A Appendix

A.1 Author Contributions

PSS devised the project, led the team, developed the models, drafted the manuscript, and otherwise contributed to all parts of MindEye development. AB drafted the manuscript, developed the models, and contributed to all parts of MindEye development including creating the low-level pipeline, conception of BiMixCo and soft CLIP loss, and modification of the DALL-E 2 diffusion prior. JG developed the models, tracked/compared model variants, and significantly contributed to the MindEye codebase. SS conceived of and implemented LAION-5B retrieval using the CLIP Retrieval client and conducted various exploratory experiments. AN implemented the Lafite pipeline for MindEye reconstructions. EC conducted various initial explorations into using a diffusion prior for aligning voxels to CLIP space. AJD created the initial webdatasets used to train MindEye and created various model architectures to compare different mapping approaches. NV conducted various exploratory experiments mapping voxels to StyleGAN-XL [73] latent space. EY shared code to automatically identify identical images for qualitative comparisons and added code to ensure LAION-5B retrieval did not retrieve ground truth images. DW conducted various exploratory experiments and helped with project discussions. KAN oversaw the project and contributed valuable feedback. TMA oversaw the project, conducted initial explorations using VQGAN [74], and helped keep the project on-track through MedARC and Stability AI communication.

A.2 Additional Dataset Information

The Natural Scenes Dataset (NSD) [26] is a public 7-Tesla fMRI dataset containing the brain responses of several human participants each spending up to 40 hours in the MRI machine passively viewing images. These square-cropped images of natural scenes were sourced from the MS-COCO dataset [27]. Each of 9,000-10,000 unique images was presented for three seconds at a time, shown three times across 30-40 scanning sessions, totaling 22,000-30,000 trials of fMRI responses per participant. fMRI responses correspond to session-wise z-scored single-trial betas output from GLMSingle [75]. Following the procedure used in other reconstruction studies that used NSD [5, 28, 3], we train individual-subject models for the four participants who completed all scanning sessions (participants 1, 2, 5, and 7) and used a test set corresponding to the shared 1,000 images presented to every participant. This yields a dataset consisting of 24,980 training samples and 2,770 test samples—we average across the three same-image repetitions for the test set (leaving 982 test samples) but not the training set, similar to Takagi and Nishimoto [3]. We use preprocessed flattened fMRI voxels in 1.8-mm native volume space corresponding to the "nsdgeneral" brain region, defined by the NSD authors as the subset of voxels in posterior cortex most responsive to the visual stimuli presented (between 13,000 to 16,000 voxels per participant). MindEye was developed using a training and validation set of Subject 1's data, with the test set (and other subjects' data) untouched until final training of models.

A.3 MindEye Architecture

PyTorch code for the MLP backbone and projector is depicted in Algorithm 1. Specifics on how we modified the open-source implementation of the DALL-E 2 diffusion prior are discussed in A.3.1.

A.3.1 Modifications from DALL-E 2 Diffusion Prior

The inputs for the diffusion prior are 257 backbone embeddings, 1 timestep embedding, and 257 noised CLIP image embeddings, and the output is 257 denoised CLIP image embeddings. Unlike the DALL-E 2 prior, we do not use learnable queries and instead directly predict denoised CLIP embeddings from the noised embeddings. This significantly saves on memory and allows us to train the backbone and prior end-to-end on a single GPU. We observe that adding absolute positional embeddings to the noised CLIP embeddings improves performance in the absence of learnable queries. We also observe that our prior can work with just 100 timesteps instead of 1000 as used in DALL-E 2. This makes our prior much faster at inference time. We conducted experiments with both causal and bidirectional attention and did not observe any significant difference in reconstruction performance. For simplicity we use bidirectional attention in our final model.

