NEUROSKETCH: A PRACTICAL DESIGN RECIPE FOR NEURAL DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural decoding is fundamental to brain-computer interfaces, with growing applications in healthcare. Previous research has focused on leveraging signal processing and deep learning methods to enhance neural decoding performance. However, systematic guidance on architectural design for neural decoding remains limited. In this study, we develop NeuroSketch, a practical design recipe for neural decoding, through a basic architecture study followed by macro- and micro-level optimization. Comparing nine basic architectures, we find that CNN-2D outperforms other architectures in neural decoding tasks and explore its effectiveness from temporal and spatial perspectives. Building on this backbone, we combine gradual feature-map expansion and early downsampling at the macro level with grouped convolutions at the micro level. These choices form the recipe, which we instantiate as NeuroSketch-Base (1.4M parameters) and NeuroSketch-Large (4.2M parameters). The recipe is developed and evaluated through over 4,500 experiments across eight tasks spanning visual, auditory, and speech modalities and EEG, SEEG, and ECoG signals. Against ten baselines, the two variants collectively achieve the best accuracy on each task. Our code is available at `https://anonymous.4open.science/r/NeuroSketch-anonymous-A5E4/`.

1 INTRODUCTION

Neural decoding infers stimuli, cognitive states, or intentions from recorded brain signals and is a central component of brain-computer interfaces (Van Gerven et al., 2019). These signals are typically acquired using non-invasive methods such as electroencephalography (EEG) (Teplan, 2002) or invasive techniques such as stereoelectroencephalography (SEEG) (Talairach, 1974) and electrocorticography (ECoG) (Shenoy et al., 2007). While EEG offers practicality and low cost for research (Zhang et al., 2021; Altaheri et al., 2023; Rahman et al., 2021) and clinical applications (Pani et al., 2022; Saminu et al., 2023), invasive methods provide superior temporal and spatial resolution by capturing signals from deeper brain structures (Chaddad et al., 2023).

In neural decoding, recorded brain signals are characterized by transient temporal dynamics (King & Dehaene, 2014) and spatial locality (Mahjoory et al., 2024), which present unique modeling demands. Traditional time series data, such as weather, electricity consumption, and traffic flow, are sampled at relatively low frequencies (*e.g.,* minutes or hours) and often exhibit significant periodicity or trends. However, brain signals require higher temporal resolution to capture the brain's rapidly changing states. Even within brain signals, their characteristics can differ significantly across various scenarios. From a temporal perspective, brain signals typically contain crucial information within much shorter time frames in neural decoding tasks compared to other scenarios, such as sleep staging. In sleep staging tasks, brain signals are recorded overnight, with a labeling resolution of 30 seconds for each sleep stage (Phan & Mikkelsen, 2022). In contrast, in the Rapid Serial Visual Presentation (RSVP) paradigm for image stimulus decoding, relevant information is encoded within just a few hundred milliseconds (Butts et al., 2007). From a spatial perspective, task-driven neural activity in decoding paradigms differs from activity associated with pathological conditions. Pathological conditions such as generalized epilepsy exhibit widespread brain activity, resulting in non-specific neural activations (Stafstrom & Carmant, 2015). In neural decoding tasks, however, external stimuli evoke localized activations that are typically restricted to functionally specialized regions and recorded by spatially proximate electrodes. For example, speech stimuli predominantly activate the left inferior

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

frontal gyrus (Leonard et al., 2024), while visual stimuli engage the visual cortex (Grill-Spector & Malach, 2004).

Previous work on neural decoding has explored both signal processing and deep learning approaches. Signal processing methods aim to improve signal quality (Peksa & Mamchur, 2023) and extract task-relevant features (Wittevrongel et al., 2020; Proix et al., 2022; Safi & Safi, 2021), but manually designing these features requires substantial domain expertise and effort. Deep learning methods reduce this reliance on handcrafted features by learning task-relevant representations directly from neural signals (Angrick et al., 2019; Makin et al., 2020; Wilson et al., 2020; Willett et al., 2023; Metzger et al., 2023; Zheng et al., 2024). However, systematic guidance on architectural design across diverse neural decoding tasks and recording types remains limited, despite the gains achieved through architectural studies in other domains (Yu et al., 2022; 2023; Luo & Wang, 2024; Wang et al., 2025b). For instance, in energy forecasting tasks, ModernTCN (Luo & Wang, 2024) reduces MSE by 13.9% on ETTm2 (Zhou et al., 2021) compared to convolution-based models by using large kernel convolutions and a ConvFFN module. In image classification tasks, ConvNeXt (Liu et al., 2022) surpasses ResNet-50 (He et al., 2016) on ImageNet (Deng et al., 2009) through optimizations in stage compute ratio and convolutional methods. We therefore develop NeuroSketch, a practical design recipe for neural decoding.

To develop this recipe, we investigate two core questions:

Q1: Which basic model architecture is best suited to capture temporal and spatial patterns in brain signals?

Q2: Based on an appropriate architecture, how can we improve neural decoding performance through macro-to-micro architectural optimization?

As shown in Figure 1(a), by answering the above questions, we consistently improve the decoding performance at each step. We first study nine basic architectures (CNN (He et al., 2016), GRU (Chung et al., 2014), Transformer (Vaswani et al., 2017), and their variants) and find that CNN-2D outperforms the others. This finding is consistent with the temporal and spatial locality of neural signals, which CNN-2D captures through local convolutional kernels. Building on the backbone study, we further develop the recipe using a macro-to-micro paradigm, analyzing the latent space transformation during forward propagation and optimizing the calculation of micro components. Specifically, we find that gradual feature-map expansion and early downsampling at the macro level, together with grouped convolutions at the micro level, help balance decoding accuracy and computational cost. Taken together, the findings across all stages form the NeuroSketch recipe. We instantiate this recipe as NeuroSketch-Base and NeuroSketch-Large, with 1.4M and 4.2M parameters, respectively. The two variants collectively achieve the best or tied-best accuracy on all eight decoding tasks among the evaluated models (Figure 1(b)), supporting the practical value of the recipe for neural decoding.

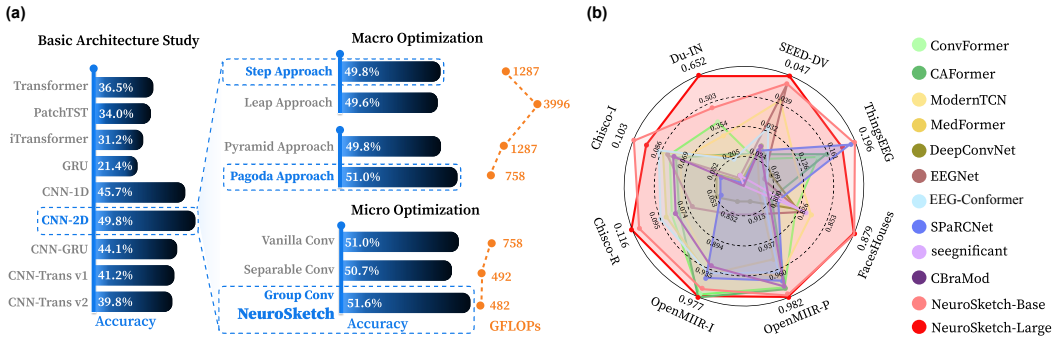


Figure 1: **Overview of NeuroSketch and its model instantiations. (a)** Roadmap for developing the practical design recipe. **(b)** Comparison of NeuroSketch-Base and NeuroSketch-Large with ten baselines across eight neural decoding tasks.

In summary, the main contributions of this research are as follows:

1. We develop NeuroSketch, a practical design recipe for neural decoding, and instantiate it as NeuroSketch-Base and NeuroSketch-Large, tailored to the temporal and spatial characteristics of brain signals.
2. From a technical perspective, we systematically reexamine the design space and consider several critical components. Starting with exploring basic architectures, we progressively investigate architectural design principles from macro- to micro-level.
3. From an experimental perspective, we develop and evaluate the recipe through over 4,500 experiments across eight neural decoding tasks related to vision, hearing, and speech, recorded by different types of brain signals. The experimental results demonstrate that the two model instantiations collectively achieve state-of-the-art decoding performance across all evaluated datasets.

2 Roadmap for Developing a Practical Design Recipe

This section develops NeuroSketch, our practical design recipe for neural decoding. After introducing the experimental setup (Section 2.1), we address Q1 through a basic architecture study (Section 2.2). We then address Q2 through macro- and micro-level optimization (Sections 2.3 and 2.4). Finally, Section 2.5 consolidates these findings into the recipe and introduces its two instantiations, NeuroSketch-Base and NeuroSketch-Large.

2.1 Experimental Setup

To develop the recipe, we conduct 1,764 experiments across eight neural decoding tasks spanning three modalities (visual, auditory, and speech) and three signal types (EEG, SEEG, and ECoG). Table 1 summarizes the datasets, with further details provided in Appendix B. Detailed model configurations and training settings are provided in Appendices C.1 and D.

