Mamba-GINR: A Scalable Framework for Spatiotemporal Representation of fMRI

Thilina Balasooriya
Columbia University
tnb2119@columbia.edu

Jubin Choi
Seoul National University
wnqlszoq123@snu.ac.kr

Kevin Valencia
UCLA
kevinval04@ucla.edu

Xihaier Luo, Shinjae Yoo & David Keetae Park
Brookhaven National Laboratory
{xluo,sjyoo,dpark1}@bnl.gov

Abstract

Generalizable implicit neural representations (GINRs) are a powerful paradigm for modeling large-scale functional MRI (fMRI) data, but their adoption is blocked by a key modeling challenge. Prior GINRs built on Transformers cannot scale to 4D fMRI due to the quadratic complexity of attention, preventing promising applications like data compression, temporal interpolation, and representation learning for large-scale scientific data. This work introduces Mamba-GINR, a framework that leverages Mamba as a backbone aiming for linear-time scaling. Our results, benchmarked on standard image datasets (CIFAR-10, CelebA), show it achieves superior reconstruction quality. Critically, we demonstrate Mamba's superior scalability on GINR: it significantly outperforms baselines given an identical token budget and was the only GINR variant that could successfully model sequences at a scale comparable to 4D fMRI data. Further analysis into the placement of learnable queries and the model's internal time delta (Δ) parameter confirms its ability to create robust, high-fidelity representations. By addressing this critical modeling bottleneck, our work has the potential to make GINRs a more viable tool for fMRI analysis. This advance in scalability could enable the continuous representation of entire fMRI sessions, potentially preserving rich temporal dynamics that are often lost to computational constraints. We present this framework as a foundational tool and invite the neuroscience community to collaborate on applying it to explore complex, long-timescale brain activity in large-scale datasets.

1 Introduction

Functional MRI (fMRI) is a cornerstone of modern neuroscience, offering a non-invasive window into brain activity through hemodynamic contrast [1, 2]. fMRI analysis faces two persistent challenges: (*i*) the sheer volume of 4D datasets, which strains storage and compute, and (*ii*) a temporal resolution that is coarse relative to the speed of neural events. These challenges have spurred the development of robust and scalable analysis methods [3]. In this work, we tackle these problems using generalizable implicit neural representations (GINRs). Implicit neural representations (INRs) are coordinate-based networks that learn a continuous mapping from input coordinates (e.g., spatial or temporal) to a signal's value, making them excellent for compactly representing complex data [4, 5]. GINRs extend this by learning a representation that can be rapidly adapted to new data instances [6, 7, 8]. We leverage this framework to tackle our target problems through two tasks: data compression to manage data volume and temporal super-resolution to enhance effective sampling rates.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Data on the Brain & Mind.

However, a critical bottleneck prevents GINRs from being applied to fMRI-scale data. Most state-of-the-art GINRs are built on Transformer backbones [9], whose self-attention mechanism scales quadratically ($O(N^2)$) with the input sequence length. This computational cost is prohibitive for the extremely long sequences of coordinate-time tuples found in fMRI. To overcome this limitation, we replace the Transformer with Mamba, a selective state-space model (SSM) that scales linearly ($O(N)$) with sequence length [10].Our contributions in this work-in-progress are three-fold:

- **Mamba-GINR.** We propose a scalable GINR architecture that replaces the quadratic-cost Transformer encoder with Mamba, enabling linear-time processing of the long coordinate sequences typical of fMRI while retaining a standard coordinate-MLP decoder.
- **Scalability & Generalization.** On standard GINR image benchmarks (CIFAR-10, CelebA), we demonstrate that Mamba-GINR matches or exceeds the performance of Transformer-based models while offering substantially better memory efficiency and throughput, especially at sequence lengths where attention becomes intractable.
- **Architectural Analysis.** We perform ablation studies on design choices specific to the SSM backbone, including chunking strategies for coordinate streams and the role of a time-delta (Δ) feature, identifying the key factors for stable and high-fidelity performance.

2 Related Works

Implicit Neural Representation (INR) and Generalizable INR (GINR). INRs are continuous representations of discretized data (often sets of coordinate-value pairs) learned using MLPs. One major drawback of MLPs is that they exhibit spectral bias, often prioritizing low-frequency information, resulting in overly smooth outputs [11]. A common solution is using Fourier-feature positional embeddings to provide rich frequency information about the coordinates [4]. Other methods, like SIREN, use sinusoidal activation functions to extract high frequency information [5, 12]. Rather than training on a single discretized data instance, GINR changes the paradigm to learn a generalizable function for a given set of discrete functions (Fig. 1). GINRs can be categorized as either a hypernetwork, meta-learning, or conditioning. Hypernetworks use a separate network to take in the data instance and output the weights of an INR hyponetwork [13, 14, 15, 6]. The meta-learning approach has two training loops, one to create an instance-agnostic INR that is initialized to the average dataset member, and an outer loop that does a few gradient descent steps based on the data instance to construct the instance-specific INR [16, 17, 18]. In conditioning methods, the INR decoder component takes a latent representation of the data instance as conditioning [7, 8, 19].