Algorithm 1 PyTorch code for MindEye MLP backbone and MLP projector

```
class BrainMLP(nn.Module):
    def __init__(self, out_dim=257*768, in_dim=15724, clip_size=768, h=4096):
        super().__init__()
        # in_dim corresponds to the subject-specific
        # number of voxels in the "nsdgeneral" brain region.
        self.lin0 = nn.Sequential(
            nn.Linear(in_dim, h, bias=False),
            nn.LayerNorm(h),
            nn.GELU(inplace=True),
            nn.Dropout(0.5))
        self.mlp = nn.ModuleList([
            nn.Sequential(
                nn.Linear(h, h, bias=False),
                nn.LayerNorm(h),
                nn.GELU(inplace=True),
                nn.Dropout(0.15)
            ) for _ in range(4)])
        self.lin1 = nn.Linear(h, out_dim, bias=True)
        self.proj = nn.Sequential(
            nn.LayerNorm(clip_size),
            nn.GELU(inplace=True),
            nn.Linear(clip_size, 2048, bias=False),
            nn.LayerNorm(2048),
            nn.GELU(inplace=True),
            nn.Linear(clip_size, 2048, bias=False),
            nn.LayerNorm(2048),
            nn.GELU(inplace=True),
            nn.Linear(2048, clip_size, bias=True))
        self.clip_size = clip_size

    def forward(self, x):
        x = self.lin0(x)
        residual = x
        for res_block in range(len(self.mlp)):
            x = self.mlp[res_block](x)
            x += residual
            residual = x
        diffusion_prior_input = self.lin1(x).reshape(len(x), -1, self.clip_size)
        disjointed_clip_fmri = self.proj(diffusion_prior_input)

        return diffusion_prior_input, disjointed_clip_fmri
```

A.3.2 Low-Level Pipeline: Mapping to Stable Diffusion Variational Autoencoder

To map to Stable Diffusion's VAE latent space we use a low-level pipeline with the same architecture as the high level pipeline. We use a separate residual MLP backbone with 4 residual blocks that maps flattened voxels to a $16 \times 16 \times 64$ dimensional latent space. The reconstruction submodule in the low-level pipeline is a CNN upsampler that upsamples these latents by $4\times$ to create embeddings of size $(64, 64, 4)$. The CNN upsampler uses a similar architecture to Stable Diffusion's VAE decoder, which does an $8\times$ upsampling. To create the targets for the upsampler we upsample NSD images to 512×512 through bilinear interpolation and encode them with the SD VAE encoder. The resulting $(64, 64, 4)$ embeddings form the targets for the high-level pipeline.

Recent works in low-level vision (super-resolution, denoising, deblurring, etc.) have observed that mean absolute error performs better than mean squared error for pixel-level metrics like PSNR and SSIM [76, 77] due to better convergence properties. It has been shown that the 4-channel SD latent space effectively compresses images, and latents can be converted to RGB images with a linear mapping from latent space to pixel space [78]. We observe that the problem of mapping to SD embedding space follows the same properties as low-level vision tasks, such that mean absolute error performs better than mean squared error. We also experiment with using a "full reconstruction" loss where we reconstruct complete images using the SD VAE decoder and apply the loss in pixel space. This performs worse than only applying the loss in latent space and also requires significantly more GPU memory.

The contrastive submodule in the low-level pipeline acts as an auxiliary loss to improve the performance of the reconstruction submodule. It uses an MLP projector that maps the $(16, 16, 64)$ backbone outputs to $(16, 16, 512)$. Since we do not care about retrieval performance for the low-level pipeline, we simply use SoftCLIP loss without BiMixCo. To maximize low-level performance we distill the knowledge of VICRegL [79] ConvNext-XXL instead of CLIP ViT. VICRegL with $\alpha = 0.75$ is specialized for low-level tasks and achieves state-of-the-art linear segmentation results, unlike CLIP which has been trained with high-level text guidance.