Table 1: **Overview of the datasets used in our experiments.** We evaluate on six datasets in total. For *Chisco* and *OpenMIIR*, we partition them by task, yielding eight distinct tasks overall.

Categories	Datasets	Tasks	Signal Type
Speech	Du-IN (Zheng et al., 2024)	Reading Aloud	SEEG
	Chisco-R (Zhang et al., 2024)	Silent Reading	EEG
	Chisco-I (Zhang et al., 2024)	Imagined Speech	EEG
Visual	FacesHouses (Miller, 2019)	Binary Image Decoding	ECoG
	ThingsEEG (Gifford et al., 2022)	Multi-Class Image Decoding	EEG
	SEED-DV (Liu et al., 2024a)	Video Decoding	EEG
Auditory	OpenMIIR-P (Stober et al., 2015)	Music Perception	EEG
	OpenMIIR-I (Stober et al., 2015)	Music Imagery	EEG

2.2 Basic Architecture Study

To address Q1, we compare nine architectures from four families: convolutional neural networks (CNNs), gated recurrent units (GRUs), Transformers, and hybrids. Each takes input $\mathbf{X} \in \mathbb{R}^{B\times C\times L}$, where B, C, and L denote batch size, recording channels, and sequence length. CNN-1D combines all input channels through temporal filters, whereas CNN-2D reshapes the input to $\mathbf{X}' \in \mathbb{R}^{B\times 1\times C\times L}$ and applies local kernels along time and channel dimensions. The GRU sequentially updates its hidden state from the current input and previous state, while the Transformer models temporal dependencies through self-attention. We also evaluate PatchTST (Nie et al., 2023) and iTransformer (Liu et al., 2024b). PatchTST divides each channel into patches of length P and stride S, yielding $N = \lfloor (L-P)/S \rfloor + 2$ patches. Transformer layers process the patched input $\mathbf{X}_p \in \mathbb{R}^{B\times C\times N\times P}$ independently across channels. In contrast, iTransformer treats each channel as a token to model inter-channel dependencies. Hybrids comprise CNN–GRU and two CNN–Transformer variants. The first CNN–Transformer stacks Transformer layers after a CNN feature extractor (Song et al., 2022), whereas the second combines convolution and self-attention within each block (Kim et al., 2022). CNN–GRU similarly stacks GRU layers after a CNN feature extractor.

Table 2: **Results of the basic architecture study and subsequent analyses.** v1 and v2 refer to the two hybrid methods discussed above. The ratio shown in the middle of the table (*e.g.,* 4:1) indicates the proportion of CNN to Transformer layers in each hybrid model. The best results are in **bold** and the second best are underlined.

Models \ Datasets	Du-IN		SEED-DV		ThingsEEG		FacesHouses		OpenMIIR-P		OpenMIIR-I		Chisco-R		Chisco-I	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Basic Architecture Study																
CNN-1D	.451	.446	.061	.054	**.202**	**.191**	.799	.796	.973	.973	.968	.968	.104	.071	.100	.073
CNN-2D	**.647**	**.641**	.063	.058	.184	.177	**.886**	**.885**	**.981**	**.980**	**.982**	**.982**	**.127**	**.113**	.114	**.095**
GRU	.353	.343	.025	.001	.040	.037	.774	.773	.146	.046	.136	.039	.123	.091	**.117**	.086
Transformer	.085	.080	.042	.028	.005	.000	.795	.794	.953	.952	.911	.910	.069	.015	.057	.012
PatchTST	.060	.051	.038	.029	.006	.001	.845	.838	.906	.904	.766	.760	.050	.005	.050	.005
iTransformer	.097	.085	.029	.005	.005	.000	.730	.728	.729	.698	.780	.774	.064	.013	.061	.014
CNN-GRU	.416	.409	.057	.049	.172	.166	.803	.801	.957	.954	.943	.942	.092	.064	.087	.063
CNN-Trans v1	.333	.326	**.067**	**.060**	.046	.027	.752	.751	.948	.947	.945	.945	.106	.061	.100	.057
CNN-Trans v2	.282	.268	.050	.045	.022	.010	.754	.752	.952	.950	.930	.930	.097	.050	.093	.059
Ratio of CNN and Transformer Layers in Hybrid Architectures																
CNN-Trans v1-4:1	**.333**	**.326**	.067	**.060**	**.046**	**.027**	**.752**	**.751**	**.948**	**.947**	**.945**	**.945**	**.106**	**.061**	**.100**	**.057**
CNN-Trans v1-3:2	.115	.091	**.069**	.057	.010	.001	.696	.694	.904	.901	.916	.914	.094	.036	.090	.041
CNN-Trans v1-2:3	.060	.032	.041	.021	.005	.000	.699	.697	.833	.828	.867	.866	.088	.030	.085	.031
CNN-Trans v1-1:4	.036	.013	.030	.008	.005	.000	.684	.674	.709	.687	.764	.760	.079	.018	.076	.019
CNN-Trans v2-4:1	**.282**	**.268**	.050	.045	.022	**.010**	**.754**	**.752**	**.952**	**.950**	**.930**	**.930**	**.097**	.050	**.093**	**.059**
CNN-Trans v2-3:2	.211	.192	**.061**	**.054**	**.024**	.009	.719	.713	.946	.944	.882	.880	.097	**.057**	.087	.051
CNN-Trans v2-2:3	.141	.124	.059	.046	.011	.002	.710	.707	.882	.881	.893	.892	.088	.050	.092	.044
CNN-Trans v2-1:4	.047	.033	.035	.019	.008	.001	.722	.720	.822	.814	.839	.837	.094	.047	.084	.041
New Patch Method for Transformers and GRUs																
Transformer	.085	.080	**.042**	.028	.005	.000	.795	.794	**.953**	**.952**	**.911**	**.910**	.069	.015	.057	.012
PatchTST	.060	.051	.038	**.029**	**.006**	**.001**	**.845**	**.838**	.906	.904	.766	.760	.050	.005	.050	.005
iTransformer	.097	.085	.029	.005	.005	.000	.730	.728	.729	.698	.780	.774	.064	.013	.061	.014
Ours	**.234**	**.226**	.033	.027	.005	.000	.799	.799	.922	.919	.843	.839	**.098**	**.046**	**.090**	**.066**
GRU	**.353**	**.343**	.025	.001	.040	.037	.774	.773	.146	.046	.136	.039	**.123**	**.091**	**.117**	**.086**
Ours	.351	.342	**.033**	**.026**	**.074**	**.068**	**.817**	**.813**	**.958**	**.956**	**.937**	**.936**	.105	.071	.091	.066

Table 3: **Macro- and micro-level studies for developing the practical design recipe.** Lower GFLOPs indicates lower computational cost. The best results are in **bold**.

Setting	GFLOPs \ Datasets	Du-IN		SEED-DV		ThingsEEG		FacesHouses		OpenMIIR-P		OpenMIIR-I		Chisco-R		Chisco-I	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Macro Study: Transformation of the Feature Map Number																	
Leap Approach	3996	.635	.630	**.064**	.058	**.216**	**.206**	.857	.856	.980	.980	.976	.976	.126	.109	**.115**	.093
Step Approach	**1287**	**.647**	**.641**	.063	**.058**	.184	.177	**.886**	**.885**	**.981**	**.980**	**.982**	**.982**	**.127**	**.113**	.114	**.095**
Macro Study: Transformation of the Feature Map Size																	
Pyramid Approach	1287	.647	.641	**.063**	**.058**	.184	.177	.886	.885	**.981**	**.980**	.982	.982	.127	.113	.114	.095
Pagoda Approach	**758**	**.701**	**.698**	.059	.053	**.204**	**.197**	**.911**	**.910**	.973	.973	**.986**	**.985**	**.128**	**.116**	**.116**	**.096**
Micro Study																	
Vanilla Conv	758	.701	.698	.059	.053	.204	.197	.911	.910	.973	.973	.986	.985	.128	**.116**	.116	.096
Separable Conv	492	.704	.699	.061	.051	.181	.176	.915	.915	.979	.979	.985	.985	.118	.107	.111	.098
Group Conv	**482**	**.707**	**.704**	**.069**	**.061**	**.207**	**.200**	**.920**	**.920**	**.983**	**.983**	**.990**	**.990**	**.129**	.112	**.120**	**.103**

Table 2 (upper) shows the performance of the 9 basic architectures discussed above. Overall, CNN-based models, which excel in capturing local patterns, outperform hybrid models on most neural decoding tasks. Meanwhile, GRU and Transformer-based models, which are better suited for modeling long-range dependencies, exhibit the lowest performance. This result aligns with the transient temporal dynamics of brain signals in neural decoding tasks, suggesting that short-range temporal information is more effective for neural decoding. We further investigate the performance gap between CNN-based and Transformer-based architectures from a representation perspective by

comparing layer-wise representations in NeuroSketch-Large and MedFormer (Appendix E.5). Among CNN-based models, CNN-2D captures local cross-channel patterns, whereas CNN-1D aggregates information across all input channels. The higher accuracy of CNN-2D suggests that explicitly modeling local cross-channel dependencies is beneficial for neural decoding. Surprisingly, the Transformer-based models perform poorly on certain datasets, particularly the ThingsEEG dataset, where their best accuracy of 0.6% is only marginally above the chance level of 0.5%. To understand the underlying reasons for these performance differences, we conduct further analyses from both temporal and spatial perspectives.

