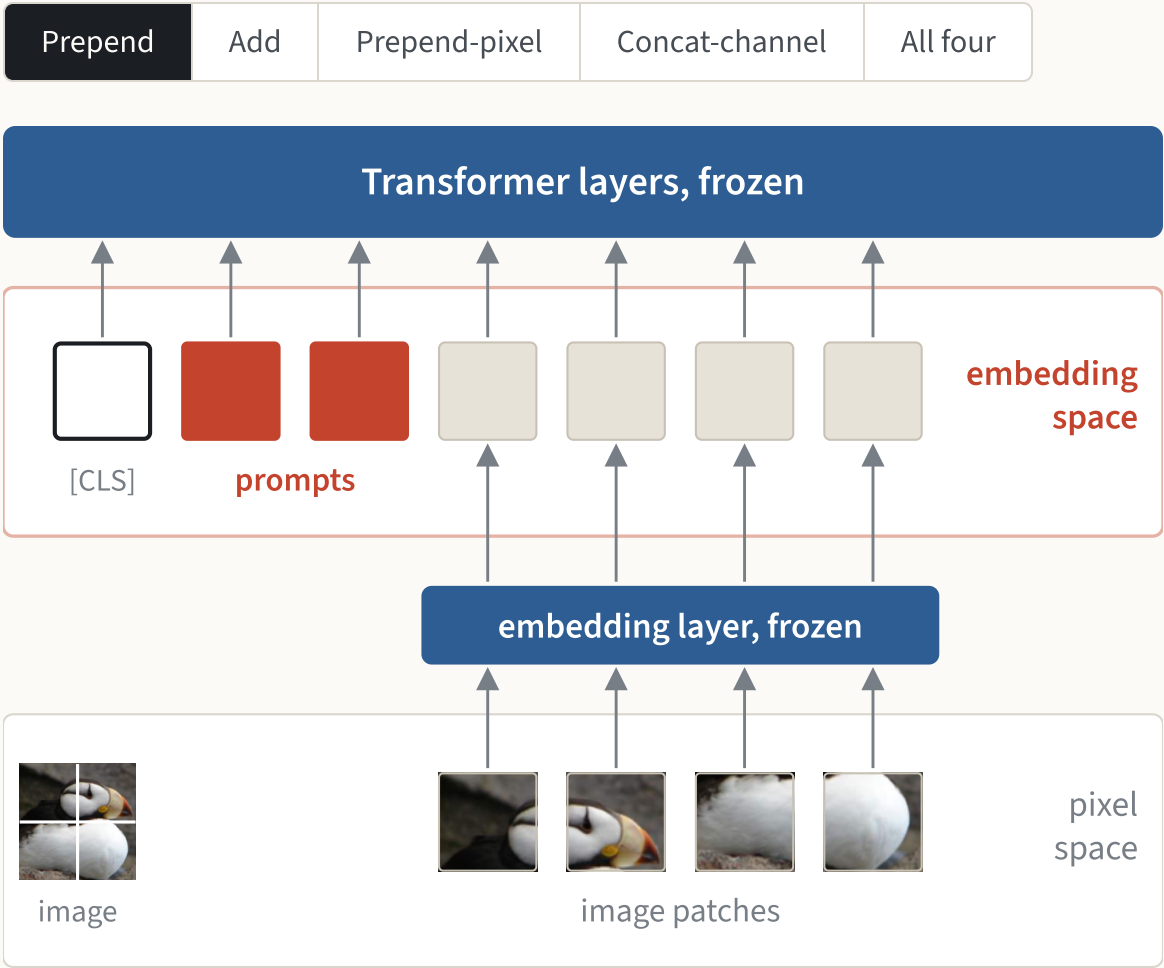


On counting and distance tasks, VPT-deep leads by 6 to 7 points with every backbone.

The linear probe stays near 26%, so the frozen features alone are not enough here, at any size.

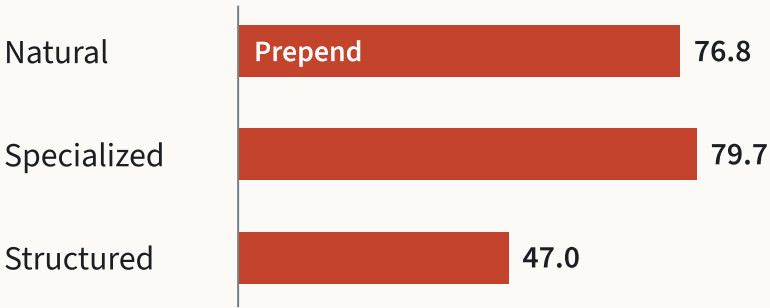
Jia et al. 2022, Fig. 4: VTAB-1k, mean test accuracy per group, backbones pretrained on ImageNet-21k. The paper gives these values only as a figure; they were read from its vector drawing in the PDF, and the ViT-B points match Table 1.

Four places to put the prompt



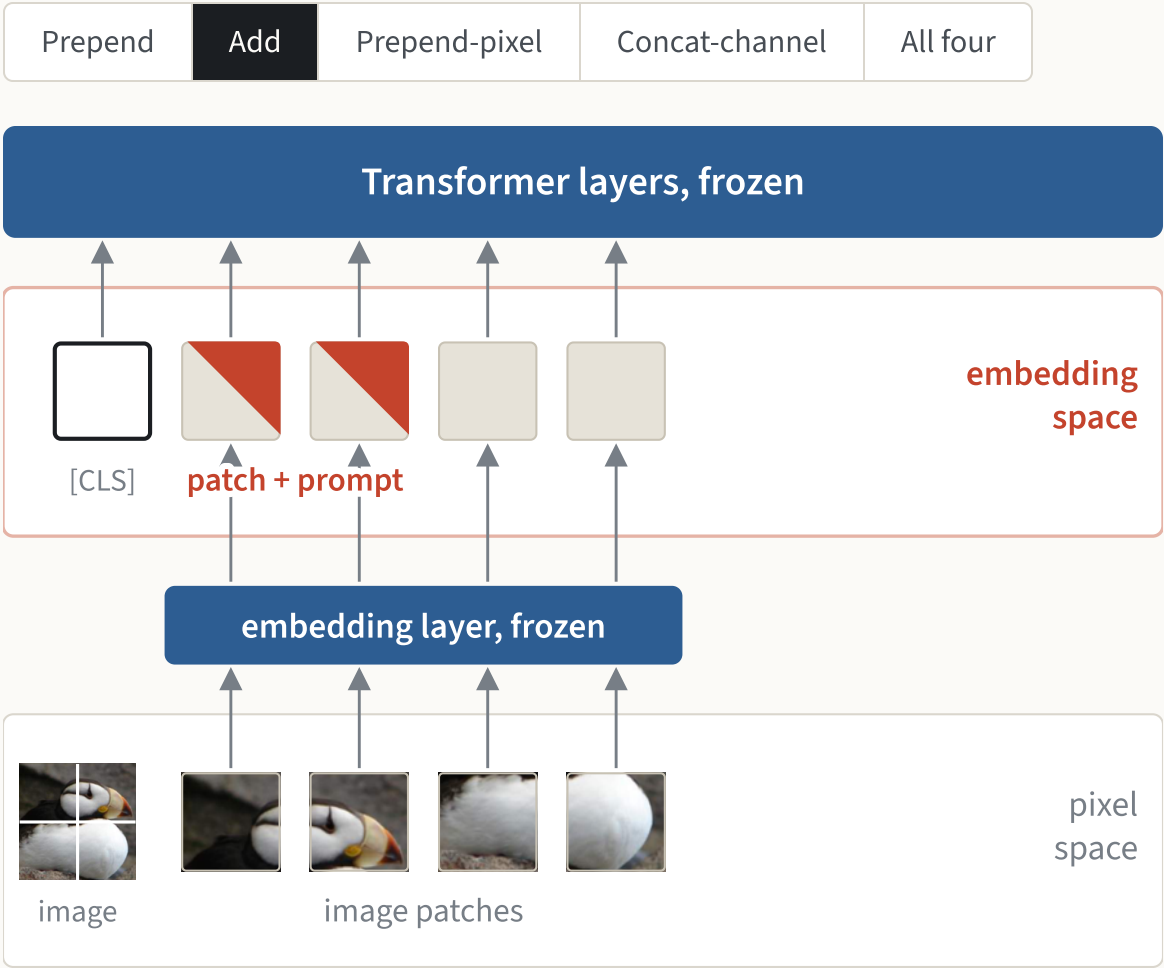
This is the default. The prompts are extra vectors placed next to the patch vectors, after the embedding layer, so every layer can attend to them as tokens of their own.

VPT-shallow accuracy (%)

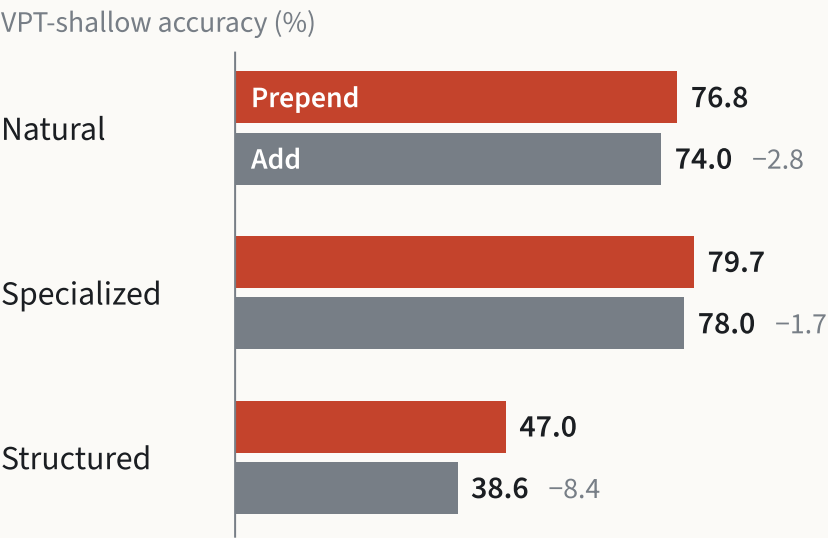


Numbers from Jia et al. 2022, Fig. 5 (ViT-B/16, prompts in the first layer only); "All four" is the mean of the three groups. Diagrams redrawn.