Neural Compression and Time Interpolation for fMRI. A growing body of work applies neural networks to compress 4D fMRI and, to a lesser extent, to enhance or interpolate its temporal sampling. Among compression-focused methods, INRs have been adapted to fMRI by fitting a compact coordinate MLP to each scan; for example, Li *et al.* learn a mapping from spatial coordinates to BOLD time series via reusable, subject-specific activation patterns, achieving strong rate–distortion performance but requiring a bespoke model per volume [20]. Complementing this, Zheng *et al.* propose a hybrid representation with six learnable 2D feature planes (xy/xz/yz and xt/yt/zt) plus a tiny MLP decoder, explicitly targeting spatio-temporal redundancy and outperforming traditional codecs as well as other INR variants, yet still trained per-scan [21]. Outside INRs, broader learned codecs and super-resolution methods have improved spatial fidelity in fMRI, but principled temporal super-resolution (sub-TR interpolation of BOLD dynamics) remains underexplored. This highlights a fundamental limitation of scan-specific INRs and motivates GINRs that learn a reusable prior over a distribution of fMRI signals, enabling cross-subject, cross-session compression and time interpolation at scale—precisely the gap our proposed Mamba-GINR framework aims to address.

3 Methods

3.1 Preliminaries

Implicit Neural Representations. A data instance $x^{(i)} = \{(c_j^{(i)}, y_j^{(i)})\}_{j=0}^{m_n}$ of some dataset $X = \{x^{(i)}\}_{i=0}^{n}$ is defined as having a set of m_n paired coordinate locations $c_j^{(i)} \in \mathbb{R}^{d_{in}}$ and values $y_j^{(i)} \in \mathbb{R}^{d_{out}}$ (e.g., a 2D color image where $d_{in} = 2$ and the RGB value of each pixel is $d_{out} = 3$).

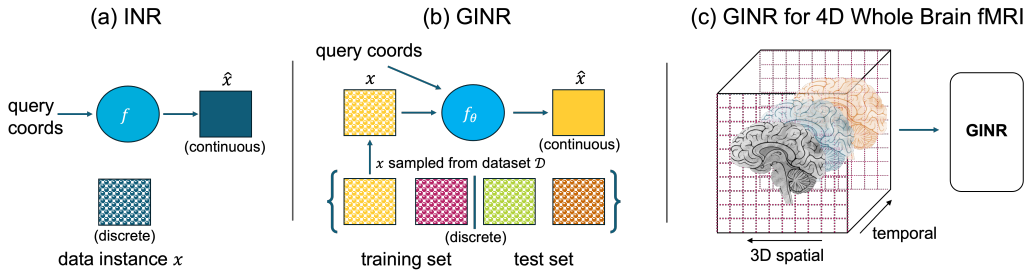

Figure 1: a) INR stores the continuous representation of a single data instance while b) GINR generalizes the process of generating a continuous representation given a data instance as a prior. c) Our goal is to create a model that fits into the GINR paradigm for 4D whole brain fMRI modeling.

An INR f_θ parameterized by parameters θ is a neural network:

$$f_\theta : \mathbb{R}^{d_{in}} \to \mathbb{R}^{d_{out}}$$

that takes coordinates $c \in \mathbb{R}^{d_{in}}$ as inputs and outputs data values $y \in \mathbb{R}^{d_{out}}$. For the example of a 2D color image, using a neural network to represent the image $I(x, y)$ is $f_\theta(x, y) : \mathbb{R}^2 \to \mathbb{R}^3$. Here, input coordinates (x, y) output pixel RGB values.

Generalizable Implicit Neural Representations. Generalizable INRs extend this idea so a single network can represent many signals and generalize to unseen ones. This is often achieved by conditioning the network on some latent representation or embedding z. Mathematically:

$$f_\theta(c, z) : \mathbb{R}^{d_{in}} \times \mathbb{R}^{d} \to \mathbb{R}^{d_{out}}$$

where c is the input coordinates, z is the latent embedding encoding the identity or properties of a specific signal from a set of signals, and θ is the network parameters shared across multiple signals.

Mamba Selective SSM. Mamba [10] is a state space model (SSM) that uses states and transition matrices to define dependencies. SSMs are governed by a set of ordinary differential equations (ODEs) that define this relationship (left), and these are then discretized by defined some Δ timestep (right) [10]:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t) \qquad h_t = \overline{\mathbf{A}}h_{t-1} + \overline{\mathbf{B}}x_{t-1}$$
$$y(t) = \mathbf{C}h(t) + \mathbf{D}x(t) \qquad y_t = \mathbf{C}h_t$$

where $h(t)$ is the current state, $x(t)$ is the input, $y(t)$ is the output, and $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$ are functions of $\mathbf{A}$, $\mathbf{B}$ and Δ. $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$ define the relationship between the change in state ($h'(t)$) and output ($y(t)$) and the state ($h(t)$) and input ($x(t)$).