From a temporal perspective, we investigate how short- and long-term temporal information affects the model performance. We control the short- and long-term information through adjusting the ratio of CNN and Transformer layers in two hybrid models. From the results in Table 2 (middle), we observe a decline in performance as the proportion of short-term information decreases, demonstrating the effectiveness of short-range information for neural decoding.

From a spatial perspective, we extend the CNN-2D finding to GRUs and Transformers through a patching scheme for local channel information. The input is divided into sliding temporal windows (Figure 2(a)). Unlike PatchTST and iTransformer, which treat channels independently or focus on global channel information, the scheme concatenates channel and patch dimensions to preserve channel relationships within each window. This scheme improves Transformer accuracy on Du-IN from 8.5% to 23.4% (Table 2, lower). GRU accuracy on OpenMIIR perception and imagination rises from chance level to 95.8% and 93.7%, respectively, further supporting spatial locality in neural decoding.

From now on, we will use CNN-2D as our basic architecture.

2.3 Macro-Level Design: Latent Space Transformation

To address the macro optimization aspect of Question 2, we study the latent space transformation during the forward propagation. For CNN-2D, its latent space representation refers to $\mathbf{S} \in \mathbb{R}^{C \times H \times W}$, where C denotes the number of feature maps, H and W denote the height and width of feature maps, respectively. During forward propagation, the network increases the number of feature maps to capture higher-level information, while reducing the size of feature maps to alleviate redundancy and improve computational efficiency. Building on this general overview, we will investigate the transformation of the number (C) and the size (H and W) of feature maps separately.

Regarding the number of feature maps, the appropriate strategy to increase them is a critical aspect of neural network design, and this can be achieved through two widely adopted approaches: **step approach** and **leap approach**. As shown in Figure 2(b), the step approach gradually increases the number of feature maps, enabling the network to extract features from low to high levels systematically. In contrast, the leap approach rapidly increases the number of feature maps to the embedding dimension in the early stages, allowing the model to focus primarily on refining higher-level features.

The upper block of Table 3 compares the accuracy and computational cost of the step and leap approaches. Surprisingly, the step approach achieves highly competitive performance compared to the leap approach, while requiring 67.8% fewer FLOPs. On FacesHouses, the step approach achieves 88.6% accuracy with 1,287 GFLOPs, compared with 85.7% and 3,996 GFLOPs for the leap approach. This indicates that neural decoding does not rely solely on high-level neural representations, and that multi-scale feature extraction is often more effective and efficient.

From now on, we will use the step approach to transform the number of feature maps.

In the forward propagation process, the transformation of the feature map size is achieved through downsampling. Therefore, we further investigate how downsampling should be distributed throughout the network. Suppose downsampling is applied only in the final layers. In that case, earlier layers must process high-resolution feature maps, which significantly increases computational complexity and undermines the original purpose of downsampling to improve computational efficiency. Therefore, we study two common strategies: **pyramid approach** and **pagoda approach**. As shown in Figure 2(b), the pyramid approach distributes the downsampling process evenly across the entire network, gradually reducing the size of the feature maps. This pyramid-like structure enables the network to retain more detailed features at the cost of increased computational complexity. The

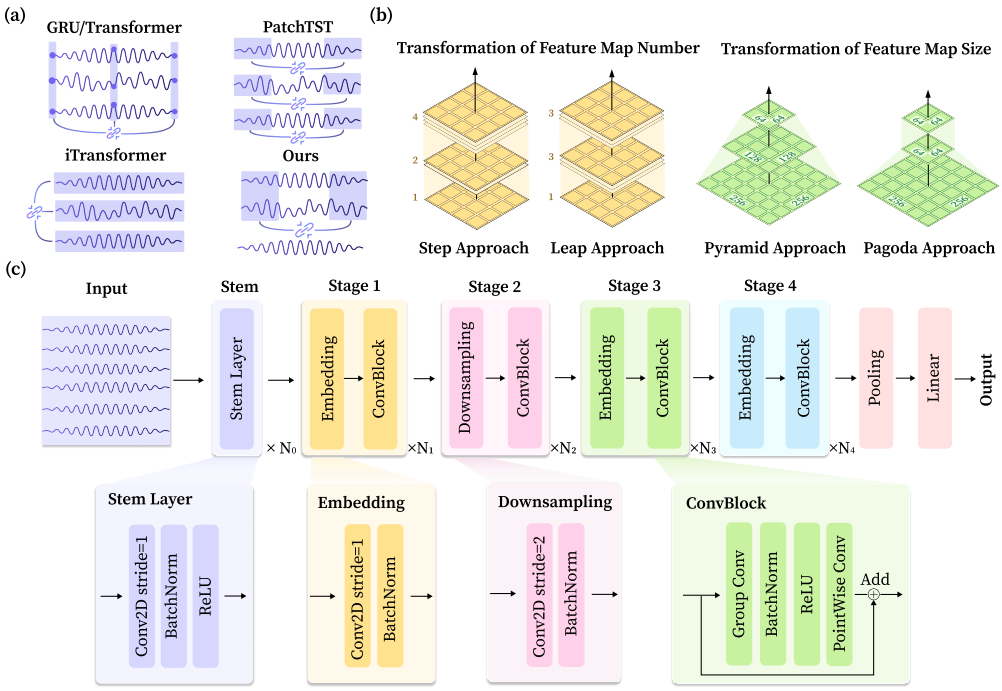


Figure 2: **(a) Comparison of patching schemes.** The vanilla GRU and Transformer use the values from all recording channels at one time step as a token; PatchTST uses a temporal window from a single channel; iTransformer uses an entire channel sequence. We also evaluate a patching scheme that jointly represents multiple time steps and recording channels. **(b) Comparison of different latent space transformation methods.** Regarding the number of feature maps, the step approach increases them gradually, whereas the leap approach increases them rapidly in the early stages. In terms of their size, the pyramid approach decreases them progressively, while the pagoda approach decreases them quickly in the early stages. **(c) The shared architecture of NeuroSketch-Base and NeuroSketch-Large.**

pagoda approach takes a more aggressive strategy by concentrating the downsampling module in the early stages of the network. This leads to a significant reduction in feature map size early on, resulting in lower computational costs. Table 3 (middle) compares the performance and computational cost between the pyramid and pagoda approach. The pagoda approach achieves slightly better accuracy than the pyramid approach, requiring 41.1% fewer FLOPs. Specifically, on the Du-IN dataset, the pagoda approach attains 70.1% accuracy, whereas the pyramid approach only reaches 64.7% with a higher computational cost. On ThingsEEG, the pagoda approach increases accuracy by 2.0 percentage points to 20.4%, exceeding CNN-1D's 20.2%. These results demonstrate that excessive extraction of detailed information from the original signal is unnecessary. Brain signals typically have low signal-to-noise ratios, and early downsampling allows the model to focus on the most relevant features while reducing noise impact, thereby enhancing both computational efficiency and performance.

From now on, we will use the pagoda approach to transform the size of feature maps.

2.4 Micro-Level Design: Convolutional Operations

To address the micro-level aspect of Q2, we compare convolutional operations. Let $\mathbf{X}_{\text{in}} \in \mathbb{R}^{C_{\text{in}} \times H_{\text{in}} \times W_{\text{in}}}$ be an input feature tensor. A standard convolution with C_{out} filters of size $U \times V$ produces an output tensor $\mathbf{X}_{\text{out}} \in \mathbb{R}^{C_{\text{out}} \times H_{\text{out}} \times W_{\text{out}}}$. Here, C_{in} and C_{out} denote the numbers of learned feature channels. The kernel tensor has shape $C_{\text{out}} \times C_{\text{in}} \times U \times V$, giving $C_{\text{out}} C_{\text{in}} UV$ kernel parameters when bias terms are excluded.

To explore more efficient alternatives to standard convolution, we consider grouped convolution (Ioannou et al., 2017) and separable convolution (Howard, 2017).

Grouped convolution. With G groups, each output channel connects to C_{in}/G input channels. The kernel tensor has shape $C_{\text{out}} \times (C_{\text{in}}/G) \times U \times V$, giving $C_{\text{out}}C_{\text{in}}UV/G$ kernel parameters, or $1/G$ of the standard convolution's kernel parameter count.