Four places to put the prompt

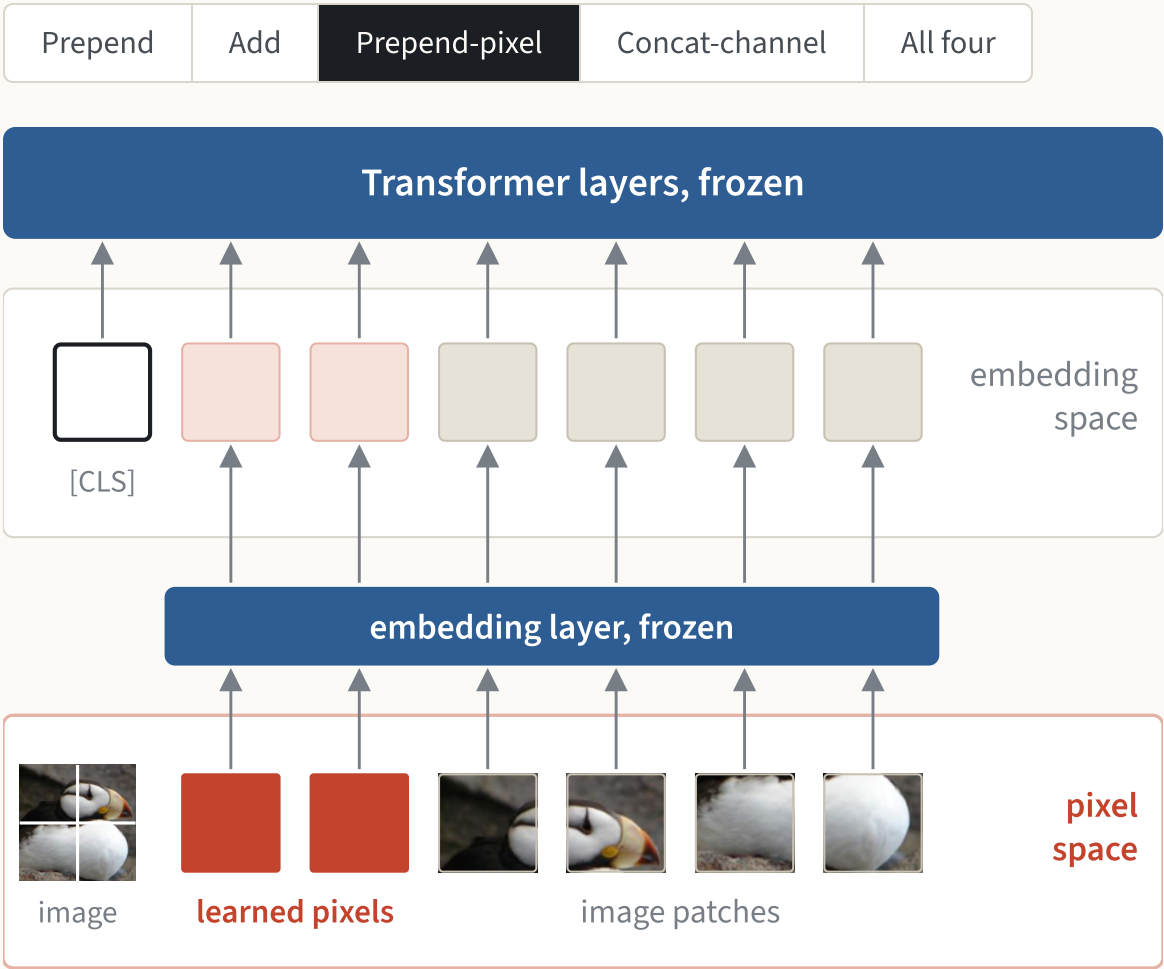


Here each prompt vector is added onto a patch vector, so the sequence keeps its length. This scores lower in all three groups, by 8.4 points on Structured.

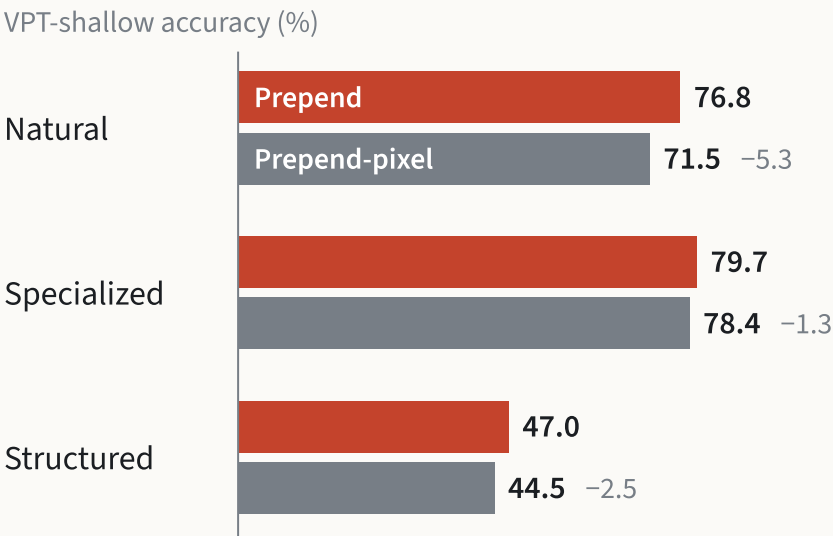


Numbers from Jia et al. 2022, Fig. 5 (ViT-B/16, prompts in the first layer only); "All four" is the mean of the three groups. Diagrams redrawn.

Four places to put the prompt

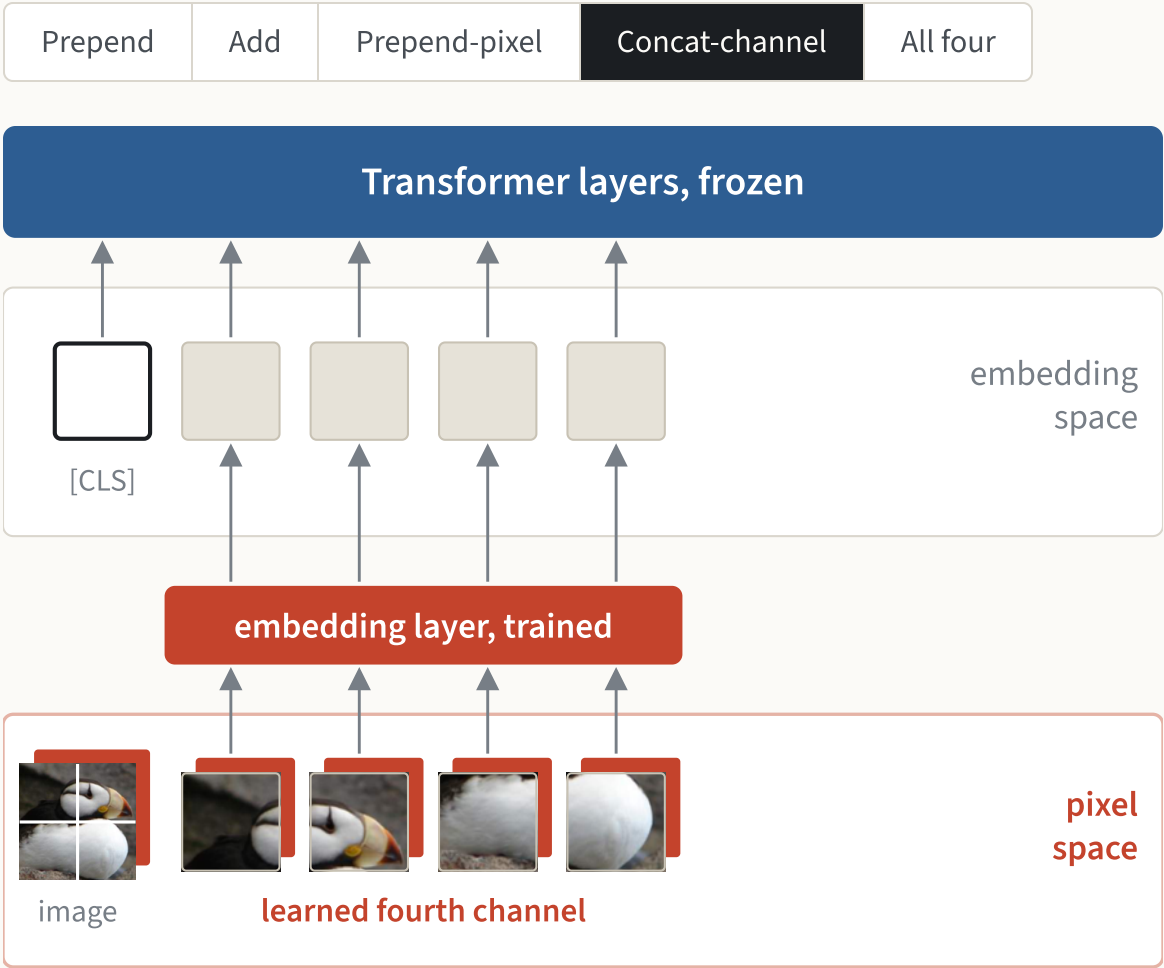


Here the prompts are learned pixel patches, placed next to the image patches before the frozen embedding layer. This also scores lower in every group, by 5.3 points on Natural.



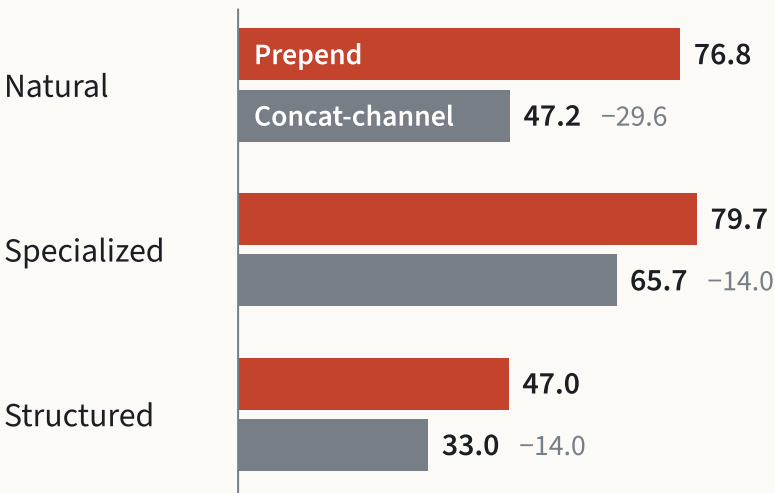
Numbers from Jia et al. 2022, Fig. 5 (ViT-B/16, prompts in the first layer only); "All four" is the mean of the three groups. Diagrams redrawn.

Four places to put the prompt



Here the prompt is a learned fourth colour channel on the image, and the embedding layer is trained to read it. This is by far the worst: 29.6 points lower on Natural, below even a linear probe there.

VPT-shallow accuracy (%)



Numbers from Jia et al. 2022, Fig. 5 (ViT-B/16, prompts in the first layer only); "All four" is the mean of the three groups. Diagrams redrawn.