Selectivity Mechanism. One of Mamba's main contribution is that it is input-dependent. In the previous iteration of the SSM (S4 [22]), $\overline{\mathbf{A}}$, $\overline{\mathbf{B}}$, and Δ were represented by single matrices applied to all inputs. However, Mamba stores an extra dimension of L (input length) for each of these, meaning that it learns a specific transformation for each token in the sequence. This allows it to dynamically forget and remember only the important parts from each token (similar to attention).

Hardware Aware Algorithm and Parallel Scan. Mamba applies a parallel scan (similar to Blelloch Scan [23]) over tokens, exploiting properties of structured matrices [24]. It also uses a hardware-aware algorithm that only materializes the larger $\overline{\mathbf{A}}$ and $\overline{\mathbf{B}}$ parameters in SRAM, a much more efficient part of GPU memory, while doing all other computations in the high-bandwidth memory (HBM) to maximize efficiency. These together make Mamba a linear alternative to Transformers that are bound by $O(L^2)$ time complexity.

3.2 Mamba-GINR

Mamba-GINR is composed of a Mamba encoder which generates the set of latents $Z^{(i)} = \{z_j^{(i)}\}_{j=0}^{n}$ (i representing the data instance and j representing each latent) and the INR Decoder which uses locality-biased cross-attention between a coordinate query and $Z^{(i)}$ as the keys to create an aggregate modulation vector that is then used to condition linear layers and output the corresponding value.

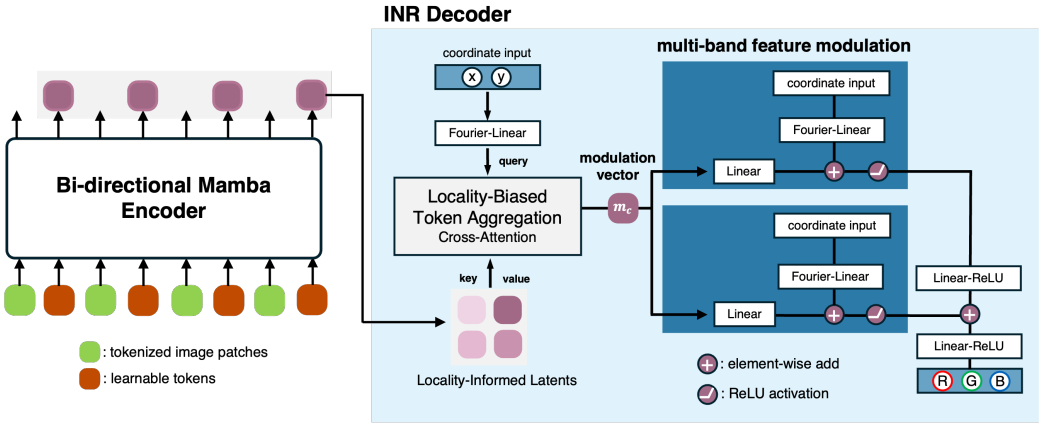

Figure 2: Mamba-GINR architecture: The Bi-directional Mamba Encoder creates a latent representation of the input data which is passed to the locality-informed decoder which uses biased cross-attention to effectively predict the value for any coordinate query. Figure inspired by [25].

Mamba Encoder. The data instance $x^{(i)}$ is first patchified and tokenized as a set of tokens $d^{(i)} = \{d_j^{(i)} \in \mathbb{R}^d\}_{j=0}^{L_d}$, where L_d and d represent the number and dimension of the data tokens, respectively. A set of instance-agnostic learnable tokens $t = \{t_i \in \mathbb{R}^d\}_{i=0}^{L_l}$ is initialized using a fixed sinusoidal function. We interleave the learnable tokens with the data tokens (Fig. 2) and pass them to the Mamba encoder, yielding a token sequence of length $L = L_d + L_l$. The Mamba encoder uses k bidirectional Mamba blocks, where k is the encoder depth. Each block takes in the previous sequences and runs it forwards and backwards through the original Mamba block, applying an RMSnorm [26] and residual connection. The scan order of the Mamba encoder can be varied depending on the input shape (e.g. for 2D images, we use bi-directional scanning method as described in [27]). The encoder output at the learnable token positions is extracted as the latent set $z^{(i)} = \{z_j^{(i)} \in \mathbb{R}^d\}_{i=0}^{L_l}$

Cross-Attention Based INR Decoder. The overall structure of the INR decoder is adapted from [25], with the main difference being the locality-bias strategy explained in the subsection below. The coordinate passed into the INR is first transformed using a Fourier positional embedding and linear projection to create the query vector:

$$\gamma_\sigma(c) = [\cos(\pi\omega_j c_i), \sin(\pi\omega_j c_i)], \quad q_c = \text{ReLU}\left(W\gamma_\sigma(c) + b\right) \tag{1}$$