Separable convolution. Separable convolution combines depthwise spatial filtering with pointwise channel mixing. Depthwise convolution applies one filter per input channel ($G = C_{\text{in}}$), with kernels $\mathbf{F}_d \in \mathbb{R}^{C_{\text{in}} \times U \times V}$. Pointwise convolution mixes channels using 1×1 kernels $\mathbf{F}_p \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times 1 \times 1}$. Together, these operations use $C_{\text{in}}UV + C_{\text{in}}C_{\text{out}}$ kernel parameters, corresponding to $\frac{1}{C_{\text{out}}} + \frac{1}{UV}$ of the standard convolution's kernel parameter count.

The lower block of Table 3 compares the three convolutional variants. Grouped convolution achieves competitive accuracy at the lowest computational cost. On SEED-DV, standard convolution achieves 5.9% accuracy at 758 GFLOPs, whereas separable convolution reaches 6.1% with 35% fewer FLOPs. At a similar cost, grouped convolution further improves accuracy to 6.9%. These results support channel grouping as an effective way to aggregate local information while improving computational efficiency.

We will use group convolution as our core calculation method, completing the practical design recipe.

2.5 The Practical Design Recipe and Its Instantiations

Synthesizing these findings, we obtain NeuroSketch, a practical design recipe for neural decoding. We instantiate it as NeuroSketch-Base and NeuroSketch-Large, with 1.4M and 4.2M parameters, respectively. Both share the architecture in Figure 2(c), differing in stage widths and depths. A stem processes the reshaped 2D input, followed by four stages with gradual feature-map expansion and downsampling concentrated in the second stage. Each stage contains grouped convolutions, batch normalization, ReLU activations, and pointwise convolutions. GeM pooling (Berman et al., 2019) aggregates the two feature-map dimensions, and a linear classifier produces the predictions. Appendix C.2 provides implementation details.

3 Experiments

To assess the practical value of the recipe, we evaluate NeuroSketch-Base and NeuroSketch-Large through 2,742 training runs. Section 3.1 describes the evaluation setup, and Section 3.2 reports results for speech, visual, and auditory decoding. Section 3.3 presents a case study of the spatiotemporal patterns captured by NeuroSketch-Large.

3.1 Experimental Setup

The evaluation covers the eight tasks introduced in Section 2 and includes NeuroSketch-Base (1.4M parameters) and NeuroSketch-Large (4.2M parameters). Appendix C.2 provides their architectures and implementation details. The comparison includes ten baselines spanning time-series modeling, computer vision, and neural decoding. ModernTCN (Luo & Wang, 2024) and MedFormer (Wang et al., 2024) represent time-series models, while ConvFormer and CAFormer (Yu et al., 2023) serve as vision backbones. Neural decoding baselines comprise DeepConvNet (Schirrmeister et al., 2017), EEGNet (Lawhern et al., 2018), EEG-Conformer (Song et al., 2022), and SPaRCNet (Jing et al., 2023). We also evaluate the iEEG foundation model seegnificant (Mentzelopoulos et al., 2024) and the EEG foundation model CBraMod (Wang et al., 2025a). Both encoders are initialized with official pretrained weights and jointly fine-tuned with newly initialized classification heads. Further details on the baselines and evaluation protocol are provided in Appendices C.3 and D, respectively.

3.2 Results

Main results. As shown in Table 4, NeuroSketch outperforms all selected baselines, underscoring its strong capacity to model brain signals across diverse decoding scenarios. In addition, we note that NeuroSketch-Base performs similarly to NeuroSketch-Large on all datasets except Du-IN, where NeuroSketch-Large outperforms NeuroSketch-Base by a substantial margin. We further examine this scaling phenomenon in Appendix E.4. Complementary analyses examine kernel size and convolution group count (Appendix E.3), as well as pooling choices and residual connections

Table 4: **Average classification accuracy** (mean ± std across three folds) for models on multiple neural decoding datasets. The best results are in **bold**. We use — to denote entries that are not applicable or not evaluated. Due to substantial differences between iEEG and EEG, we evaluate each pretrained model only on the modality it was pretrained on, the corresponding cross-modality cells are therefore marked with —. In addition, CBraMod uses a patch size of 200, whereas *ThingsEEG* has an input length of 100, so CBraMod cannot be run on *ThingsEEG* and is marked —.

Models \ Datasets	Speech Decoding			Visual Decoding			Auditory Decoding	
	Du-IN	Chisco-R	Chisco-I	SEED-DV	Things-EEG	Faces-Houses	Open-MIIR-P	Open-MIIR-I
ConvFormer	.394±.005	.086±.001	.080±.004	.024±.000	.151±.003	.820±.013	.975±.005	.976±.002
CAFormer	.221±.003	.081±.003	.076±.002	.025±.001	.160±.009	.800±.012	.976±.006	.977±.002
ModernTCN	.339±.005	.079±.003	.074±.002	.040±.001	.123±.003	.835±.006	.964±.010	.935±.002
MedFormer	.212±.006	.089±.003	.085±.001	.025±.000	.045±.002	.756±.011	.948±.001	.941±.001
DeepConvNet	.053±.002	.053±.004	.052±.001	.029±.001	.091±.001	.822±.005	.896±.009	.789±.005
EEGNet	.060±.003	.071±.002	.082±.002	.046±.002	.070±.000	.830±.006	.915±.010	.852±.006
EEG-Conformer	.205±.003	.095±.002	.087±.002	.033±.002	.077±.002	.762±.001	.962±.013	.930±.002
SPaRCNet	.026±.001	.005±.000	.005±.000	.026±.001	.189±.001	.805±.007	.969±.007	.948±.005
seegnificant	.053±.002	—	—	—	—	.831±.006	—	—
CBraMod	—	.084±.002	.077±.001	.026±.002	—	—	.975±.011	.933±.008
NeuroSketch-Base	.472±.006	.111±.001	**.103±.002**	.045±.002	**.196±.002**	.879±.008	.981±.010	.970±.002
NeuroSketch-Large	**.652±.002**	**.116±.001**	.095±.005	**.047±.003**	.177±.001	**.879±.007**	**.982±.009**	**.977±.002**

(Appendix E.2). Further evaluations compare NeuroSketch with additional foundation models (Appendix F.1), assess cross-subject generalization (Appendix F.2), and examine robustness to reduced training data (Appendix F.3).

Speech decoding. On Du-IN, NeuroSketch-Large achieves 65.2% accuracy, compared with 39.4% for the strongest baseline, ConvFormer. By comparison, the iEEG foundation model seegnificant reaches 5.3%, which may reflect the limited capacity of its single-layer Transformer architecture on this task. For semantic classification during sentence reading and imagination, our model instantiations also lead on both Chisco-R and Chisco-I despite the low overall accuracies. On Chisco-R, the Large variant attains 11.6% accuracy versus 9.5% for EEG-Conformer, a relative improvement of 22.1%. On Chisco-I, the Base variant reaches 10.3%, exceeding EEG-Conformer's 8.7%.

Visual decoding. For static image decoding on FacesHouses, NeuroSketch-Large achieves 87.9% accuracy, compared with 83.5% for the strongest baseline, ModernTCN. This advantage also holds on the more challenging ThingsEEG dataset, where NeuroSketch-Base reaches 19.6%, exceeding SPaRCNet's 18.9%. For video decoding on SEED-DV, NeuroSketch-Large achieves 4.7% accuracy, while most baselines perform near chance level (2.5%).

Auditory decoding. NeuroSketch achieves the best or tied-best accuracy among the evaluated models on both auditory perception and imagery tasks. For perception (OpenMIIR-P), NeuroSketch-Large reaches 98.2%, exceeding the strongest brain-domain baseline, CBraMod (97.5%). For imagery (OpenMIIR-I), its accuracy of 97.7% matches CAFormer and slightly exceeds ConvFormer (97.6%). This comparable imagery performance is achieved with approximately one-fifth as many parameters as the two vision baselines.

3.3 Case Study

To examine the spatiotemporal features captured by NeuroSketch, we conducted a case study using Subjects 02 and 11 from Du-IN. For this analysis, NeuroSketch-Large was trained using all 115 channels from Subject 02 and all 90 channels from Subject 11. Score-CAM (Wang et al., 2020) was then applied to the trained model to visualize the spatial and temporal regions associated with its predictions.