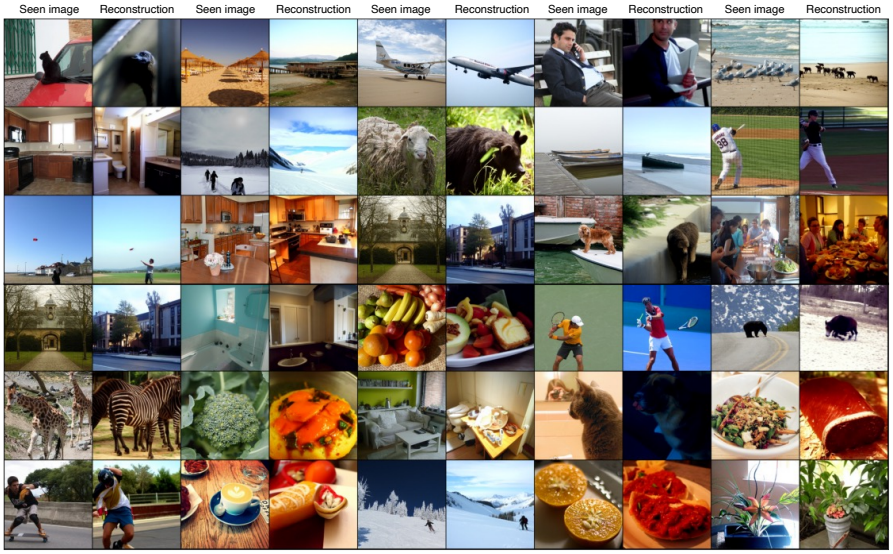

Figure 5: Additional MindEye reconstructions for Subject 1, randomly selected.

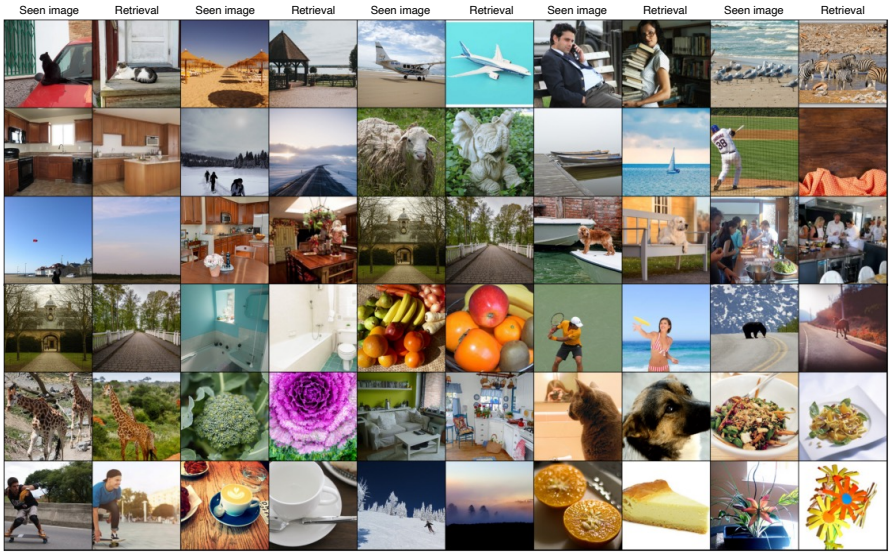

Figure 6: Additional MindEye retrievals from LAION-5B for Subject 1, randomly selected.

A.4 More Reconstructions / Retrievals

Images containing all 982 reconstructions and retrievals for each subject are on GitHub. Figure 5 depicts a subset of randomly selected reconstruction examples from Subject 1 (first try random selection of 30 samples). Figure 6 likewise depicts randomly selected examples from LAION-5B retrieval. Figure 7 depicts randomly selected example reconstructions from the low-level pipeline. Figure 8 depicts randomly selected reconstructions for the models described in 3.3.

A.5 UMAP Comparison

As depicted in Figure 9, the CLIP image and CLIP fMRI embedding spaces are disjointed before being fed through the diffusion prior. While the MLP projector does improve alignment compared to the outputs of the MLP backbone, the diffusion prior does a much better job at aligning the two spaces as shown by decreased euclidean distance between data points following UMAP dimensionality reduction.