Where $1 < i < d_{in}$ and $0 < j < n-1$. The Fourier feature $\gamma_\sigma(c)$ is calculated with some bandwidth $\sigma > 1$ where $n = \frac{d}{2d_{\text{in}}}$. The frequencies from 1 to σ are distributed on a logarithmic scale for each ω_j [25]. As shown in Fig. 2, following the INR decoder method described in [25], we use a projected Fourier-feature positional embedding of the input coordinate as a query for cross-attention (to aggregate latents) with the latent set $z^{(i)}$. The purpose of the cross-attention is to create a locality aware modulation vector m_c by learning the locality information pertinent to the evaluation coordinate in each latent vector $z_j^{(i)}$. However, in contrast to [25] which uses a Transformer encoder, the Mamba encoder's implicit sequential bias allows us to heuristically determine the importance of each latent based on how far it is in the input sequence from the queried coordinate. For examples, after patching $x^{(i)}$ into the token set $d^{(i)}$, for any given coordinate c we can find the patch $p_c^{(i)}$ such that $c \in p_c^{(i)}$ which corresponds to the token $d_c^{(i)}$. The distance of a latent $z_j^{(i)}$ from the position of $d_c^{(i)}$ can then be used as a heuristic to bias the cross attention. This is formulated as the following:

$$\text{bias}_{c,z_l^{(i)}} = -\alpha\left(\frac{l}{L_l} - \frac{c}{L_d}\right)^2, \tag{2}$$

where $\text{bias}_{c,l}$ represents the bias between a coordinate in the c^{th} data token and the l^{th} latent $z_l^{(i)}$. With this bias, we formulate cross-attention latent aggregation as follows:

$$m_c = \text{softmax}\left(\frac{q_c z^{(i)\top}}{\sqrt{d}} + \text{bias}_{c,z^{(i)}}\right) z^{(i)} \tag{3}$$

We adopt the multi-band feature modulation of [25] and refer there for details.

4 Experiments

Datasets. Our primary dataset is the WU-Minn HCP dataset of healthy subject resting-state fMRI volumes. The dataset contains 1,084 subjects with 4D fMRI volumes of size $96 \times 96 \times 96 \times 1200$ (H, W, D, T). We split each sequence into 2 second windows and train on 12 windows from 400 subjects (4,800 volumes of size $96 \times 96 \times 96 \times 2$). Volumes are resized to smaller spatial dimensions and sliced into 2D images for various tasks (see below). For 2D image reconstruction benchmark testing against state of the art methods, we also use the standard CIFAR-10 (32×32) and CelebA (64×64) image datasets.

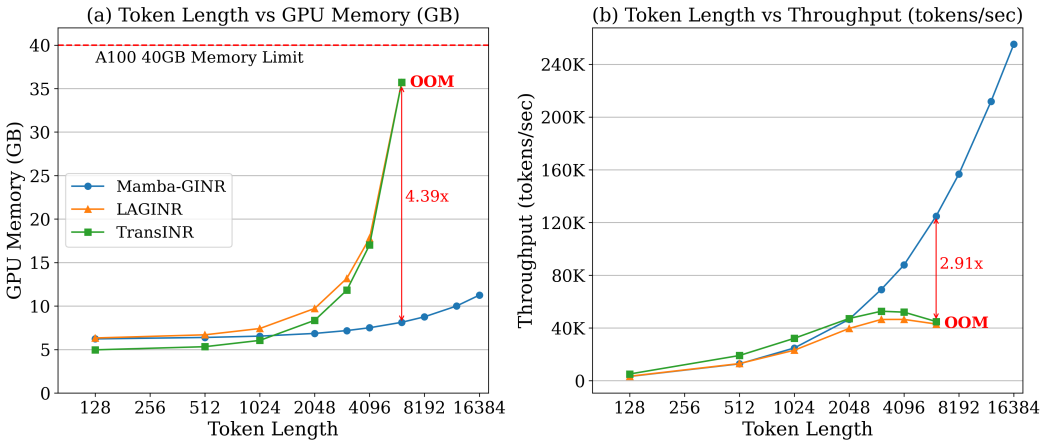

Figure 3: (a) Maximum GPU memory occupied by each model at inference time plotted against the number of input data tokens and (b) total throughput of tokens per seconds plotted against the number of input data tokens. OOM: Out-of-Memory Error

4.1 Analysis of Scaling Metrics

Experimental Setting. We benchmark Mamba-GINR against two patch-based GINR state-of-the-art techniques, LAGINR [25] and TransINR [28] by comparing GPU usage and throughput as we increase the number of input data tokens. We use the dataset of $96 \times 96 \times 96 \times 2$ volumes and define a set of different patch sizes, each of which results in a different number of data tokens/patches L_d. These sets of patches are passed through each model for inference to determine the maximum GPU memory utilized and the throughput (tokens/sec). We set a constant small number of $L_l = 128$ learnable tokens to reduce and standardize the effect of decoding each of the $96^3 \times 2$ output pixel queries between models. Throughput is calculated as an average of 20 training steps:

$$throughput = \frac{num\_tokens}{t_{elapsed}} = \frac{20 \times (n + L) \times B}{t_{elapsed}} \tag{4}$$

We test on a single NVIDIA A100 40GB GPU with a batch size of $B = 2$.