Score-CAM estimates feature-map importance without gradients and projects the resulting activations onto the input space. This procedure yields a saliency map for each sample, highlighting the spatiotemporal regions associated with the predicted class. Averaging these maps across samples

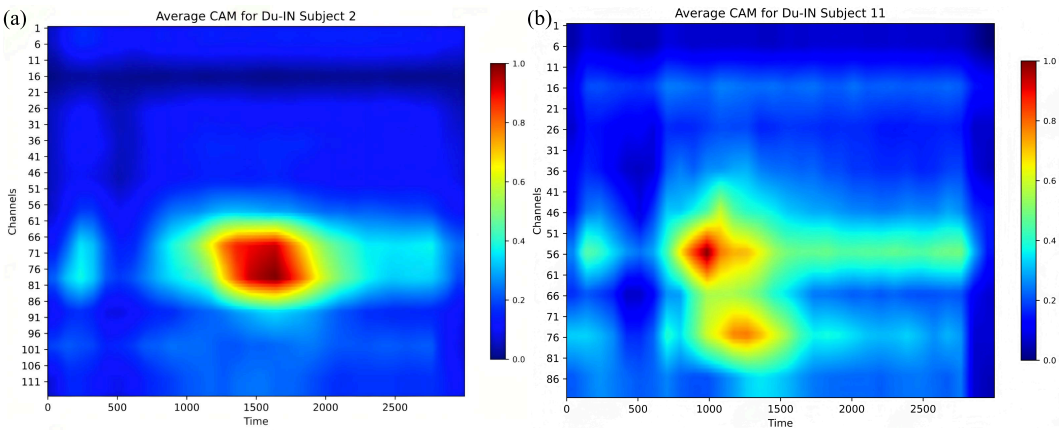


Figure 3: **Score-CAM saliency maps on Du-IN. (a)** Subject 02. **(b)** Subject 11. Maps are averaged across samples for each subject. Warmer colors indicate greater importance to model predictions. Saliency is concentrated in specific cortical regions and mid-trial time windows, highlighting spatial and temporal locality.

from each subject produces the subject-level maps shown in Figure 3. These aggregated maps provide the basis for identifying spatial and temporal regions of interest (ROIs).

To characterize the spatial patterns in these maps, we averaged each subject-level map over time to obtain a saliency score for each channel. The channels with the highest scores cluster around the ventral sensorimotor cortex (vSMC) and bilateral superior temporal gyrus (STG). These regions have established roles in speech processing, with vSMC coordinating articulatory movements and STG processing auditory feedback (Bouchard et al., 2013; Hickok & Poeppel, 2007; Chartier et al., 2018). This correspondence supports the neurophysiological interpretability of the spatial patterns captured by NeuroSketch. Along the temporal dimension, the saliency maps show peak activations during the middle of each trial, with weaker contributions from earlier and later segments. These peaks occur at 800–1600 ms for Subject 11 and 1200–1800 ms for Subject 02. Such temporal concentration is consistent with the transient nature of task-relevant neural activity.

Together, these observations are consistent with the temporal and spatial motivations underlying the practical design recipe.

4 Conclusion and Future Work

Conclusion. In this paper, we develop NeuroSketch, a practical design recipe for neural decoding, through a basic architecture study followed by macro- and micro-level optimization. The basic architecture study favors CNN-2D for modeling short-range temporal dynamics and local cross-channel dependencies. Building on this backbone, macro-level studies of latent space transformations support gradual feature-map expansion and early downsampling, while micro-level studies favor grouped convolutions. To evaluate the resulting recipe, we assess NeuroSketch-Base and NeuroSketch-Large across eight tasks spanning visual, auditory, and speech modalities and EEG, SEEG, and ECoG signals. This evaluation supports the practical value of the recipe as a guide for neural decoding model design.

Limitations and future work. The current practical design recipe is developed and evaluated in neural decoding settings. Given the favorable scaling behavior observed in Appendix E.4, we hypothesize that scaling both data and model capacity will yield further gains. Accordingly, we will explore large-scale pretraining strategies guided by the NeuroSketch recipe in the future. We also plan to extend the evaluation to a broader range of neural decoding tasks to assess the recipe's generalizability.

AI USE STATEMENT

In preparing this manuscript, we made limited use of a large language model (LLM) to assist with polishing the writing for grammar and clarity. In addition, the LLM was occasionally employed for small-scale code completion, such as generating boilerplate functions or suggesting minor syntax corrections. The authors developed the core ideas and experimental designs, conducted the analyses, and verified the conclusions.

ETHICS STATEMENT

All datasets used in this study are publicly available. The corresponding dataset links are provided in Appendix B.1. Consequently, our work does not involve personally sensitive information and does not pose apparent ethical concerns.

REPRODUCIBILITY STATEMENT

We place great importance on ensuring the reproducibility of our work. To this end, dataset preprocessing is described in Appendix B.2, and model implementations and configurations are detailed in Appendix C. Appendix D documents data splitting, augmentation, model and training hyperparameters, foundation-model fine-tuning, and computational resources. The complete experimental results are available at `https://anonymous.4open.science/r/NeuroSketch_Results-BA5B`. Our code and scripts are available at `https://anonymous.4open.science/r/NeuroSketch-anonymous-A5E4/`.

REFERENCES

Hajar Ahmadieh, Farnaz Gassemi, and Mohammad Hasan Moradi. Visual image reconstruction based on EEG signals using a generative adversarial and deep fuzzy neural network. *Biomedical Signal Processing and Control*, 87:105497, 2024.

Hamdi Altaheri, Ghulam Muhammad, Mansour Alsulaiman, Syed Umar Amin, Ghadir Ali Altuwaijri, Wadood Abdul, Mohamed A. Bencherif, and Mohammed Faisal. Deep learning techniques for classification of electroencephalogram (EEG) motor imagery (MI) signals: A review. *Neural Computing and Applications*, 35(20):14681–14722, July 2023. ISSN 0941-0643, 1433-3058. doi: 10.1007/s00521-021-06352-5.

Miguel Angrick, Christian Herff, Emily Mugler, Matthew C. Tate, Marc W. Slutzky, Dean J. Krusienski, and Tanja Schultz. Speech synthesis from ECoG using densely connected 3D convolutional neural networks. *Journal of Neural Engineering*, 16(3):036019, 2019.

Maxim Berman, Hervé Jégou, Andrea Vedaldi, Iasonas Kokkinos, and Matthijs Douze. MultiGrain: a unified image embedding for classes and instances. *arXiv preprint arXiv:1902.05509*, 2019.

Kristofer E Bouchard, Nima Mesgarani, Keith Johnson, and Edward F Chang. Functional organization of human sensorimotor cortex for speech articulation. *Nature*, 495(7441):327–332, 2013.

Daniel A. Butts, Chong Weng, Jianzhong Jin, Chun-I. Yeh, Nicholas A. Lesica, Jose-Manuel Alonso, and Garrett B. Stanley. Temporal precision in the neural code and the timescales of natural vision. *Nature*, 449(7158):92–95, 2007.

Ahmad Chaddad, Yihang Wu, Reem Kateb, and Ahmed Bouridane. Electroencephalography signal processing: A comprehensive review and analysis of methods and techniques. *Sensors*, 23(14): 6434, 2023.

Josh Chartier, Gopala K Anumanchipalli, Keith Johnson, and Edward F Chang. Encoding of articulatory kinematic trajectories in human speech sensorimotor cortex. *Neuron*, 98(5):1042–1054, 2018.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555, December 2014. URL `https://arxiv.org/abs/1412.3555`.

Aline W. De Borst, Giancarlo Valente, Iiro P. Jääskeläinen, and Pia Tikka. Brain-based decoding of mentally imagined film clips and sounds reveals experience-based information patterns in film professionals. *NeuroImage*, 129:428–438, 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE, 2009.

Berkay Döner, Thorir Mar Ingolfsson, Luca Benini, and Yawei Li. LUNA: Efficient and topology-agnostic foundation model for EEG signal analysis. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 70682–70708. Curran Associates, Inc., 2025. doi: 10.52202/085713-2377. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/66969a9e6bd7a26dfeccea7227178ca7-Paper-Conference.pdf.

Chen Feng, Lu Cao, Di Wu, En Zhang, Ting Wang, Xiaowei Jiang, Jinbo Chen, Hui Wu, Siyu Lin, Qiming Hou, Junming Zhu, Jie Yang, Mohamad Sawan, and Yue Zhang. Acoustic inspired brain-to-sentence decoder for logosyllabic language. *Cyborg and Bionic Systems*, 6:0257, 2025. doi: 10.34133/cbsystems.0257.

Guy Gaziv, Roman Beliy, Niv Granot, Assaf Hoogi, Francesca Strappini, Tal Golan, and Michal Irani. Self-supervised natural image reconstruction and large-scale semantic classification from brain activity. *NeuroImage*, 254:119121, 2022.

Alessandro T Gifford, Kshitij Dwivedi, Gemma Roig, and Radoslaw M Cichy. A large and rich EEG dataset for modeling human visual object recognition. *NeuroImage*, 264:119754, 2022.

Kalanit Grill-Spector and Rafael Malach. The human visual cortex. *Annual Review of Neuroscience*, 27(1):649–677, July 2004. ISSN 0147-006X, 1545-4126. doi: 10.1146/annurev.neuro.27.070203.144220.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Gregory Hickok and David Poeppel. The cortical organization of speech processing. *Nature Reviews Neuroscience*, 8(5):393–402, 2007.

Andrew G. Howard. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Yani Ioannou, Duncan Robertson, Roberto Cipolla, and Antonio Criminisi. Deep roots: Improving CNN efficiency with hierarchical filter groups. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1231–1240, 2017.