Four places to put the prompt

Prepend

Add

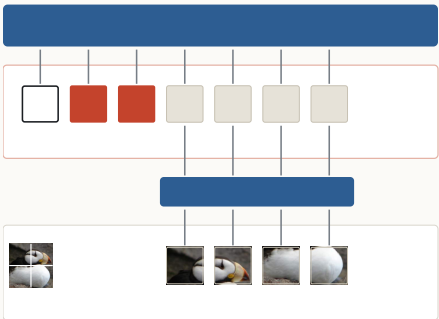
Prepend-pixel

Concat-channel

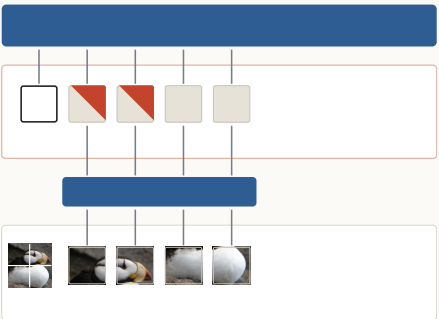
All four

Prompt in the embedding space

Prepend

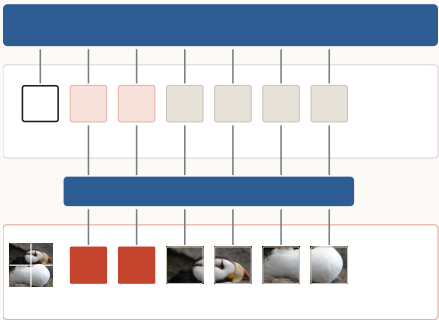


Add

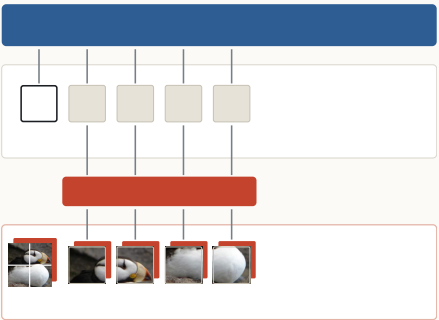


Prompt in pixel space

Prepend-pixel

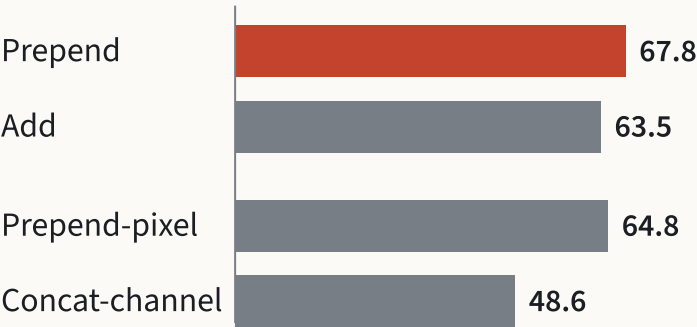


Concat-channel



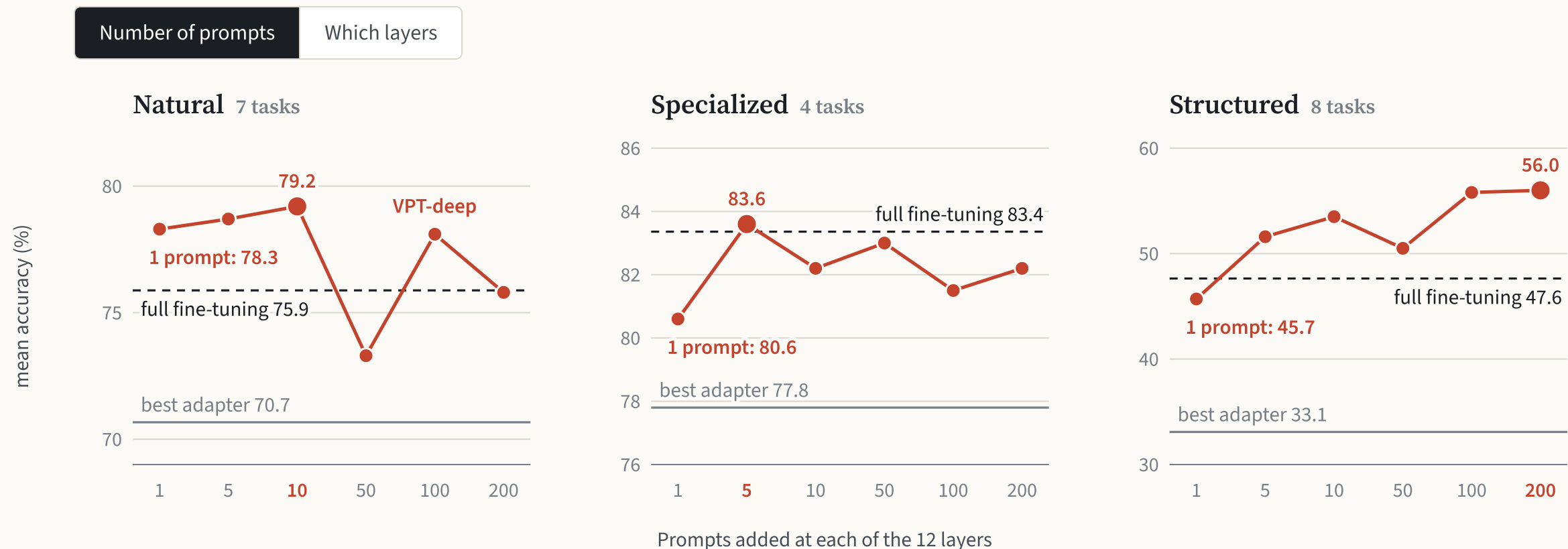
Prepend scores highest in every group. A prompt works best as separate vectors in the embedding space; adding it onto the patches or moving it into the pixels costs accuracy.

VPT-shallow accuracy (%), mean of the three groups



Numbers from Jia et al. 2022, Fig. 5 (ViT-B/16, prompts in the first layer only); "All four" is the mean of the three groups. Diagrams redrawn.

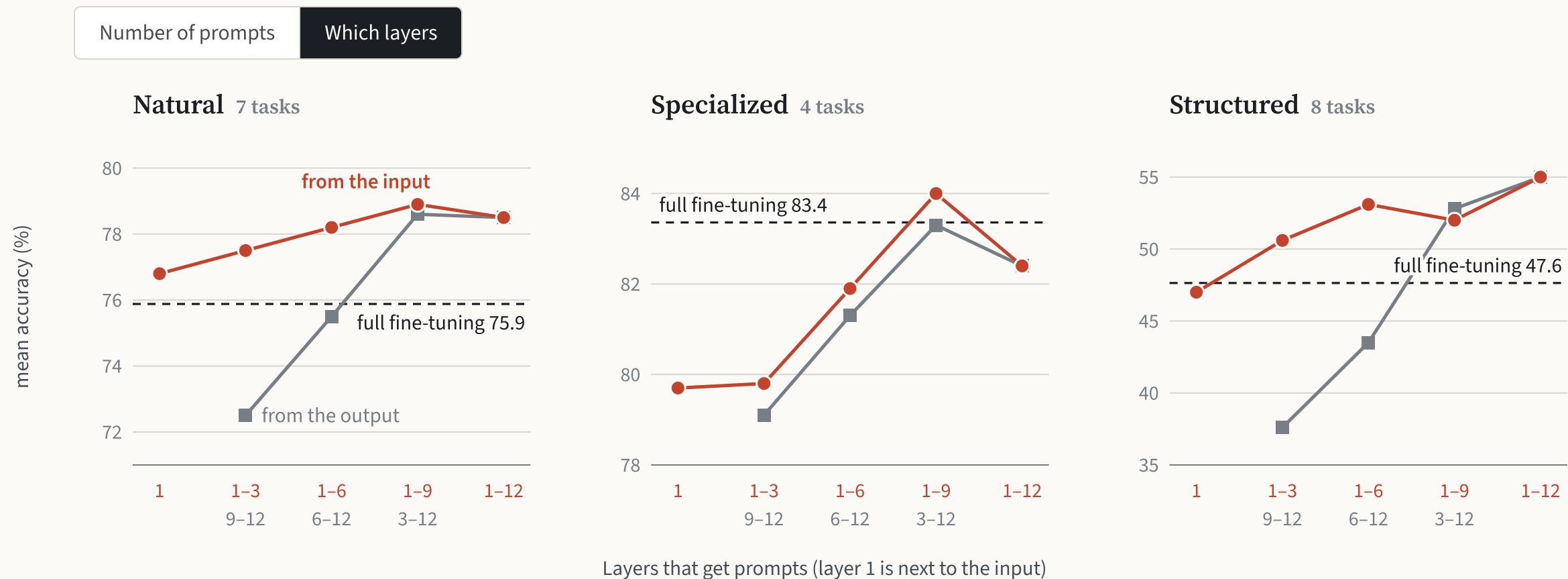
How many prompts, and at which layers



Each group prefers a different number: Natural does best with 10 prompts per layer, Specialized with 5 and Structured with 200. Even a single prompt per layer beats the best adapter in every group, and on Natural it already beats full fine-tuning.

Jia et al. 2022, Fig. 6: VPT-deep on ViT-B/16, mean test accuracy per VTAB group. The paper gives these only as a figure; values are read off its plotted points. Full fine-tuning and the adapter (best of three sizes): Table 13.

How many prompts, and at which layers



Prompting more layers generally helps: layer 1 alone is VPT-shallow, all twelve is VPT-deep. Layers next to the input matter most: prompts placed only in the layers near the output score lower, the opposite of what was found for text prompts.

Jia et al. 2022, Fig. 7, each variant with its best prompt length. Values are read off the figure's plotted points; layer 1 alone and layers 1-12 match VPT-shallow and VPT-deep in Table 13.

With prompts, the same frozen network separates the classes

House numbers (SVHN)

Satellite land use (EuroSAT)

Counting objects (CLEVR)

Linear probe

36.6%

frozen network, no prompts



VPT-deep

78.1%

same frozen network, with prompts



Full fine-tuning

87.4%

every weight retrained



Each dot is one test image, coloured by its true class; the number is test accuracy. With the frozen features alone, the classes overlap. With prompts, the same frozen network produces clearly separated groups, so a simple linear head can tell them apart.

With prompts, the same frozen network separates the classes

House numbers (SVHN)

Satellite land use (EuroSAT)

Counting objects (CLEVR)

Linear probe

87.5%

frozen network, no prompts



VPT-deep

96.1%

same frozen network, with prompts



Full fine-tuning

95.7%

every weight retrained



Each dot is one test image, coloured by its true class; the number is test accuracy. With the frozen features alone, the classes overlap. With prompts, the same frozen network produces clearly separated groups, so a simple linear head can tell them apart.