Results. Fig. 3 a) demonstrates that the transformer-based methods LAGINR and TransINR are unable to parse more than $L = 6,144$ data tokens on the A100 40GB GPU, while Mamba-GINR can handle much greater than $L = 16,384$ (less than 35% of the GPU capacity is used up to this point). Fig. 3 b) shows that transformer-based methods throughput plateaus and decreases after a token length around 3000 while Mamba-GINR throughput continues to increase. Due to a large amount of queries in the decoder, there is a constant initial amount of time that is added to all experiments. This explains the increase in throughput for small amounts of data tokens, because the number of tokens increases fast while the time elapsed in the encoder is dominated by time spent in the decoder (which is constant and standardized between models). However, for Transformer baselines the encoder quickly dominates runtime, whereas for Mamba-GINR the decoder remains the bottleneck even at large token counts, so throughput continues to rise.

Model	Patch Size†	MSE for 4D Volume Size†			
		$24^3 \times 2$	$36^3 \times 2$	$48^3 \times 2$	$64^3 \times 2$
Mamba-GINR	$2^3 \times 2$	$\mathbf{8.00 \times 10^{-5}}$	$\mathbf{9.24 \times 10^{-5}}$	$\mathbf{1.59 \times 10^{-4}}$	OOM
Mamba-GINR	$4^3 \times 2$	1.52×10^{-4}	1.82×10^{-4}	2.32×10^{-4}	$\mathbf{2.60 \times 10^{-4}}$
LAGINR [25]	$2^3 \times 2$	1.81×10^{-4}	OOM	OOM	OOM
LAGINR [25]	$4^3 \times 2$	1.44×10^{-4}	2.20×10^{-4}	4.78×10^{-4}	OOM
TransINR [28]	$2^3 \times 2$	2.28×10^{-4}	OOM	OOM	OOM
TransINR [28]	$4^3 \times 2$	1.73×10^{-4}	2.38×10^{-4}	5.56×10^{-4}	OOM
SCENT [29]	$1^3 \times 1$	7.81×10^{-4}	OOM	OOM	OOM

Table 1: MSE loss of 4D fMRI volume reconstruction for each model at different patch sizes. L_l is chosen such that the ratio $L_d : L_l = 9 : 1$ to maintain fairness between models. †: values are represented as **spatial**^{3}$\times$**temporal** sizes; OOM: Out-of-Memory Error

4.2 4D fMRI Reconstruction

Experimental Setting. We benchmark Mamba-GINR on reconstruction of 4D fMRI volumes against two the patch-based GINR state-of-the-art techniques, LAGINR [25] and TransINR [28], and also a non-patched Perceiver-IO cross-attention based method SCENT [29]. We take the 4800 volumes from the WU-Minn dataset and downsample each 4D volume into four different spatial dimensions ($dim = 24, 36, 48, 64$) and also compare the effect of two different patch sizes ($2^3 \times 2$ and $4^3 \times 2$). To fairly assess performance, since the extraction of learnable tokens to summarize data token information is a dimension reduction technique, we use the same ratio of $L_d : L_l = 9 : 1$ of data tokens to learnable tokens for each volume/patch size. The number of data tokens is the same as the number of patches. For Mamba-GINR, we do a bidirectional row-major order scan along dimensions H, W, D, T in that order. Pixel-wise MSE loss is calculated as an evaluation metric for reconstruction on a test set of 12 data volumes from each of 50 subjects (different from training subjects).

Results. In Table 1 for the three smaller volume sizes, Mamba-GINR with a small patch size shows the best performance with lowest MSE reconstruction loss, while for the $64^3 \times 2$ volume only the Mamba-GINR model with $4^3 \times 2$ patch size was able to run without memory errors. Attention-based methods were unable to use smaller patches for medium-large size data while Mamba-GINR is able to do so and while also demonstrating that the smaller patch size is better for minimizing loss, likely due to preserving more rich encoding of the volume in the latent space. We also do ablations on patch size and number of learnable tokens below to demonstrate this effect. For the $4^3 \times 2$ patch size it is also noteworthy that as we scale from a spatial volume size of 24 to 48, the transformer-based methods' reconstruction MSE degrades by a factor > 3.2 while Mamba-GINR's performance only worsens by a factor of 1.5 on the more complex data. Fig. 4 demonstrates a qualitative evaluation of axial slices where we can observe that Mamba-GINR is able to more accurately reproduce the high frequency information in the volume.

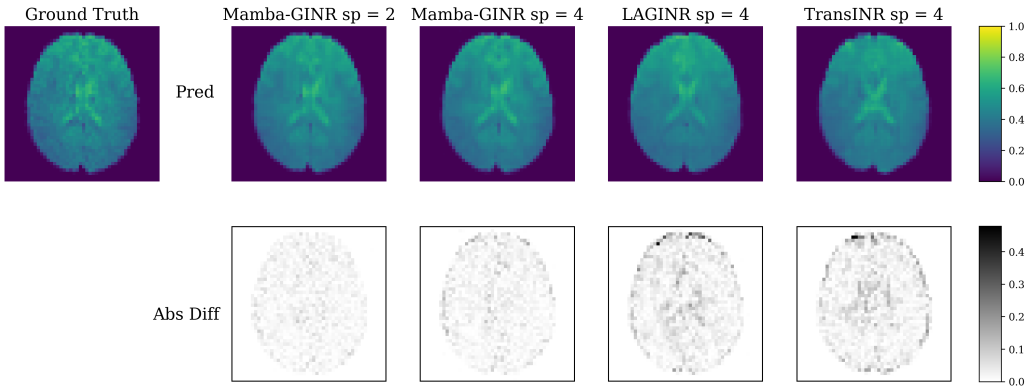

Figure 4: Qualitative evaluation of 2D slices taken from the full $48^3 \times 2$ 4D fMRI volume reconstruction for each GINR model. Intensity is fMRI BOLD signal scaled to [0, 1]. (sp - spatial patch size)