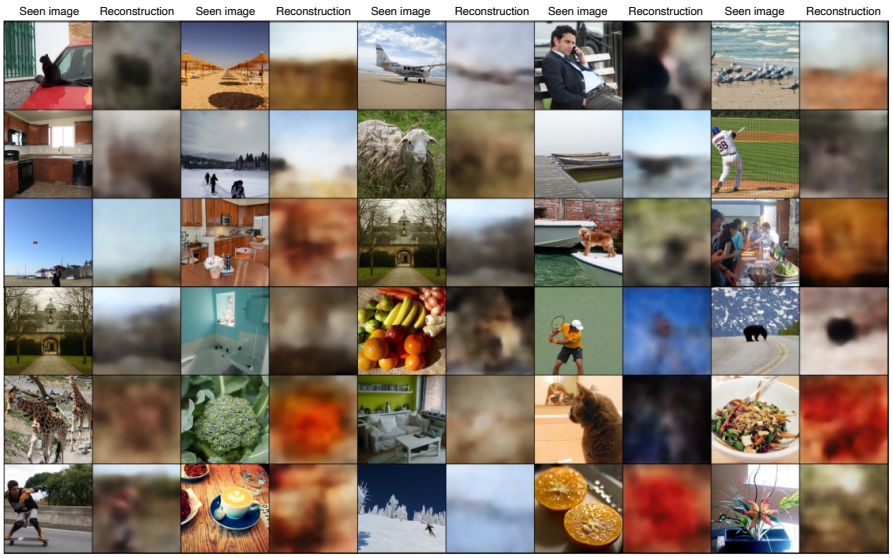

Figure 7: Example MindEye reconstructions for Subject 1 output from the low-level pipeline.