With prompts, the same frozen network separates the classes

House numbers (SVHN)

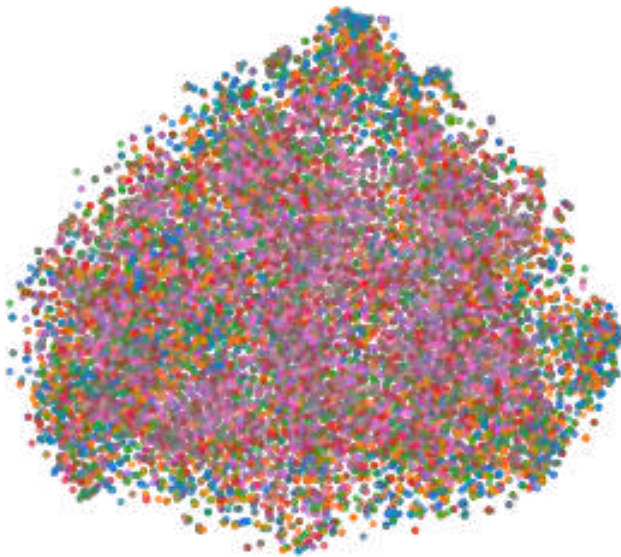
Satellite land use (EuroSAT)

Counting objects (CLEVR)

Linear probe

34.3%

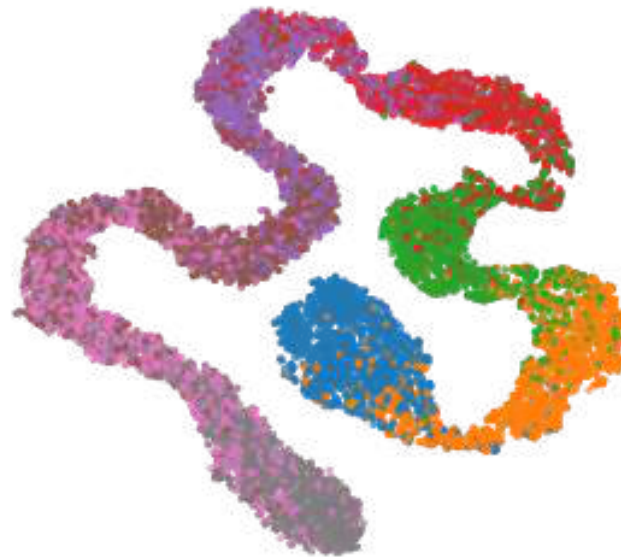
frozen network, no prompts



VPT-deep

68.5%

same frozen network, with prompts



Full fine-tuning

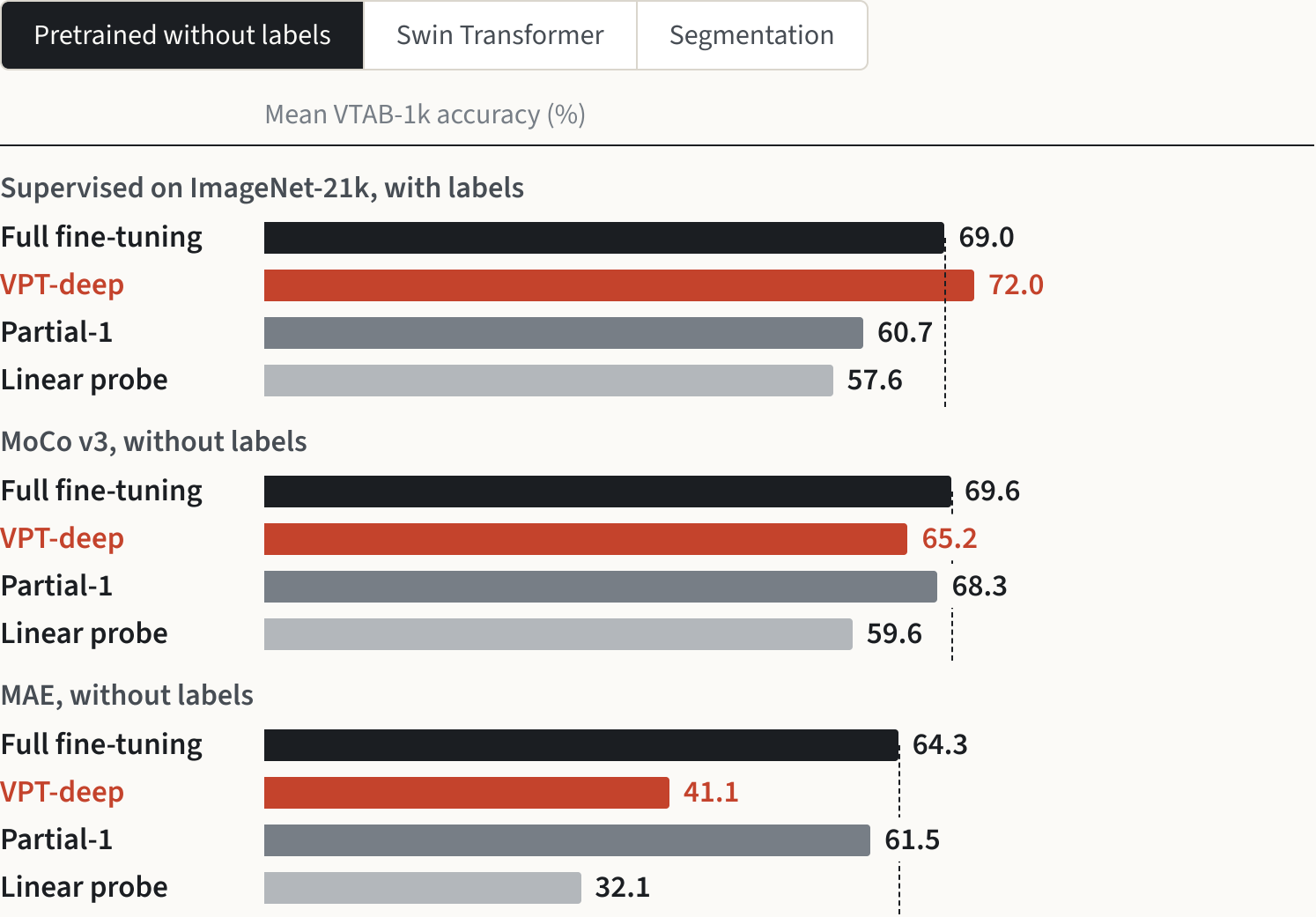
56.3%

every weight retrained



Each dot is one test image, coloured by its true class; the number is test accuracy. With the frozen features alone, the classes overlap. With prompts, the same frozen network produces clearly separated groups, so a simple linear head can tell them apart.

Where VPT falls short

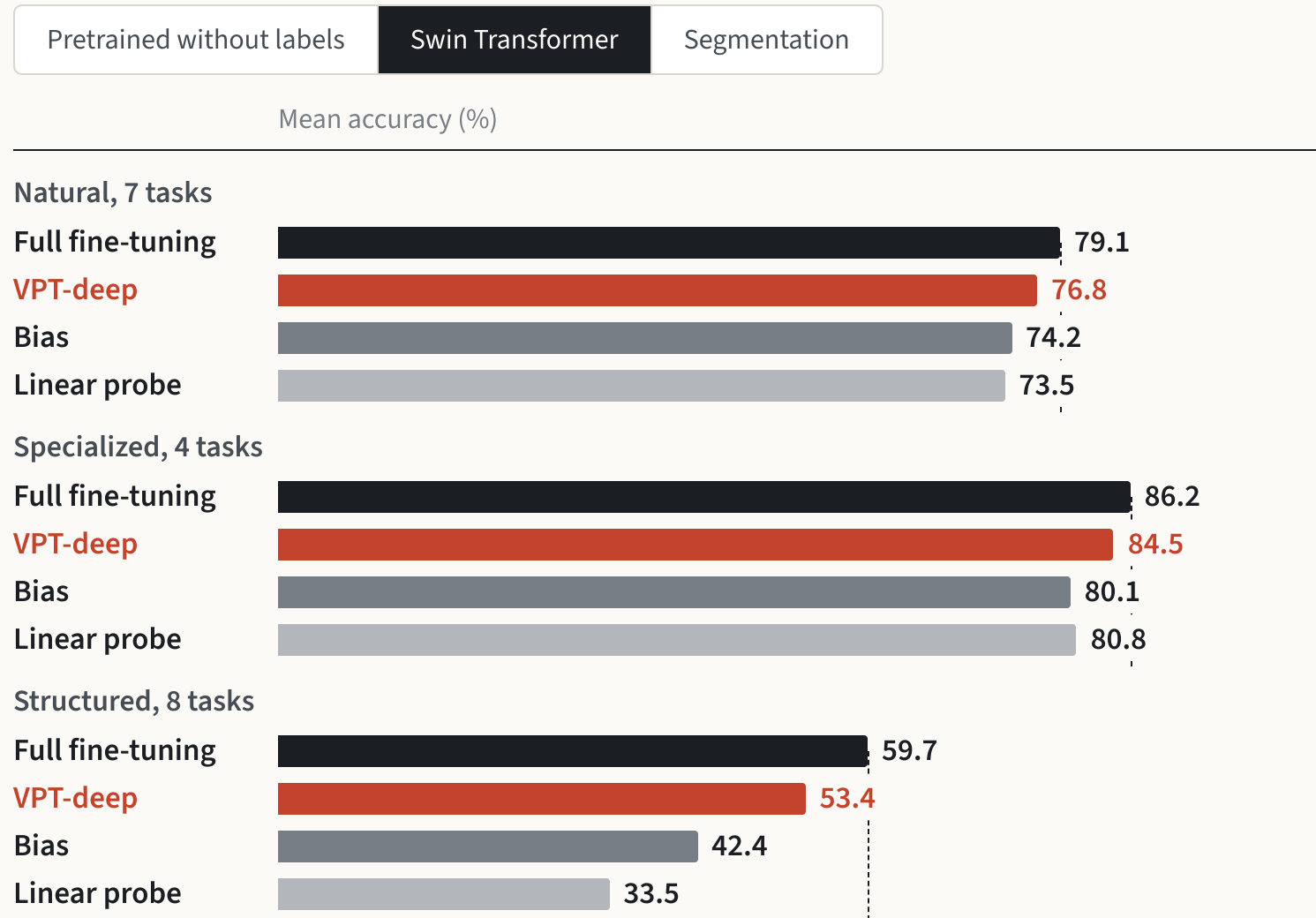


Everything so far used a ViT pretrained with labels. The paper also tries the same ViT-B pretrained without labels, and there VPT loses its lead.

With MoCo v3 it ends four points behind full fine-tuning. With MAE it drops towards the linear probe, while Partial-1, which retrain only the last block, stays close to full fine-tuning. The authors leave the reason open.

Jia et al. 2022, Table 1 (with labels) and Table 4 (MoCo v3, MAE): ViT-B/16, mean of the three VTAB-1k group averages.