L_l	**Ratio** $L_d : L_l$	**MSE**
128	32	3.90×10^{-4}
256	16	3.09×10^{-4}
512	8	2.70×10^{-4}
768	$5.\overline{33}$	2.49×10^{-4}
1024	4	$\mathbf{2.14 \times 10^{-4}}$

(a) Patch size $3^3 \times 2$ and volume size $48^3 \times 2$, constant $L_d = 4096$ for 400 epochs.

Patch Size	**#Data Tokens**	**MSE**
$4^3 \times 2$	1728	4.36×10^{-4}
$4^3 \times 1$	3456	3.94×10^{-4}
$3^3 \times 2$	4096	3.88×10^{-4}
$3^3 \times 1$	8192	3.50×10^{-4}
$2^3 \times 2$	13824	3.56×10^{-4}
$2^3 \times 1$	27648	$\mathbf{2.16 \times 10^{-4}}$

(b) Constant ratio $L_d : L_l$ 20:1 and volume size $48^3 \times 2$

Table 2: Ablations on the number of learnable tokens L_l and the patch size using Mamba-GINR.

Ablations on # of Learnable Tokens and Patch Size We do ablations on the number of learnable tokens L_l and patch size (which correlates to number of data tokens L_d per volume) to demonstrate their role in reconstruction performance. Both increasing the number of learnable tokens and decreasing the patch size (therefore increasing the number of data tokens for a given volume) correlate with a better reconstruction. The former is likely due to a more expressive latent representation while the latter is likely due to a richer tokenized representation prior to the latent dimension reduction.

4.3 2D Image Reconstruction

Experimental Setting. In addition to 4D volumes, we benchmark Mamba-GINR against LAGINR [25], TransINR [28], and SCENT [29] on reconstruction of images from the standard CIFAR-10 and CelebA baseline datasets and also 2D slices of the HCP fMRI volumes. To prepare the 2D fMRI data, we take the same 4800 volumes from the WU-Minn dataset and take the middle slice along the W dimension at time $t = 0$ to get a 96×96 slice. We choose the patch sizes and TransINR n_groups parameter so that for each of these we have $L_d = 256$ data tokens and $L_l = 256$ learnable tokens for fair comparison. For Mamba-GINR, we do a bidirectional row-major order scan along dimensions H, W in that order, and again pixel-wise MSE loss is calculated for reconstruction on the test sets for each dataset.

Results. Mamba-GINR exhibits superior performance on the CelebA/HCP slices and also performs competitively on the CIFAR-10 baseline with LAGINR while far outperforming other models. It is likely that LAGINR and Mamba-GINR perform closely on CIFAR-10 due to the simplicity and granularity of the dataset; Mamba-GINR's bias mechanism would be more advantageous when there is a meaningful spatial continuity in the data (which is more true for higher resolution images).

4.4 Learnable Tokens Distribution Studies

This work also presents findings about the placement of learnable tokens in a sequence for Mamba encoding and analysis of how Mamba handles these learnable tokens using its selectivity mechanism.

	CIFAR-10 (32×32)	**CelebA (64×64)**	**HCP 2D Slice (96×96)**
Mamba-GINR (Ours)	60.40	**50.65**	**35.59**
LAGINR [25]	**61.40**	46.92	33.66
TransINR [28]	41.15	36.47	28.50
SCENT [29]	28.26	28.81	23.16

Table 3: Reconstruction PSNR for each model. The patch size for each dataset is chosen such that L_d and L_l are both 256 for each of the patch-based methods.

t-SNE Analysis of Learnable and Data Tokens Mamba-GINR is run on an image from the CIFAR-10 32x32 resolution dataset with a patch size of 2x2 and 256 learnable tokens, which leads to token lengths $L_l = 512$, $L_d = 512$, $L = 512$. t-SNE analysis is done on the total set of L tokens, divided into two classes for learnable tokens and data tokens. Fig. 5 b) shows that there is obvious clustering of learnable tokens and data tokens, demonstrating that the selectivity mechanism in the Mamba encoder is able to effectively create latents that have a role separate from the sequential data information in the data tokens, despite being a sequentially-informed model. The following section