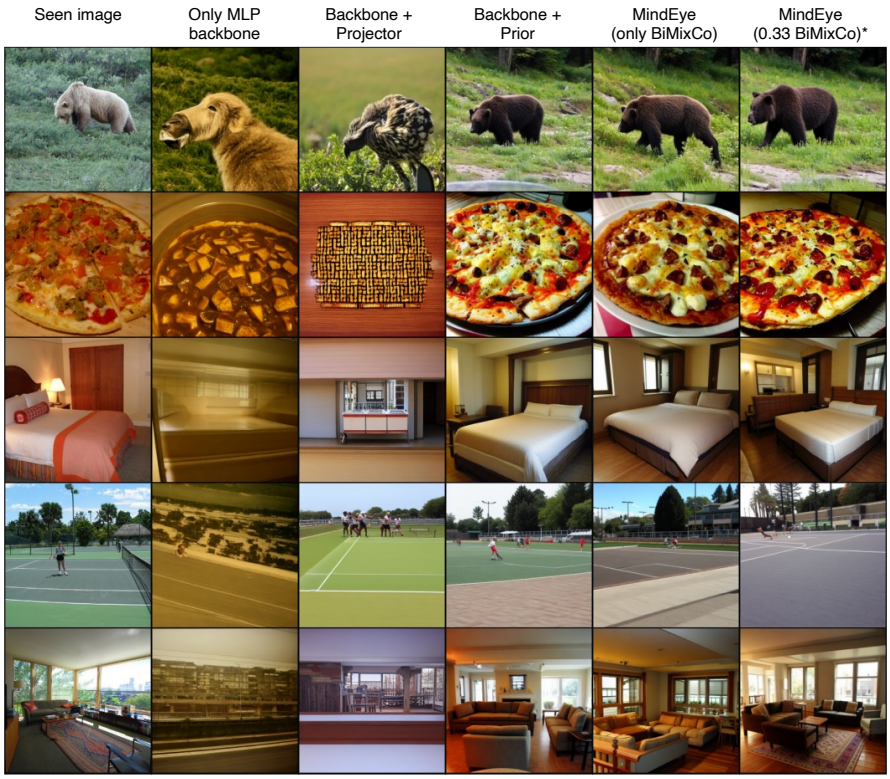

Figure 8: Example reconstructions for ablation models from Table 4

Img2Img Strength	Low-Level				High-Level			
	PixCorr ↑	SSIM ↑	Alex(2) ↑	Alex(5) ↑	Incep ↑	CLIP ↑	Eff ↓	SwAV ↓
1.0 (Only low-level)	**.456**	**.493**	87.1%	84.1%	61.6%	62.4%	.992	.638
0.7	.439	.416	92.7%	95.1%	90.0%	87.5%	.803	.514
0.5	.429	.389	96.3%	98.4%	**94.7%**	92.3%	.674	.405
0.3	.410	.358	**97.5%**	**98.8%**	**94.7%**	94.5%	.638	.362
0.15*	.390	.337	97.4%	98.7%	94.5%	94.6%	**.630**	**.358**
0.0 (Only high-level)	.209	.318	92.8%	98.0%	94.5%	**94.8%**	.635	.361

Table 5: Evaluations from Subject 1 varying img2img strength from 0 (no img2img) to 1 (only low-level pipeline). The final MindEye uses an img2img strength of 0.15.

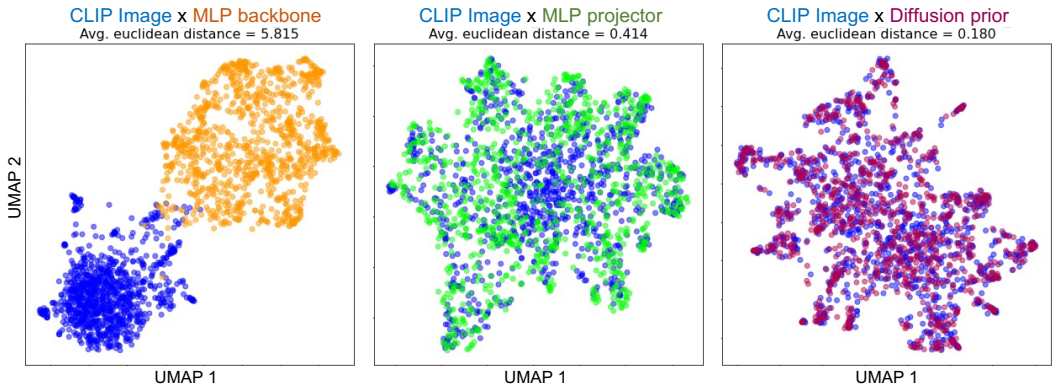

Figure 9: UMAP plots depict CLIP image latents (blue), MindEye MLP backbone latents (orange), MindEye MLP projector latents (green), and MindEye diffusion prior latents (red). UMAPs were estimated from 1,000 random samples from Subject 1. CLIP image latents correspond to the last hidden layer of ViT-L/14. Euclidean distance between the given MindEye embedding space and CLIP image space is lowest for the diffusion prior, suggesting that the diffusion prior helps to align the two embedding spaces.

A.6 Reconstruction Evaluations: Additional Information

Two-way identification was performed in the same manner as Ozcelik and VanRullen [4]. For each model, we computed the Pearson correlation between embeddings for the ground truth image and the reconstructed image, as well as the correlation between the ground truth image and a different reconstruction elsewhere in the test set. If the correlation for the former was higher than the latter, this was marked as correct. For each test sample, performance was averaged across all possible pairwise comparisons using the other 981 reconstructions to ensure no bias from random sample selection. This yielded 982 averaged percent correct outputs, which we averaged across to obtain the metrics reported in Table 1.

Retrieval evaluations for Ozcelik and VanRullen [4] were not reported in the original paper; we calculated image/brain retrieval ourselves with the help of the Brain-Diffuser GitHub repository.