Jin Jing, Wendong Ge, Shenda Hong, Marta Bento Fernandes, Zhen Lin, Chaoqi Yang, Sungtae An, Aaron F. Struck, Aline Herlopian, Ioannis Karakis, et al. Development of Expert-Level Classification of Seizures and Rhythmic and Periodic Patterns During EEG Interpretation. *Neurology*, 100 (17), April 2023. ISSN 0028-3878, 1526-632X. doi: 10.1212/WNL.0000000000207127.

Sehoon Kim, Amir Gholami, Albert Shaw, Nicholas Lee, Karttikeya Mangalam, Jitendra Malik, Michael W. Mahoney, and Kurt Keutzer. Squeezeformer: An efficient transformer for automatic speech recognition. *Advances in Neural Information Processing Systems*, 35:9361–9373, 2022.

Jean-Rémi King and Stanislas Dehaene. Characterizing the dynamics of mental representations: The temporal generalization method. *Trends in Cognitive Sciences*, 18(4):203–210, 2014.

Vernon J. Lawhern, Amelia J. Solon, Nicholas R. Waytowich, Stephen M. Gordon, Chou P. Hung, and Brent J. Lance. EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces. *Journal of Neural Engineering*, 15(5):056013, 2018.

Matthew K. Leonard, Laura Gwilliams, Kristin K. Sellers, Jason E. Chung, Duo Xu, Gavin Mischler, Nima Mesgarani, Marleen Welkenhuysen, Barundeb Dutta, and Edward F. Chang. Large-scale single-neuron speech sound encoding across the depth of human cortex. *Nature*, 626(7999): 593–602, 2024.

Zida Liang, Jiayi Zhu, and Weiqiang Sun. Why attention fails: The degeneration of transformers into MLPs in time series forecasting. *arXiv preprint arXiv:2509.20942*, 2025.

Xuan-Hao Liu, Yan-Kai Liu, Yansen Wang, Kan Ren, Hanwen Shi, Zilong Wang, Dongsheng Li, Bao-Liang Lu, and Wei-Long Zheng. EEG2video: Towards decoding dynamic visual perception from EEG signals. *Advances in Neural Information Processing Systems*, 37:72245–72273, 2024a.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, pp. 11116–11140, 2024b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022.

Donghao Luo and Xue Wang. ModernTCN: A modern pure convolution structure for general time series analysis. In *The Twelfth International Conference on Learning Representations*, pp. 1–43, 2024.

Jingying Ma, Feng Wu, Qika Lin, Yucheng Xing, Chenyu Liu, Ziyu Jia, and Mengling Feng. CodeBrain: Bridging decoupled tokenizer and multi-scale architecture for EEG foundation model. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, pp. 83526–83574, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/86d1ab582afb247ccaa84bec4a7e24f7-Paper-Conference.pdf.

Keyvan Mahjoory, Andreas Bahmer, and Molly J. Henry. Convolutional neural networks can identify brain interactions involved in decoding spatial auditory attention. *PLOS Computational Biology*, 20(8):e1012376, 2024.

Joseph G. Makin, David A. Moses, and Edward F. Chang. Machine translation of cortical activity to text with an encoder–decoder framework. *Nature Neuroscience*, 23(4):575–582, 2020.

Georgios Mentzelopoulos, Evangelos Chatzipantazis, Ashwin G Ramayya, Michelle J Hedlund, Vivek P Buch, Kostas Daniilidis, Konrad P Kording, and Flavia Vitale. Neural decoding from stereotactic EEG: accounting for electrode variability across subjects. *Advances in Neural Information Processing Systems*, 37:108600–108624, 2024.

Sean L. Metzger, Kaylo T. Littlejohn, Alexander B. Silva, David A. Moses, Margaret P. Seaton, Ran Wang, Maximilian E. Dougherty, Jessie R. Liu, Peter Wu, and Michael A. Berger. A high-performance neuroprosthesis for speech decoding and avatar control. *Nature*, 620(7976): 1037–1046, 2023.

Kai J. Miller. A library of human electrocorticographic data and analyses. *Nature Human Behaviour*, 3(11):1225–1235, 2019.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023.

Sara Maria Pani, Luca Saba, and Matteo Fraschini. Clinical applications of EEG power spectra aperiodic component analysis: A mini-review. *Clinical Neurophysiology*, 143:1–13, 2022.

Janis Peksa and Dmytro Mamchur. State-of-the-art on brain-computer interface technology. *Sensors*, 23(13):6001, 2023.

Huy Phan and Kaare Mikkelsen. Automatic sleep staging of EEG signals: Recent development, challenges, and future directions. *Physiological Measurement*, 43(4):04TR01, 2022.

Timothée Proix, Jaime Delgado Saa, Andy Christen, Stephanie Martin, Brian N. Pasley, Robert T. Knight, Xing Tian, David Poeppel, Werner K. Doyle, and Orrin Devinsky. Imagined speech can be decoded from low-and cross-frequency intracranial EEG features. *Nature Communications*, 13(1): 48, 2022.

Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10428–10436, 2020.

Md Mustafizur Rahman, Ajay Krishno Sarkar, Md Amzad Hossain, Md Selim Hossain, Md Rabiul Islam, Md Biplob Hossain, Julian MW Quinn, and Mohammad Ali Moni. Recognition of human emotions using EEG signals: A review. *Computers in Biology and Medicine*, 136:104696, 2021.

Mehrnoosh Sadat Safi and Seyed Mohammad Mehdi Safi. Early detection of Alzheimer's disease from EEG signals using Hjorth parameters. *Biomedical Signal Processing and Control*, 65:102338, 2021.

Sani Saminu, Guizhi Xu, Shuai Zhang, Isselmou Ab El Kader, Hajara Abdulkarim Aliyu, Adamu Halilu Jabire, Yusuf Kola Ahmed, and Mohammed Jajere Adamu. Applications of artificial intelligence in automatic detection of epileptic seizures using EEG signals: A review. In *Artificial Intelligence and Applications*, volume 1, pp. 11–25, 2023.

Gerwin Schalk. EEG Motor Movement/Imagery Dataset. *PhysioNet*, 2009. doi: 10.13026/C28G6P. URL `https://doi.org/10.13026/C28G6P`. Version 1.0.0.

Robin Tibor Schirrmeister, Jost Tobias Springenberg, Lukas Dominique Josef Fiederer, Martin Glasstetter, Katharina Eggensperger, Michael Tangermann, Frank Hutter, Wolfram Burgard, and Tonio Ball. Deep learning with convolutional neural networks for EEG decoding and visualization. *Human Brain Mapping*, 38(11):5391–5420, 2017.

Pradeep Shenoy, Kai J. Miller, Jeffrey G. Ojemann, and Rajesh PN Rao. Generalized features for electrocorticographic BCIs. *IEEE Transactions on Biomedical Engineering*, 55(1):273–280, 2007.

Prajwal Singh, Pankaj Pandey, Krishna Miyapuram, and Shanmuganathan Raman. EEG2IMAGE: Image reconstruction from EEG brain signals. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Dhananjay Sonawane, Krishna Prasad Miyapuram, Bharatesh Rs, and Derek J. Lomas. GuessTheMusic: Song Identification from Electroencephalography response. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, pp. 154–162, Bangalore India, January 2021. ACM. ISBN 978-1-4503-8817-7. doi: 10.1145/3430984.3431023.

Yonghao Song, Qingqing Zheng, Bingchuan Liu, and Xiaorong Gao. EEG conformer: Convolutional transformer for EEG decoding and visualization. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 31:710–719, 2022.

Carl E. Stafstrom and Lionel Carmant. Seizures and epilepsy: An overview for neuroscientists. *Cold Spring Harbor Perspectives in Medicine*, 5(6):a022426, 2015.

Sebastian Stober, Avital Sternin, Adrian M. Owen, and Jessica A. Grahn. Towards music imagery information retrieval: Introducing the OpenMIIR dataset of EEG recordings from music perception and imagination. In *ISMIR*, pp. 763–769, 2015.

J. Talairach. Approche nouvelle de la neurochirurgie de l'epilepsie. Methodologie sterotaxique et resultats therapeutiques. *Neurochirurgie*, 20:1–240, 1974.

Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Michal Teplan. Fundamentals of EEG measurement. *Measurement Science Review*, 2(2):1–11, 2002.

Marcel A. J. Van Gerven, Katja Seeliger, Umut Güçlü, and Yağmur Güçlütürk. Current Advances in Neural Decoding. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller (eds.), *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, volume 11700, pp. 379–394. Springer International Publishing, Cham, 2019. ISBN 978-3-030-28953-9 978-3-030-28954-6. doi: 10.1007/978-3-030-28954-6_21.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 24–25, 2020.

Jiquan Wang, Sha Zhao, Zhiling Luo, Yangxuan Zhou, Haiteng Jiang, Shijian Li, Tao Li, and Gang Pan. CBraMod: A criss-cross brain foundation model for EEG decoding. In *International Conference on Learning Representations*, 2025a.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. TimeMixer++: A general time series pattern machine for universal predictive analysis. In *International Conference on Learning Representations*, 2025b.