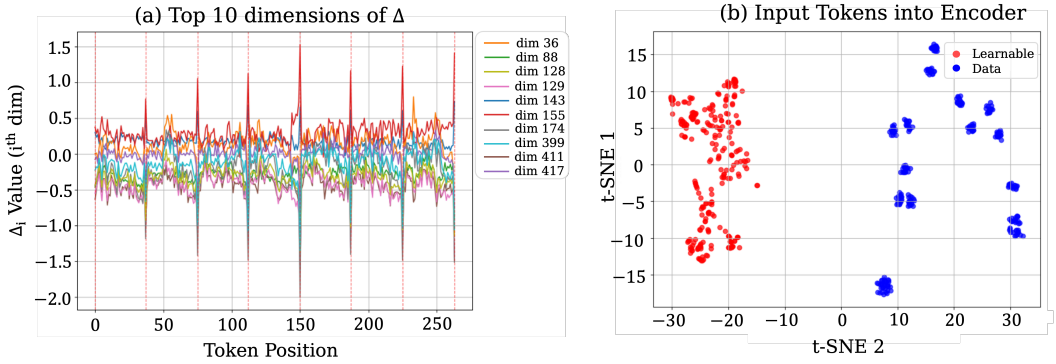

Figure 5: (a) The top $k = 10$ dimensions of Δ ranked by normalized difference between learnable token and data token values (b) t-SNE analysis of all input tokens, divided by class of data token or learnable token

provides insight into how the Mamba model may be able to make this distinction internally between learnable tokens and data tokens.

Comparison of Δ for Learnable vs Data Tokens As described in 3.1.2, Mamba utilizes Δ as a parameter to discretize the set of ordinary differential equations that describe the state space model. In addition, the selectivity mechanism in Mamba ensures that Δ is input dependent (i.e. varies for each input token), which allows it to describe how much each output should change based on individual tokens. To demonstrate the effect of adding equidistantly interleaved learnable queries to an input sequence on Mamba's learning, we plot the values of a subset of the dimensions of Δ at each input token position. For clarity, we choose to plot only the top k dimensions using the normalized different between mean values at learnable vs data token positions, which ranks the dimensions according to the amount of difference observed between the positions of the learnable token and the data token. Fig. 5 a) shows that at learnable token positions, denoted with the dotted red vertical line, these dimensions show a vast difference compared to data token positions. This demonstrates that Mamba is able to learn to distinguish these token classes and effectively utilize the learnable tokens.

5 Discussion and Conclusion

Mamba-GINR delivers linear-time scaling for GINR while maintaining or improving reconstruction quality. It sustains long input sequences that exhaust Transformer-based GINRs and increases throughput with sequence length (Fig. 3); across feasible settings it attains the lowest 4D fMRI reconstruction errors with improved high-frequency fidelity (Table 1, Fig. 4). Ablations show that increasing learnable tokens and shrinking patches reduces error, and analyses of embeddings and the input-dependent Δ highlight distinct roles for data and learnable tokens (Fig. 5). By replacing the quadratic-cost encoder with a selective SSM and pairing it with a locality-biased cross-attention decoder, whole-brain sequences become tractable as a linear operation.

Our next step is a neural compression and time-interpolation pipeline that uses the modulation vector for storage and continuous-time queries to produce physiologically consistent sub-TR estimates. While these components are not yet implemented, this work establishes the necessary foundations—(i) a linear-time encoder that processes whole-brain sequences, (ii) locality-preserving latents that admit queries at arbitrary coordinates, and (iii) empirical scaling laws (token count, patch size). If realized, a whole-brain GINR could reframe motion correction, resampling, denoising, and slice-time alignment as queries to a single continuous representation; enable on-the-fly reslicing to any atlas or resolution, mixed-resolution analyses, and streaming storage via compact decodable codes. The same recipe—long-sequence SSM encoders with locality-aware decoders—extends naturally to other 3D + time scientific data (e.g., climate reanalysis), offering unified continuous surrogates that reduce I/O and storage while enabling analysis at arbitrary spatiotemporal granularity.

Acknowledgments and Disclosure of Funding

This work was supported by the U.S. Department of Energy (DOE), Office of Science (SC), Advanced Scientific Computing Research program under award DE-SC-0012704 and used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility using NERSC award NERSC DDR-ERCAP0030592.

References

[1] Kenneth K. Kwong, John W. Belliveau, David A. Chesler, Inna E. Goldberg, Robert M. Weisskoff, Brigitte P. Poncelet, David N. Kennedy, Bernice E. Hoppel, Mark S. Cohen, Robert Turner, Hui-Ming Cheng, Thomas J. Brady, and Bruce R. Rosen. Dynamic magnetic resonance imaging of human brain activity during primary sensory stimulation. *Proceedings of the National Academy of Sciences of the United States of America*, 89(12):5675–5679, 1992.

[2] Peter A. Bandettini. Twenty years of functional MRI: the science and the stories. *NeuroImage*, 62(2):575–588, 2012.

[3] Russell A. Poldrack, Chris I. Baker, Joke Durnez, Krzysztof J. Gorgolewski, Paul M. Matthews, Marcus R. Munafò, Thomas E. Nichols, Jean-Baptiste Poline, Edward Vul, and Tal Yarkoni. Scanning the horizon: towards transparent and reproducible neuroimaging research. *Nature Reviews Neuroscience*, 18(2):115–126, 2017.