A.7 Reconstructions from Stable Diffusion (Image Variations) and Lafite

We also attempted reconstructions using Stable Diffusion (Image Variations) [31] and Lafite [32] rather than Versatile Diffusion. Reconstructions from these models for Subject 1 are depicted in Figure 10, with metrics reported in Table 6.

For Stable Diffusion (Image Variations) we use the same approach as MindEye + Versatile Diffusion except we map from voxels to the 1×768 final layer outputs of ViT-L/14 (same architecture as "4 ResBlocks + Only CLS" in Table 2). For the diffusion prior we fine-tune an open-sourced implementation of the DALL-E 2 prior that was trained to generate CLIP image embeddings from CLIP text embeddings using 250M image-caption pairs from LAION-Aesthetics. We note that using this pretrained prior works much better than training from scratch, suggesting that a similar large-scale pretrained prior for Versatile Diffusion might further improve fMRI reconstructions. We

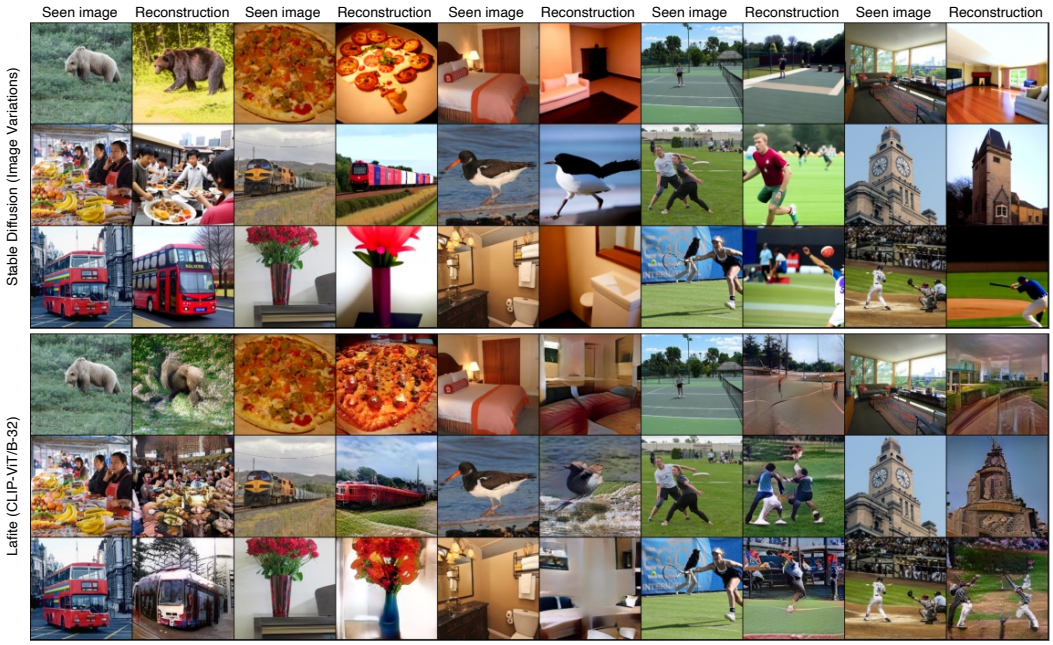

Figure 10: Corresponding reconstructions to Figure 1 when swapping Versatile Diffusion with Stable Diffusion (Image Variations) or Lafite.

use this MindEye model variant both for reconstructing via Stable Diffusion (Image Variations) and for retrieving the top-16 nearest neighbors in CLIP space for LAION-5B image retrieval. This is because the CLIP Retrieval client [30] only has precomputed CLIP embeddings for the final layer of CLIP, not the last hidden layer as used by Versatile Diffusion.