Yihe Wang, Nan Huang, Taida Li, Yujun Yan, and Xiang Zhang. Medformer: A multi-granularity patching transformer for medical time-series classification. In *Advances in Neural Information Processing Systems*, volume 37, pp. 36314–36341. Curran Associates, Inc., 2024. doi: 10.52202/079017-1145.

Haiguang Wen, Junxing Shi, Yizhen Zhang, Kun-Han Lu, Jiayue Cao, and Zhongming Liu. Neural encoding and decoding with deep learning for dynamic natural vision. *Cerebral Cortex*, 28(12): 4136–4160, 2018.

Francis R. Willett, Erin M. Kunz, Chaofei Fan, Donald T. Avansino, Guy H. Wilson, Eun Young Choi, Foram Kamdar, Matthew F. Glasser, Leigh R. Hochberg, and Shaul Druckmann. A high-performance speech neuroprosthesis. *Nature*, 620(7976):1031–1036, 2023.

Guy H. Wilson, Sergey D. Stavisky, Francis R. Willett, Donald T. Avansino, Jessica N. Kelemen, Leigh R. Hochberg, Jaimie M. Henderson, Shaul Druckmann, and Krishna V. Shenoy. Decoding spoken English from intracortical electrode arrays in dorsal precentral gyrus. *Journal of Neural Engineering*, 17(6):066007, 2020.

Benjamin Wittevrongel, Elvira Khachatryan, Evelien Carrette, Paul Boon, Alfred Meurs, Dirk Van Roost, and Marc M. Van Hulle. High-gamma oscillations precede visual steady-state responses: A human electrocorticography study. *Human Brain Mapping*, 41(18):5341–5355, December 2020. ISSN 1065-9471, 1097-0193. doi: 10.1002/hbm.25196.

Annan Yu, Danielle C Maddix, Boran Han, Xiyuan Zhang, Abdul Fatir Ansari, Oleksandr Shchur, Christos Faloutsos, Andrew Gordon Wilson, Michael W Mahoney, and Yuyang Wang. Understanding transformers for time series: Rank structure, flow-of-ranks, and compressibility. *arXiv preprint arXiv:2510.03358*, 2025.

Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. MetaFormer is actually what you need for vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10819–10829, 2022.

Weihao Yu, Chenyang Si, Pan Zhou, Mi Luo, Yichen Zhou, Jiashi Feng, Shuicheng Yan, and Xinchao Wang. MetaFormer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(2):896–912, 2023.

Xiang Zhang, Lina Yao, Xianzhi Wang, Jessica Monaghan, David Mcalpine, and Yu Zhang. A survey on deep learning-based non-invasive brain signals: Recent advances and new frontiers. *Journal of Neural Engineering*, 18(3):031002, 2021.

Zihan Zhang, Xiao Ding, Yu Bao, Yi Zhao, Xia Liang, Bing Qin, and Ting Liu. Chisco: An EEG-based BCI dataset for decoding of imagined speech. *Scientific Data*, 11(1):1265, 2024.

Hui Zheng, Haiteng Wang, Weibang Jiang, Zhongtao Chen, Li He, Peiyang Lin, Penghu Wei, Guoguang Zhao, and Yunzhe Liu. Du-IN: Discrete units-guided mask modeling for decoding speech from Intracranial Neural signals. *Advances in Neural Information Processing Systems*, 37: 79996–80033, 2024.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021.

Yuchen Zhou, Jiamin Wu, Zichen Ren, Zhouheng Yao, Weiheng Lu, Kunyu Peng, Qihao Zheng, Chunfeng Song, Wanli Ouyang, and Chao Gou. CSBrain: A cross-scale spatiotemporal brain foundation model for EEG decoding. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 87150–87195. Curran Associates, Inc., 2025. doi: 10.52202/085713-2915. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/7e199ad8ae40eb19b2980f61f659cb07-Paper-Conference.pdf.

APPENDIX CONTENTS

A Related Work 18

B Datasets and Preprocessing 18

B.1 Dataset Description 18

B.2 Data Preprocessing 19

C Model Architectures and Configurations 20

C.1 Architectures in the Roadmap Study 20

C.2 NeuroSketch Implementation 20

C.3 Baseline Architectures and Configurations 21

D Training and Evaluation Protocols 23

D.1 Data Splitting and Evaluation 23

D.2 Data Augmentation 24

D.3 Training Hyperparameters 24

D.4 Compute Resources 25

E Recipe and Model Analyses 25

E.1 Revisiting the Recipe After Scaling Down 25

E.2 Component-wise Ablation Study 26

E.3 Sensitivity to Kernel Size and Group Count 26

E.4 Scaling Behavior 27

E.5 Representation Analysis of CNN-2D and Transformer-Based Architectures 27

F Extended Evaluation 30

F.1 Additional Foundation-Model Evaluation 30

F.2 Cross-Subject Evaluation 30

F.3 Robustness to Reduced Training Data 31

APPENDIX OVERVIEW

The appendices provide implementation details and supplementary analyses for the experiments reported in the main text.

Appendix A: Related Work. We review prior work on visual, auditory, and speech decoding, together with studies of systematic neural architecture design.

Appendix B: Datasets and Preprocessing. Section B.1 describes the recording settings and experimental paradigms of the six datasets used in the roadmap study and main evaluation. Section B.2 details channel selection, sampling rates, input windows, and classification targets for each task.

Appendix C: Model Architectures and Configurations. Section C.1 specifies the architectures and design alternatives evaluated in the roadmap study. Section C.2 describes the input representation and architecture of NeuroSketch, including the Base and Large configurations. Section C.3 documents the baseline architectures, implementation settings, and parameter counts.

Appendix D: Training and Evaluation Protocols. Section D.1 specifies data partitions, checkpoint selection, and the aggregation of evaluation metrics. Sections D.2 and D.3 detail data augmentation, optimization settings, and fine-tuning procedures for the main foundation-model baselines. Section D.4 reports the hardware, number of training runs, and average training times.

Appendix E: Recipe and Model Analyses. Section E.1 revisits the architectural choices in the recipe after scaling down. Section E.2 evaluates pooling choices and residual connections through component-wise ablations. Section E.3 examines sensitivity to kernel size and convolution group count. Section E.4 investigates the scaling behavior of NeuroSketch across Du-IN participants. Section E.5 compares layer-wise representations in NeuroSketch-Large and MedFormer.

Appendix F: Extended Evaluation. Section F.1 reports comparisons with additional pretrained models. Sections F.2 and F.3 evaluate generalization to unseen participants and performance with fewer labeled training examples.

A Related Work

Neural decoding. Neural decoding infers sensory experiences and cognitive states from recorded brain activity. Auditory decoding includes perception, which concerns responses to external sounds (Stober et al., 2015; Sonawane et al., 2021), and imagery, which concerns internally generated sounds (Stober et al., 2015; De Borst et al., 2016). Speech decoding targets linguistic information at different levels, including syllables (Feng et al., 2025), words (Zheng et al., 2024), and sentences (Zhang et al., 2024). Visual decoding aims to infer visual stimuli from neural activity. Reconstruction methods recover static images (Gaziv et al., 2022; Ahmadieh et al., 2024; Singh et al., 2023) or dynamic videos (Wen et al., 2018; Liu et al., 2024a).

Neural architecture design. Recent advances in neural architecture design have diverged into two directions: automated neural architecture search (NAS) and manual design driven by experience. Tan & Le (2019) utilize NAS to scale all dimensions of depth, width, and resolution through an effective compound coefficient, leading to a family of models known as EfficientNets. Radosavovic et al. (2020) further integrate NAS with manual design principles to explore the design space, resulting in the development of RegNet. Meanwhile, manual architectural innovations have reinvigorated traditional components. Yu et al. (2022) reveal that the general Transformer architecture, termed MetaFormer rather than any specific token mixer (e.g., attention), is essential for achieving high performance. Liu et al. (2022) modernize a standard CNN towards the design of Swin Transformer (Liu et al., 2021) through macro- and micro-level optimizations, proposing a family of pure CNNs called ConvNeXt.

B Datasets and Preprocessing

B.1 Dataset Description

The evaluation uses six datasets covering eight neural decoding tasks. The dataset descriptions below summarize their recording settings and experimental paradigms.

Du-IN (Zheng et al., 2024) records SEEG signals during spoken Mandarin word production. Each trial displays a word in white for 0.5 seconds, followed by a 2-second green cue prompting articulation. The word then disappears and a fixation cross appears. The dataset includes 12 participants, with 7–13 SEEG electrodes providing 72–158 recording channels per participant. Each participant reads 61 Chinese words, repeated 50 times each, while SEEG and audio are recorded simultaneously. Each participant contributes approximately 15 hours of data sampled at 2000 Hz, including around 3 hours of task-related activity. These segments yield approximately 3000 three-second trials and are subsequently downsampled to 1000 Hz. The dataset is available under CC BY 4.0 at `https://huggingface.co/datasets/liulab-repository/Du-IN`.