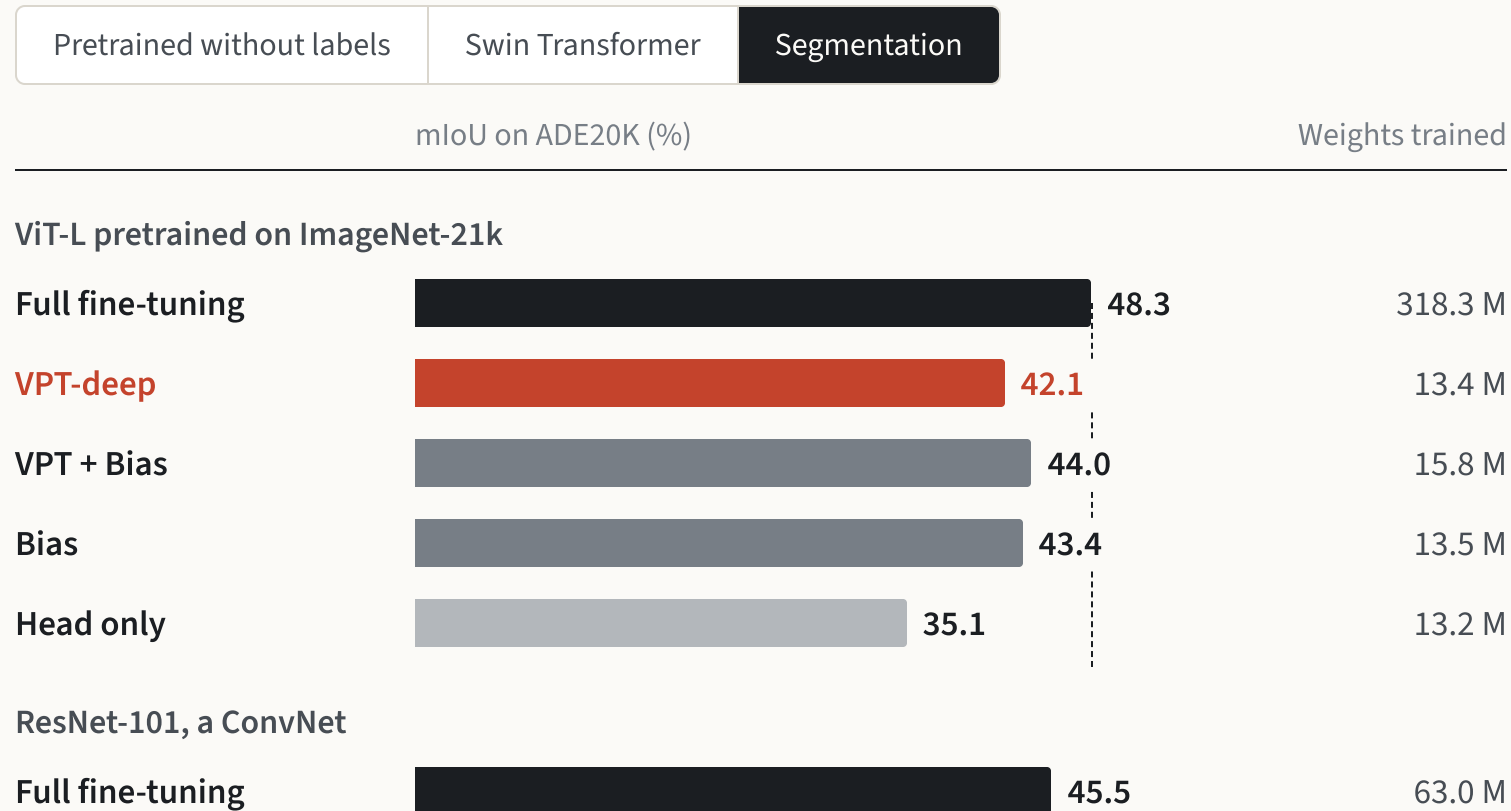
Where VPT falls short



Swin is a vision Transformer that looks at small windows of the image and merges patches as it goes deeper.

On Swin, VPT still beats the other cheap methods. But full fine-tuning is ahead of VPT-deep in all three groups, by 6.3 points on Structured.

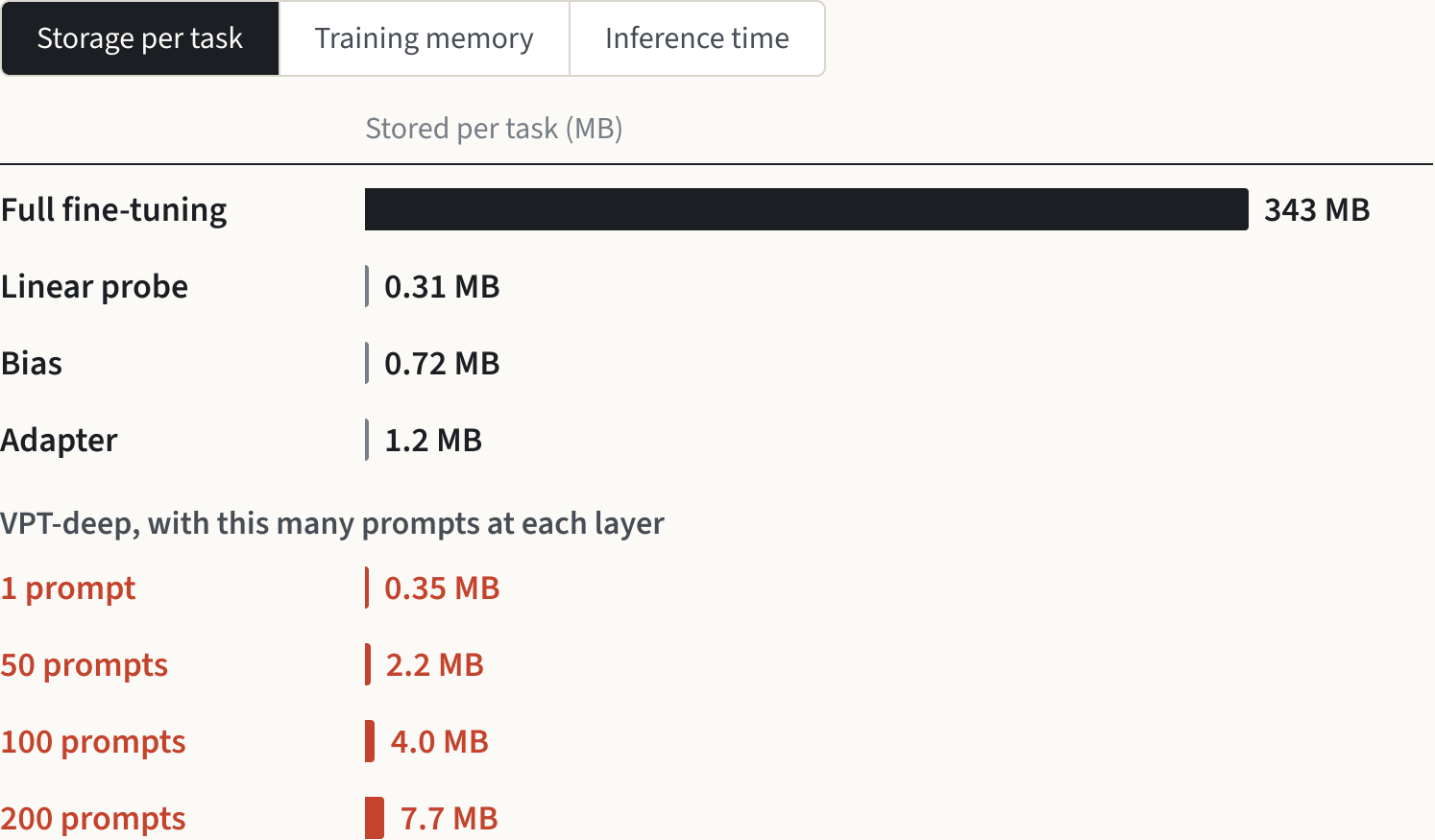
Where VPT falls short



Segmentation labels every pixel of a scene. The score, mIoU, measures how well the predicted regions overlap the true ones.

Here VPT-deep reaches 42.1 against 48.3 for full fine-tuning, about the same as tuning only the biases. Prompts and biases together come within 1.5 points of a fully fine-tuned ResNet-101.

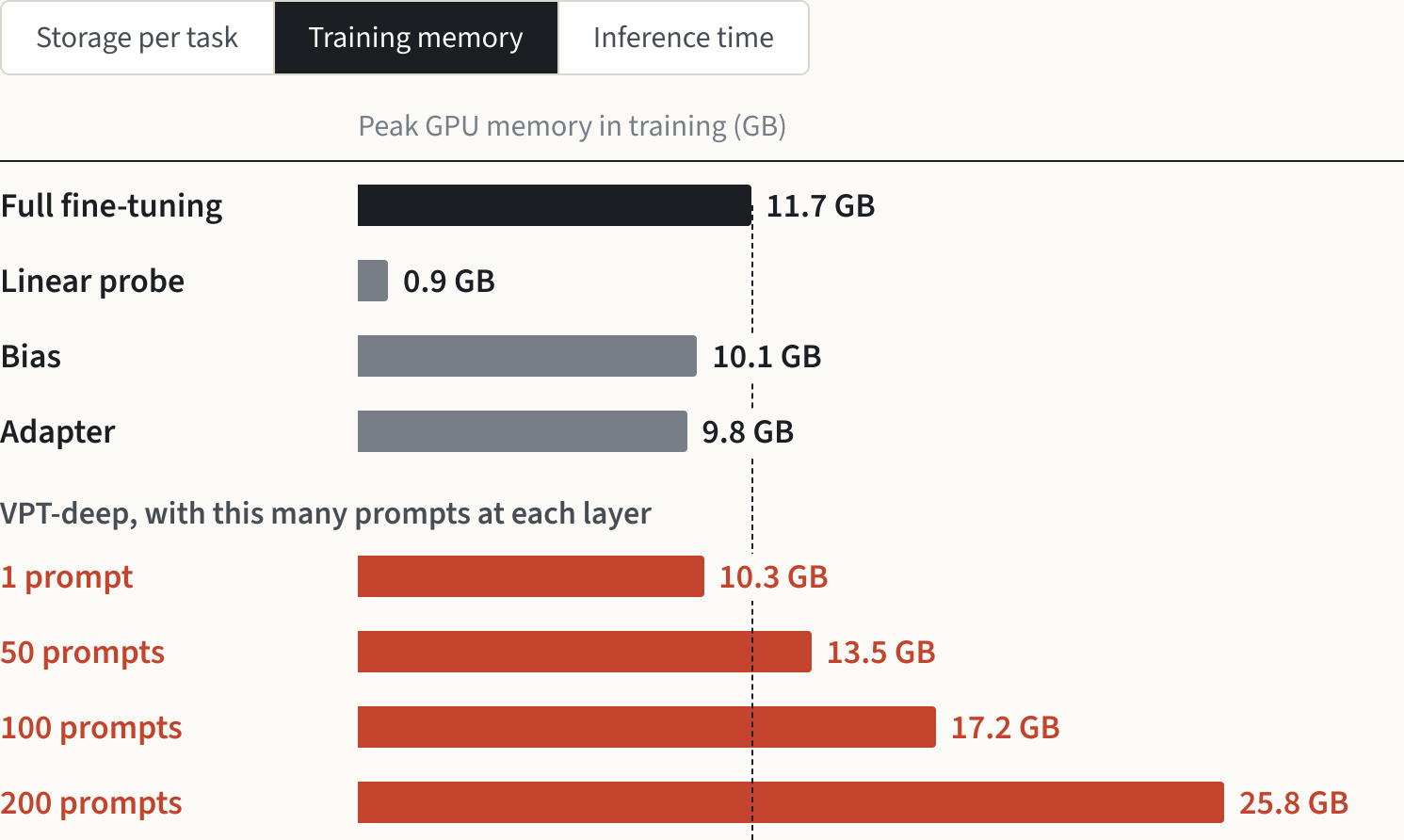
VPT saves storage, not training memory or inference time



A task trained with VPT-deep keeps only its prompts and its head: 0.35 MB with one prompt per layer, 7.7 MB with 200. A fully fine-tuned copy is 343 MB for every task.