For Lafite we tried to replicate the same approach as Lin et al. [11] but with inputs from the MindEye MLP backbone. Lafite is a conditional image generation pipeline that uses the CLIP-aligned voxel embeddings as "condition vectors". In particular, Lafite leverages a StyleGAN that uses the CLIP embeddings as "style" vectors to generate images. Lafite's discriminator is trained to distinguish generated images from ground truth images and also to semantically align the CLIP embedding of the generated image with the condition vector using contrastive learning. Here we train two mapping models f_{mi} and f_{mc} that map voxels to the final layer of CLIP ViT-B/32, where f_{mi} is contrastively aligned with CLIP image embeddings and f_{mc} is contrastively aligned with CLIP text embeddings. We used the same contrastive learning schedule as MindEye with BiMixCo for the first one-third of the training cycle and SoftCLIP for the rest. Note that Lafite doesn't require training a prior so we only train the MLP backbone. Once the mapping models f_{mi} and f_{mc} are trained, we follow Lin et al. [11] to fine-tune a pretrained language-free Lafite model provided by [32]. Finally, we use a low-level "perceptual" pipeline by aligning layer-2 ResNet features of the generated image with those of the ground truth image using contrastive learning. The ResNet was trained using a self-supervised VICReg loss [80].

Method	Low-Level				High-Level			
	PixCorr ↑	SSIM ↑	Alex(2) ↑	Alex(5) ↑	Incep ↑	CLIP ↑	Eff ↓	SwAV ↓
Versatile Diffusion (S1)	**.390**	.337	**97.4%**	**98.7%**	**94.5%**	**94.6%**	**.630**	**.358**
SD Image Variations (S1)	.376	**.350**	95.7%	96.4%	92.5%	92.5%	.734	.446
Lafite (S1)	.241	.304	92.5%	98.1%	93.7%	87.0%	.701	.436

Table 6: Evaluations for Subject 1 across three pretrained final image generation models (Lafite was fine-tuned in the same manner as Lin et al. [11]). Both Versatile Diffusion and Stable Diffusion (image variations) used an img2img strength of .15 with the low-level reconstructions output from MindEye (Lafite is a GAN and not compatible with the same img2img process).

Figure 11: Corresponding reconstructions to Figure 1 for the other 3 NSD participants.

A.8 Subject-Specific Results

Here we depict reconstructions across the other 3 NSD participants in Figure 11, with individual subject evaluations metrics in Table 7.

Method	Low-Level				High-Level				Retrieval	
	PixCorr ↑	SSIM ↑	Alex(2) ↑	Alex(5) ↑	Incep ↑	CLIP ↑	Eff ↓	SwAV ↓	Image ↑	Brain ↑
MindEye (Subj 1)	.390	.337	97.4%	98.7%	94.5%	94.6%	.630	.358	97.2%	94.7%
MindEye (Subj 2)	.318	.327	95.8%	98.1%	93.2%	93.7%	.656	.368	97.1%	93.9%
MindEye (Subj 3)	.265	.311	93.2%	97.8%	94.9%	94.9%	.628	.353	90.7%	85.7%
MindEye (Subj 4)	.261	.316	92.3%	96.6%	92.4%	93.0%	.666	.387	89.4%	85.9%

Table 7: MindEye retrieval and reconstruction performance for individual participants. These scores were averaged across participants for the values shown in Table 1.

A.9 Single-Trial Results

In the main paper we report results from the test dataset following the standard approach of averaging voxels across the three same-image repetitions. Reconstruction evaluations using only one brain sample for each image is shown in Table 8, with example reconstructions in Figure 12.