[4] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[6] Yinbo Chen and Xiaolong Wang. Transinr: A meta-learning approach to implicit neural representation. In *European Conference on Computer Vision (ECCV)*, 2022.

[7] Junyoung Kim, Seohee Kim, Seungjoo Lee, Jong Chul Choi, and Kyoung Mu Lee. Generalizable implicit neural representations via instance pattern composers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12701–12711, 2023.

[8] Doyup Lee, Chiheon Kim, Minsu Cho, and Wook-Shin Han. Locality-aware generalizable implicit neural representation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[10] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024.

[11] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks, 2019.

[12] Xihaier Luo, Wei Xu, Yihui Ren, Shinjae Yoo, and Balu Nadiga. Continuous field reconstruction from sparse observations with implicit neural networks. *arXiv preprint arXiv:2401.11611*, 2024.

[13] Qi Wu, David Bauer, Yuyang Chen, and Kwan-Liu Ma. Hyperinr: A fast and predictive hypernetwork for implicit neural representations via knowledge distillation, 2023.

[14] Filip Szatkowski, Karol J. Piczak, Przemysław Spurek, Jacek Tabor, and Tomasz Trzciński. Hypernetworks build implicit neural representations of sounds, 2023.

[15] Elizabeth Fons, Alejandro Sztrajman, Yousef El-laham, Alexandros Iosifidis, and Svitlana Vyetrenko. Hypertime: Implicit neural representation for time series, 2022.

[16] Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one, 2022.

[17] Emilien Dupont, Hrushikesh Loya, Milad Alizadeh, Adam Goliński, Yee Whye Teh, and Arnaud Doucet. Coin++: Neural compression across modalities, 2022.

[18] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations, 2021.

[19] Shuyi Zhang, Ke Liu, Jingjun Gu, Xiaoxu Cai, Zhihua Wang, Jiajun Bu, and Haishuai Wang. Attention beats linear for fast implicit neural representation generation, 2024.

[20] Ruoran Li, Runzhao Yang, Wenxin Xiang, Yuxiao Cheng, Tingxiong Xiao, Lu Yang, and Jinli Suo. A compact implicit neural representation for efficient storage of massive 4d functional magnetic resonance imaging. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 4914–4922, 2025.

[21] Wuyang Zheng, Jiarui Meng, Jiaqi Zhang, Jian Zhang, and Siwei Ma. Hybrid representation for 4d medical image compression. In *2024 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, pages 1–5. IEEE, 2024.

[22] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces, 2022.

[23] Guy E Blelloch. Scans as primitive parallel operations. *IEEE Transactions on Computers*, 38(11):1526–1538, 1989.

[24] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Re. Hippo: Recurrent memory with optimal polynomial projections, 2020.

[25] Lee Doyup, Kim Chiheon, Cho Minsu, and Han Wook-Shin. Locality-aware generalizable implicit neural representation, 2023.

[26] Biao Zhang and Rico Sennrich. Root mean square layer normalization, 2019.

[27] Haohao Qu, Liangbo Ning, Rui An, Wenqi Fan, Tyler Derr, Hui Liu, Xin Xu, and Qing Li. A survey of mamba, 2025.

[28] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations, 2022.

[29] David Keetae Park, Xihaier Luo, Guang Zhao, Seungjun Lee, Miruna Oprescu, and Shinjae Yoo. Scent: Robust spatiotemporal learning for continuous scientific data via scalable conditioned neural fields, 2025.

A Implementation Details

A.1 Tokenization

Tokenization of the data tokens is done by first dividing the n-dimensional volume into patches based on a decided patch size. Each patch is then passed through a linear layer and a Fourier Positional Embedding is concatenated based on its coordinate position to create each token. Each token is designated a position by laying them out in a specific sequence (in this case, we choose an order of dimensions to unfold the patches). This is done so that it can be passed into the Bi-directional Mamba sequential encoder.

A.2 Training Details

Below is a table of training model and hyperparameters details for ease of reproducibility and transparency.

	DATA TYPE AND TASK	
DATA STATISTICS	**Image Reconstruction**	**fMRI Reconstruction**
Input Spatial Size	See 4.3 and Table 3	See Table 1.
Input Temporal Dim Size	N/A	2
Train / Val size	50000 / 5000 (CIFAR-10) 163000 / 20000 (CelebA) 4800 / 600 (fMRI HCP)	4800 / 600
Normalization	RGB each to {0, 1}	global min-max scaling to {0, 1}
TRAINING		
Epochs	200	400
Batch size (train/val)	16 / 16	2 / 2
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$)	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$)
LR schedule	Constant + Warmup + LR Cooldown	CosineAnnealing + Warmup
Warmup Epochs	15%	10%
LR (min, max)	$(2 \times 10^{-4}, 1 \times 10^{-6})$	$(1.5 \times 10^{-4}, 3 \times 10^{-5})$
MODEL		
Size (# params)	9.0M	35.1M
Latent dim size	256	512
# Learnable tokens	256	See Table 1.

Table 4: Dataset statistics and training specifics for the two data types used in this work.