All tasks share one frozen backbone, so this is the real saving, and it grows with every task.

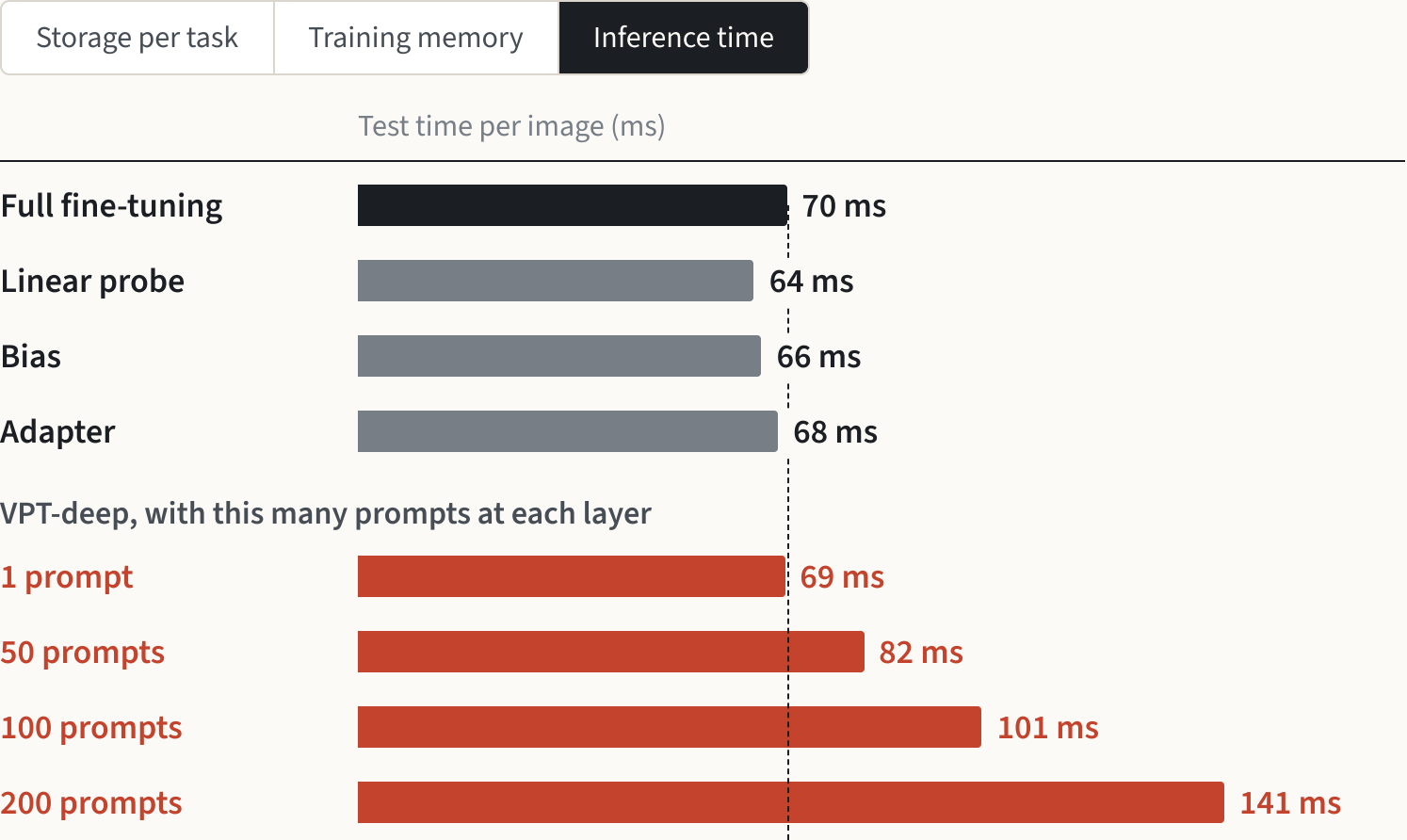
VPT saves storage, not training memory or inference time



In training, VPT-deep needs about as much GPU memory as full fine-tuning, and with 200 prompts more than twice as much.

To train the prompts in the first layer, the gradient has to travel back through all twelve frozen layers, so their activations must be kept, as in full fine-tuning. Only the linear probe avoids this.

VPT saves storage, not training memory or inference time



At test time every image carries the prompts through all twelve layers, so the sequence is longer. Fifty prompts make it 82 ms instead of 70, and 200 prompts double it.

The prompts cannot be folded into the weights, so this cost stays. LoRA, the next paper, avoids it.

My experiments

I re-implemented VPT and trained it on three medical imaging datasets from MedMNIST. A linear probe and full fine-tuning ran next to it, all three starting from the paper's ViT-B/16.

First the results, then all three models running live in this browser.



Skin lesions



Blood cells



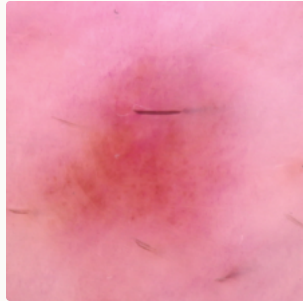
Chest X-rays

I tested VPT on three medical datasets

Each one was trained twice: on 1,000 images, as in the paper's VTAB-1k, and on its full training set.



melanoma

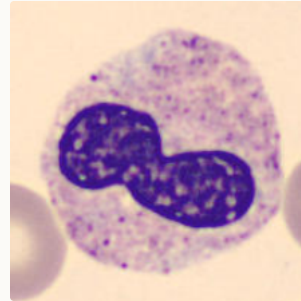


nevus

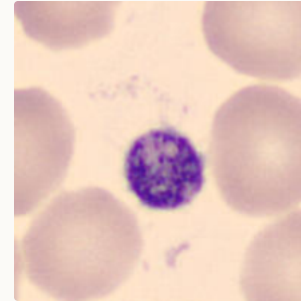
Skin lesions

Dermoscopy, 7 classes

 1,000 or all 7,007 training images



neutrophil



platelet

Blood cells

Microscopy, 8 cell types

 1,000 or all 11,959 training images



normal



pneumonia

Chest X-rays

Radiographs, pneumonia or normal

 1,000 or all 4,708 training images

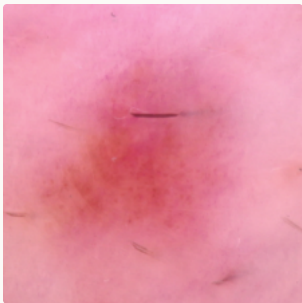
MedMNIST+ at 224×224 (Yang et al. 2023); skin lesions from HAM10000 (Tschandl et al. 2018). Trained on an RTX 5070 Ti laptop GPU with the paper's recipe: SGD for linear probe and VPT, AdamW for full fine-tuning, learning rate picked on validation.

I tested VPT on three medical datasets

Each one was trained twice: on 1,000 images, as in the paper's VTAB-1k, and on its full training set.



melanoma

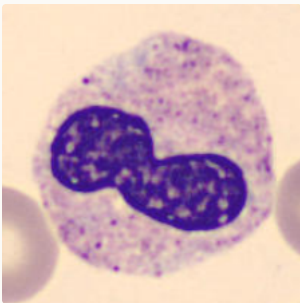


nevus

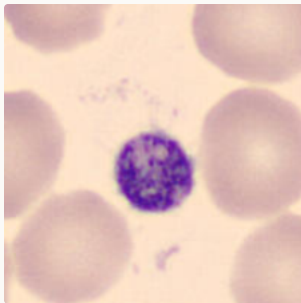
Skin lesions

Dermoscopy, 7 classes

1,000 or all 7,007 training images



neutrophil



platelet

Blood cells

Microscopy, 8 cell types

1,000 or all 11,959 training images



normal

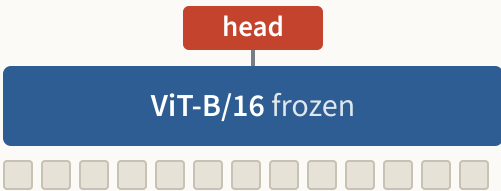


pneumonia

Chest X-rays

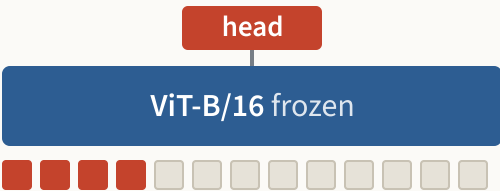
Radiographs, pneumonia or normal

1,000 or all 4,708 training images



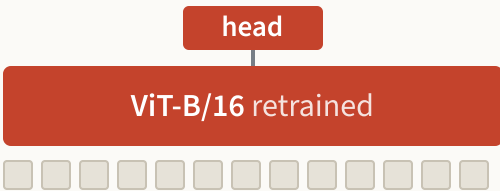
Linear probe

Trains only a new head.



VPT-deep

Trains a head and 50 prompts per layer.