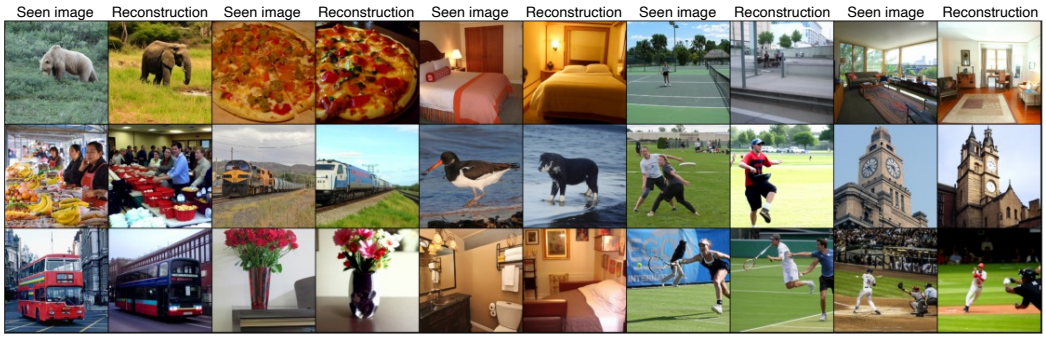

Figure 12: Corresponding reconstructions to Figure 1 using brain activity from only the first sample of every image. This is in contrast to Figure 1 which reconstructed from brain activity averaged across three same-image repetitions.

Method	Low-Level				High-Level				Retrieval	
	PixCorr ↑	SSIM ↑	Alex(2) ↑	Alex(5) ↑	Incep ↑	CLIP ↑	Eff ↓	SwAV ↓	Image ↑	Brain ↑
MindEye	.309	.323	94.7%	97.8%	93.8%	94.1%	.645	.367	93.6%	90.1%
MindEye (single-trial)	.255	.308	91.6%	95.9%	91.3%	91.6%	.691	.398	80.3%	77.6%
MindEye (single-, S1)	.329	.323	94.8%	97.3%	92.8%	92.7%	.680	.387	89.0%	86.5%
MindEye (single-, S2)	.267	.311	93.1%	96.9%	91.5%	91.2%	.687	.398	88.5%	86.1%
MindEye (single-, S3)	.217	.297	90.2%	96.3%	93.2%	94.0%	.671	.381	75.3%	71.8%
MindEye (single-, S4)	.209	.302	88.3%	93.1%	87.6%	88.7%	.727	.427	68.5%	66.1%

Table 8: MindEye retrieval and reconstruction performance for single-trial brain activations, chosen randomly out of three possible samples per unique image. Other than using single-trial brain activity, the same settings were used as in Table 1.

A.10 Performance with varying dataset size

Method	Low-Level				High-Level				Retrieval	
	PixCorr ↑	SSIM ↑	Alex(2) ↑	Alex(5) ↑	Incep ↑	CLIP ↑	Eff ↓	SwAV ↓	Image ↑	Brain ↑
All Data (High-Level)	.209	.318	92.8%	98.0%	94.5%	94.8%	.635	.361	97.2%	94.7%
Half Data (High-Level)	.149	.276	87.7%	94.3%	87.1%	90.1%	.738	.424	77.5%	60.8%
2-Sessions (High-Level)	.119	.281	81.0%	88.2%	79.2%	84.4%	.824	.472	17.9%	12.0%

Table 9: Quantitative comparison of MindEye performance with varying dataset sizes on Subject 1 with the high-level pipeline. Half Data corresponds to MindEye trained with half of the training samples randomly removed. 2-Sessions corresponds to MindEye trained with a random selection of 500 training image samples (or 1,500 training fMRI samples given 3 repetitions per image), equivalent to the number of samples collected across two scan sessions. Notably, image and brain retrieval metrics maintained state-of-the-art performance even when training the model with half of the training samples removed, and reconstruction performance remained competitive with previous models even with reduced training data. This suggests that our MindEye approach is flexible to being trained with smaller datasets.

A.11 Model size comparison with other methods

Method		Parameter Count
Lin et al.		2×1.17M deep models + StyleGAN
Takagi et al.	Low Level	37M linear regression model
	High Level	450M linear regression model
Ozcelik et al.	Low Level	1.45B linear regression model
	High Level	257 separate 12M linear regression models
MindEye	Low Level	206M residual MLP + CNN decoder model
	High Level	996M residual MLP + diffusion prior model

Table 10: Comparison of MindEye parameter count with other competing methods. Other methods primarily rely on linear regression or relatively small deep models.