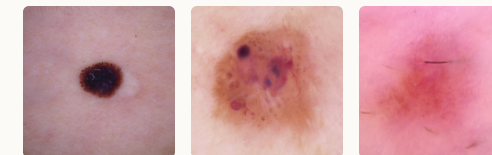
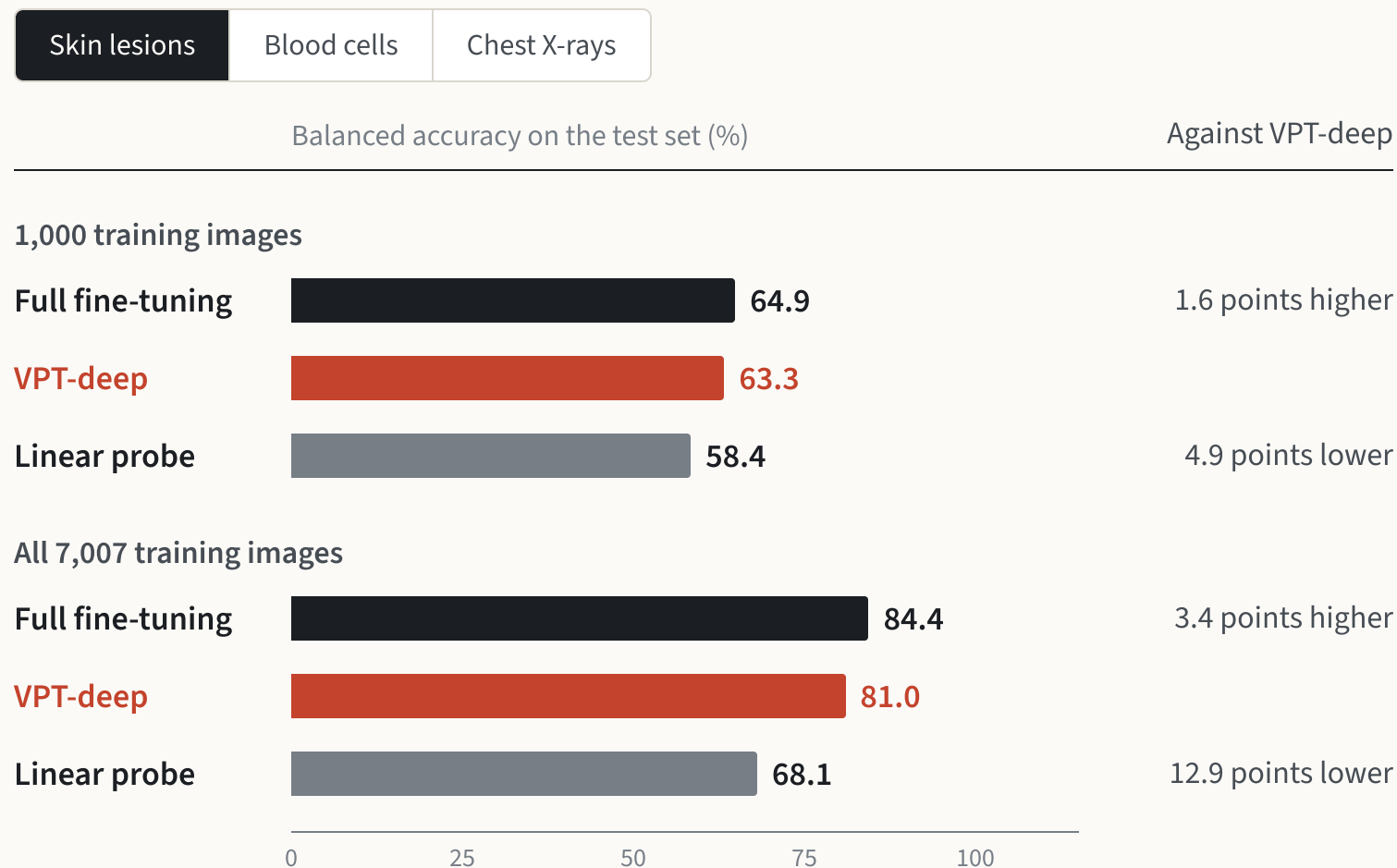


Full fine-tuning

Retrains all 85.8 million weights.

MedMNIST+ at 224×224 (Yang et al. 2023); skin lesions from HAM10000 (Tschandl et al. 2018). Trained on an RTX 5070 Ti laptop GPU with the paper's recipe: SGD for linear probe and VPT, AdamW for full fine-tuning, learning rate picked on validation.

VPT always beat the linear probe and came close to full fine-tuning

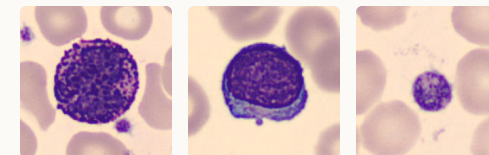
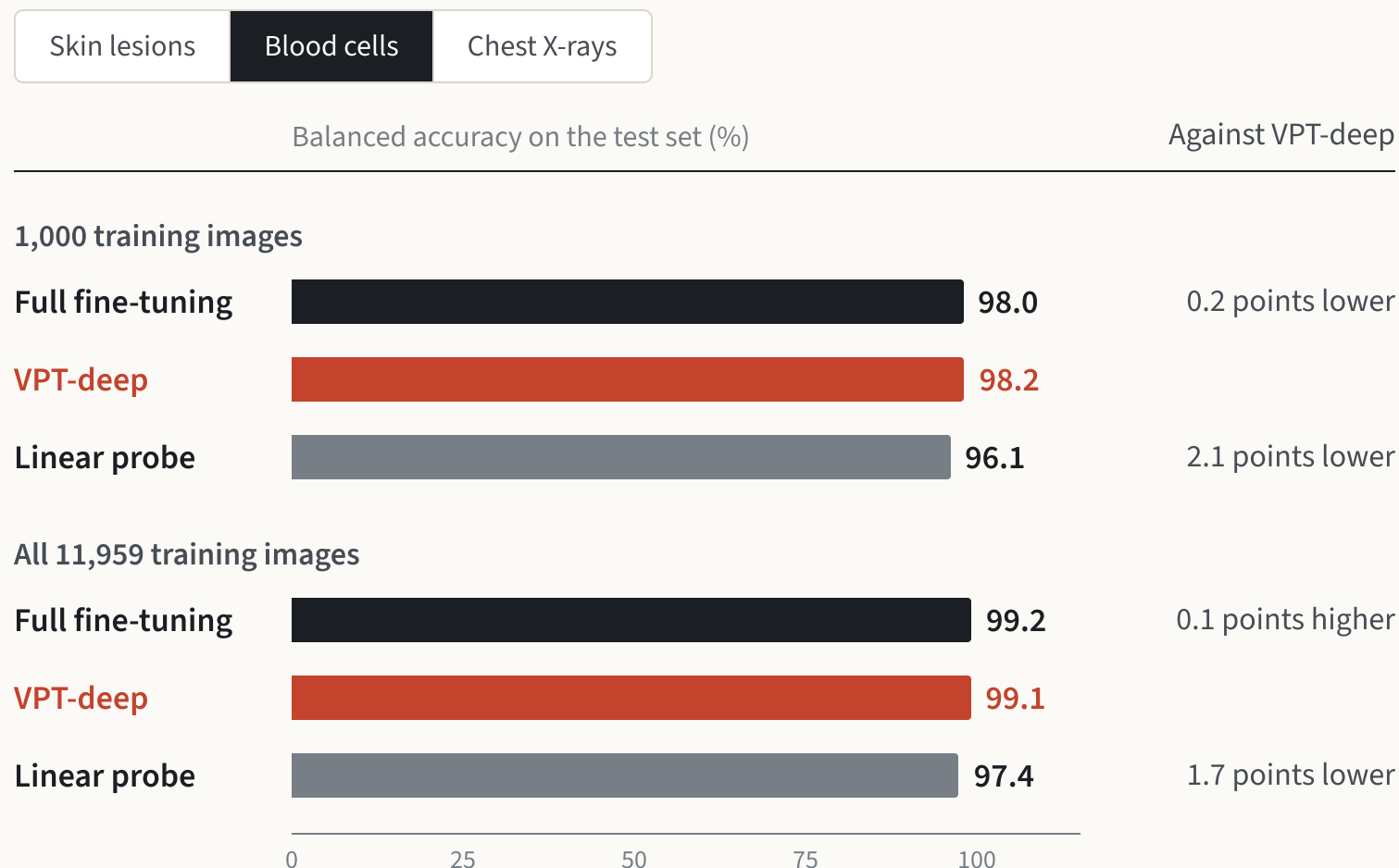


Seven kinds of skin lesion, and two thirds of the images show harmless moles. This is the hardest of the three tasks.

With all 7,007 images, VPT-deep is 12.9 points ahead of the linear probe, up from 4.9. Full fine-tuning stays ahead of VPT-deep, by 1.6 and 3.4 points.

My runs: ViT-B/16 (ImageNet-21k), MedMNIST+ 224 × 224, balanced accuracy on the test set (the mean per-class recall: on skin lesions, always answering "nevus" scores 67% accuracy but only 14% balanced accuracy), bars start at 0. Each method was trained once with 1,000 images and once with the full training set; there are no points in between.

VPT always beat the linear probe and came close to full fine-tuning

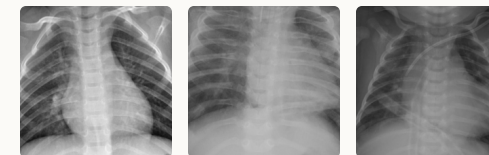
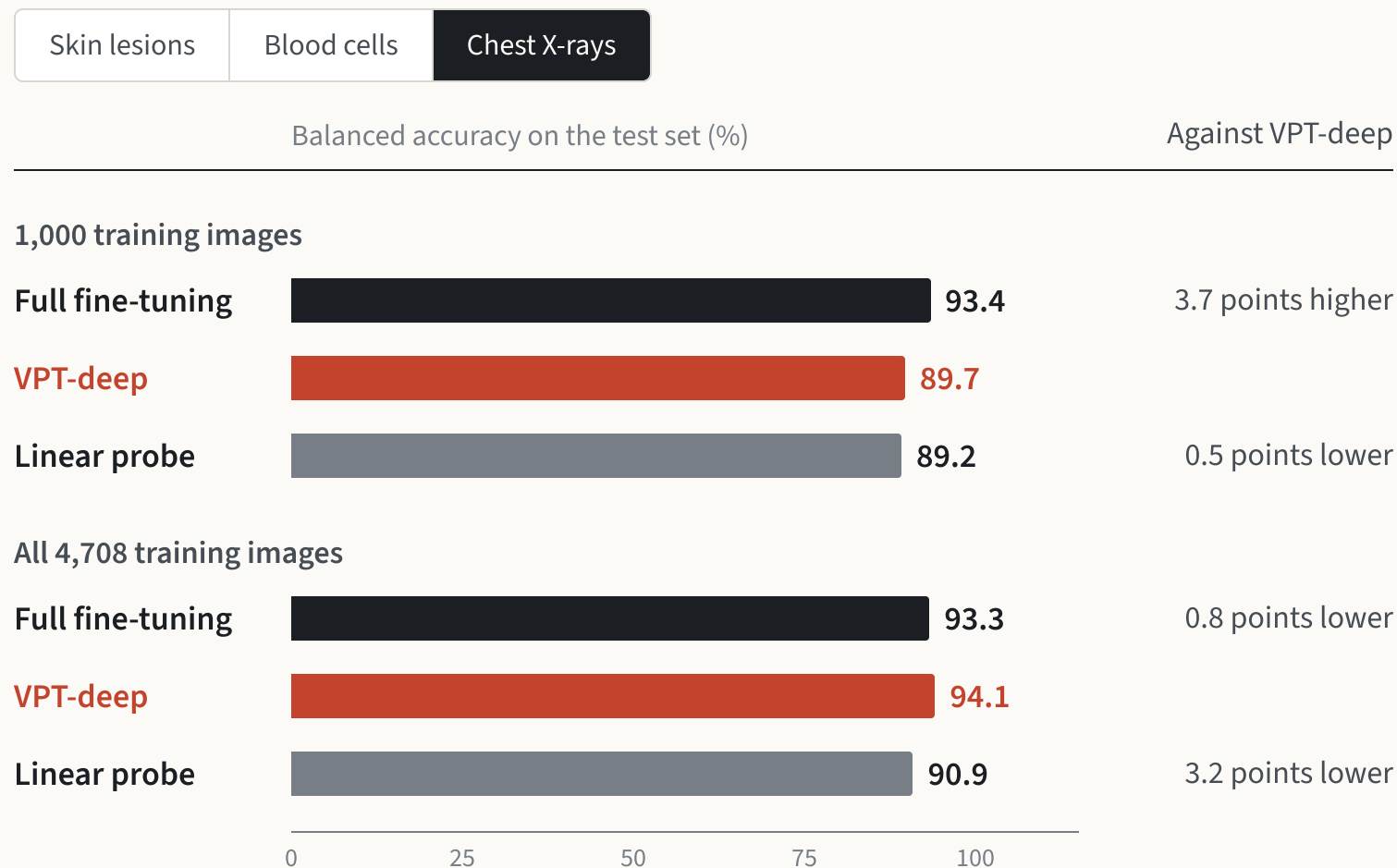


Eight kinds of blood cell under a microscope. Every method scores above 96 here, so the bars are almost the same length.

VPT-deep and full fine-tuning are within 0.2 points of each other with either amount of data.

My runs: ViT-B/16 (ImageNet-21k), MedMNIST+ 224 × 224, balanced accuracy on the test set (the mean per-class recall: on skin lesions, always answering "nevus" scores 67% accuracy but only 14% balanced accuracy), bars start at 0. Each method was trained once with 1,000 images and once with the full training set; there are no points in between.

VPT always beat the linear probe and came close to full fine-tuning



Chest X-rays of children, labelled normal or pneumonia.

With 1,000 images, VPT-deep is only 0.5 points above the linear probe and 3.7 points behind full fine-tuning. With all 4,708 images it scores highest of the three.

My runs: ViT-B/16 (ImageNet-21k), MedMNIST+ 224 × 224, balanced accuracy on the test set (the mean per-class recall: on skin lesions, always answering "nevus" scores 67% accuracy but only 14% balanced accuracy), bars start at 0. Each method was trained once with 1,000 images and once with the full training set; there are no points in between.

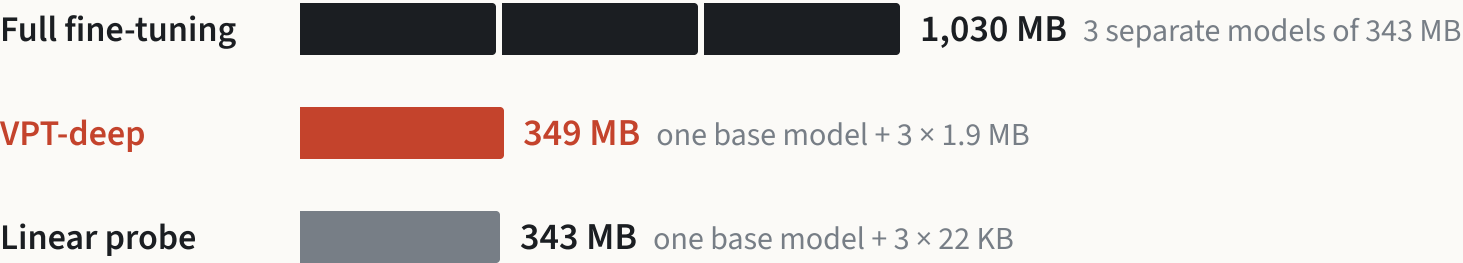
What each method costs on skin lesions

Storage and accuracy

Training

Inference

Stored for all three tasks



Balanced accuracy on the test set



VPT stores 1.9 MB per task instead of 343 MB, because every task shares one frozen copy of the base model. For three tasks that is 349 MB instead of 1,030 MB.

It reaches 81.0% balanced accuracy, a few points below full fine-tuning at 84.4%, and far above the linear probe at 68.1%.

What each method costs on skin lesions

Storage and accuracy

Training

Inference

Training time, 30 epochs on 7,007 images

Full fine-tuning 20 min

VPT-deep 14 min

Linear probe 5 min

Peak GPU memory during training, batch 32

Full fine-tuning 3.5 GB

VPT-deep 2.3 GB

Linear probe 0.5 GB

Training VPT took 14 minutes and 2.3 GB of GPU memory, against 20 minutes and 3.5 GB for full fine-tuning.

Only the linear probe is really cheap. To reach the prompts, the error signal still has to pass back through all twelve frozen layers, the same pattern as the paper's Table 12.

What each method costs on skin lesions

Storage and accuracy

Training

Inference

Time per image on the CPU, 2 threads



Tokens that every layer processes



On the CPU, VPT needs 227 ms per image, 19% more than the fine-tuned model.

Every layer has to process 50 extra tokens, and the prompts cannot be merged into the weights after training. The other two run the unchanged network, so they take about the same time.

Live demo: one image, three models

Skin lesion, one test image, models trained on all training images.



True label: **melanoma**

Linear probe

shared frozen model + head

68.5% test accuracy

melanocytic nevi	73.7%
melanoma	23.8%
benign keratosis-like	2.1%

Wrong

139 ms per image

VPT-deep

shared frozen model + prompts

81.0% test accuracy

melanoma	99.5%
benign keratosis-like	0.3%
melanocytic nevi	0.1%

Correct

173 ms per image

Full fine-tuning

its own retrained model

84.4% test accuracy

melanoma	97.7%
melanocytic nevi	2.2%
benign keratosis-like	0.0%

Correct

141 ms per image

Test accuracy is balanced accuracy on the whole test set, for exactly these models. Changing the training set swaps one 1.9 MB file for VPT-deep, but a whole second 172 MB model for full fine-tuning.

Live demo: one image, three models

Skin lesion, one test image, models trained on 1,000 training images.



True label: **melanoma**

Linear probe

shared frozen model + head

58.1% test accuracy

melanocytic nevi	84.5%
benign keratosis-like	7.9%
melanoma	7.5%

Wrong

174 ms per image

VPT-deep

shared frozen model + prompts

62.5% test accuracy

melanoma	69.2%
melanocytic nevi	30.8%
dermatofibroma	0.0%

Correct

206 ms per image

Full fine-tuning

its own retrained model

64.8% test accuracy

melanocytic nevi	95.1%
melanoma	4.2%
benign keratosis-like	0.2%

Wrong

145 ms per image

Test accuracy is balanced accuracy on the whole test set, for exactly these models. Changing the training set swaps one 1.9 MB file for VPT-deep, but a whole second 172 MB model for full fine-tuning.

Summary

0.6%

of the weights are trained: a few vectors among the patch tokens and a small head, while the Transformer stays frozen.

20 of 24

tasks where VPT-deep beat full fine-tuning, on a ViT-B pretrained with labels.

99.4%

less storage per task than a fully fine-tuned copy: 2.2 MB instead of 343 MB. Training memory and inference time are not saved.

87.6%

average balanced accuracy of VPT-deep on my three medical datasets, against 88.8% for full fine-tuning and 83.3% for the linear probe (both training-set sizes).

References

This paper

Jia, Tang, Chen, Cardie, Belongie, Hariharan & Lim 2022. *Visual prompt tuning*. ECCV. arXiv:2203.12119.

Prompting before VPT

Brown et al. 2020. *Language models are few-shot learners* (GPT-3). NeurIPS.

Lester, Al-Rfou & Constant 2021. *The power of scale for parameter-efficient prompt tuning*. EMNLP.

Li & Liang 2021. *Prefix-tuning: optimizing continuous prompts for generation*. ACL.

Zhou et al. 2021. *Learning to prompt for vision-language models* (CoOp). arXiv:2109.01134; IJCV 2022.

Bahng et al. 2022. *Exploring visual prompts for adapting large-scale models*. arXiv:2203.17274.

Parameter-efficient methods

Houlsby et al. 2019. *Parameter-efficient transfer learning for NLP* (Adapter). ICML.

Ben Zaken, Goldberg & Ravfogel 2022. *BitFit: simple parameter-efficient fine-tuning for Transformer-based masked language-models* (Bias). ACL.

Zhang et al. 2020. *Side-tuning: a baseline for network adaptation via additive side networks* (Sidetune). ECCV.

Hu et al. 2022. *LoRA: low-rank adaptation of large language models*. ICLR.

Models

Dosovitskiy et al. 2021. *An image is worth 16×16 words: Transformers for image recognition at scale* (ViT). ICLR.

Liu et al. 2021. *Swin Transformer: hierarchical vision Transformer using shifted windows*. ICCV.

Chen, Xie & He 2021. *An empirical study of training self-supervised vision Transformers* (MoCo v3). ICCV.

He et al. 2022. *Masked autoencoders are scalable vision learners* (MAE). CVPR.

Zheng et al. 2021. *Rethinking semantic segmentation from a sequence-to-sequence perspective with Transformers* (SETR). CVPR.

Chen et al. 2018. *Encoder-decoder with atrous separable convolution for semantic image segmentation* (DeepLab v3+). ECCV.

Datasets

Zhai et al. 2019. *A large-scale study of representation learning with the visual task adaptation benchmark* (VTAB). arXiv:1910.04867.

Zhou et al. 2019. *Semantic understanding of scenes through the ADE20K dataset*. IJCV.

Yang et al. 2023. *MedMNIST v2: a large-scale lightweight benchmark for 2D and 3D biomedical image classification*. Scientific Data.

Tschandl, Rosendahl & Kittler 2018. *The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions*. Scientific Data.