Chisco (Zhang et al., 2024) records EEG during silent reading and imagined speech. Our evaluation includes five participants in both Chisco-R and Chisco-I, with recordings acquired using a 125-channel system at 1000 Hz. The experiment comprises 45 blocks of 150 trials. Each trial includes 5 seconds of silent reading and 3.3 seconds of imagined speech, with continuous EEG recording. The stimuli comprise 6,681 Chinese sentences of 6–15 characters from 39 semantic categories. Ten trials per block are randomly selected for verbal recall. A recall differing from the original sentence by more than four characters is marked incorrect. Two or more errors trigger a rest period followed by repetition of the block. The dataset is available under CC0 at `https://openneuro.org/datasets/ds005170/versions/1.1.2`.

FacesHouses (Miller, 2019) records ECoG during the viewing of static images. The dataset includes 14 epilepsy patients with subdural electrode strips over the inferior temporal cortex. Participants view grayscale faces and houses in randomized order while ECoG is recorded at 1000 Hz. Each experiment comprises three sessions, each containing 50 face images and 50 house images. Images appear for 400 ms, separated by 400-ms blank intervals. The dataset is available under CC BY-SA at `https://purl.stanford.edu/zk881ps0522`.

ThingsEEG (Gifford et al., 2022) records EEG responses to images presented in a rapid serial visual presentation (RSVP) paradigm. Ten healthy participants were recorded with a 64-channel system at 1000 Hz. Each training image is presented four times, and each test image 80 times. Stimuli appear at a 200-ms stimulus onset asynchrony, with a target-detection task used to maintain engagement.

Our decoding evaluation uses the 200-image test partition, as detailed in Appendix B.2. The dataset is available under CC BY 4.0 at https://osf.io/hd6zk/.

SEED-DV (Liu et al., 2024a) records EEG at 1000 Hz from 20 participants viewing video clips. The stimuli comprise 1,400 two-second clips covering 40 semantic concepts, with 35 clips per concept. Each participant completes seven blocks, separated by rest periods. Each block includes all 40 concepts in randomized order, with five clips per concept. Participants receive the target concept before viewing its associated clips. Access requires an application at https://bcmi.sjtu.edu.cn/ApplicationForm/apply_form/.

OpenMIIR (Stober et al., 2015) records EEG during music perception and imagery. The dataset includes 10 participants, eight with formal musical training, recorded with a 64-channel system at 512 Hz. The experiment comprises two sessions of five trials each. Twelve musical stimuli are presented in randomized order under four conditions: perception with auditory cues, imagery with cues, imagery without cues, and imagery without cues with feedback. The stimuli use 3/4 or 4/4 time signatures and tempos of 104–212 beats per minute. They include songs with lyrics, corresponding instrumental versions, and instrumental pieces. The dataset is available under ODC-PDDL at https://hyper.ai/en/datasets/5591.

B.2 Data Preprocessing

The following procedures define the inputs and classification targets used in our evaluation, following the preprocessing described in the source studies.

For **Du-IN**, we select 10 SEEG channels per subject using the configuration from the original study. Signals are downsampled to 1000 Hz and segmented into 2.5-second inputs. Each input is assigned one of the 61 spoken-word labels.

For **Chisco**, the tasks are silent reading (Chisco-R) and imagined speech (Chisco-I). The two electrooculography channels (VEO and HEO) and the trigger channel are excluded, leaving 122 EEG channels. Signals are downsampled to 500 Hz. Inputs span 5 seconds for Chisco-R and 3.3 seconds for Chisco-I. Both tasks use the 39 semantic categories as classification targets.

For **FacesHouses**, continuous ECoG recordings are segmented using stimulus markers, retaining only face and house trials. Channel-wise z-score normalization reduces differences in signal scale. Inputs contain 31–102 channels per subject, sampled at 1000 Hz over 400 ms. The classification target distinguishes faces from houses.

For **ThingsEEG**, we use the preprocessed test partition released by the dataset authors to construct a 200-class supervised decoding task. Each image has 80 repetitions, yielding a tensor of shape $200 \times 80 \times 17 \times 100$ for each subject. The 17 channels are O1, Oz, O2, PO7, PO3, POz, PO4, PO8, P7, P5, P3, P1, Pz, P2, P4, P6, and P8. These channels cover occipital and parietal recording locations. Reshaping the first two axes yields 16,000 labeled trials, each containing 17 channels and 100 time steps. For the main evaluation, 20% of trials form a fixed test set; the remaining 80% are used for three-fold training and validation. Models are evaluated separately for each subject. This setting measures classification among the 200 selected images rather than generalization to unseen image classes.

For the **SEED-DV** dataset, we adopted the first benchmark task from the original SEED-DV study: 40-class classification of fine-grained video concepts, to evaluate dynamic visual stimulus decoding. Following the preprocessing protocol described in the original work, we downsampled the EEG signals to 200 Hz, with each trial spanning 2 seconds and containing 62 channels. The classification labels correspond to the 40 predefined video concepts. To standardize our evaluation pipeline, we merged data across all blocks for each subject, randomly shuffled the trials, and partitioned them into training, validation, and test sets.

For **OpenMIIR**, we evaluate music perception and imagery in nine participants. EEG recordings are band-pass filtered between 0.5 and 30 Hz at the original sampling rate of 512 Hz. Perception uses condition 1, whereas imagery combines conditions 2–4. Data partitioning follows Appendix D.1. Channel-wise z-score statistics are estimated from the training set and applied unchanged to validation and test data.

C Model Architectures and Configurations

C.1 Architectures in the Roadmap Study

This section specifies the architectures used to develop the practical design recipe in Section 2.

Basic architecture study. The model depth and embedding dimension during the basic architecture study are shown in Table 5. In both the CNN-1D and CNN-2D architectures, the convolution kernel size is set to 3. In CNN-1D, the number of channels increases from 1 to 64 in the stem layer, followed by 4 stages, each consisting of 5 layers. The number of channels in these stages increases progressively from 128 to 256, 512, and finally 1024. In CNN-2D, the channel configuration also starts from 1 to 64 in the stem layer. This is followed by 4 stages, each containing 4 layers, with the number of channels increasing from 96 to 192, 384, and 768, respectively. In the CNN-GRU architecture, the ratio of convolutional layers to GRU layers is set to 3:1. In the CNN-Transformer architecture, the initial ratio of convolutional layers to Transformer modules is set to 4:1. To fairly investigate how short- and long-term temporal information impacts model performance, we fix the total number of layers at 20 and systematically adjust the proportion of Transformer modules. To systematically evaluate the effectiveness of the proposed patching method, we vary only the data segmentation approach while keeping the embedding and subsequent self-attention-based feature extraction pipeline unchanged. All other hyperparameters are fixed to ensure that the observed performance differences can be attributed to the effectiveness of the patching method.

Table 5: **Model configurations for the basic architecture study.**

Model	# Params (M)	# Layers	# Dimension
CNN-1D	30	20	1024
CNN-2D	35	16	768
GRU	25	4	512
Transformer	27	8	512
PatchTST	27	8	512
iTransformer	27	8	512
CNN-GRU	34	16	512
CNN-Transformer	32	20	768
CNN-Transformer v2	36	20	1024

Macro optimization. From a macro perspective, we optimize the transformation of the latent space during forward propagation. Specifically, we investigate two strategies for increasing the number of feature maps: the step approach and the leap approach. In the step approach, the number of feature maps is gradually increased: starting from 1 to 64 through the stem layers, then progressively to 96, 192, 384, and 768 across 4 stages, each containing 4 layers. In contrast, the leap approach increases the number of feature maps from 1 to 64 in the stem layer, and then directly increases it to 384 using an embedding layer. Subsequently, 16 layers are used to refine high-level features. We also investigate two strategies for decreasing the size of feature maps: the pyramid approach and the pagoda approach. In the pyramid approach, a downsampling layer is placed in each of the last three stages, with a downsampling ratio of 2 at each stage. In contrast, the pagoda approach places three identical downsampling layers consecutively within the second stage.

Micro optimization. From a micro perspective, we optimize the computation method. Specifically, we investigate two variants of vanilla convolution: the group convolution and the separable convolution. For the group convolution, the number of groups is set to 4. For the separable convolution, the number of groups is set to C_{in}, followed by a pointwise convolution with a kernel size of 1 to aggregate channel-wise information.

C.2 NeuroSketch Implementation

The NeuroSketch recipe is developed through staged architectural refinement and instantiated at two model scales. Starting from the approximately 35M-parameter CNN-2D backbone, macro-level optimization yields a 35.6M-parameter configuration. Grouped convolutions subsequently reduce the backbone parameter count to 14.4M. After establishing the recipe, we further reduce stage widths and depths to obtain its two instantiations, NeuroSketch-Base and NeuroSketch-Large, with

approximately 1.4M and 4.2M parameters, respectively. The smaller models can still perform better because the gain mainly comes from a practical design recipe with a more suitable backbone family and improved macro and micro design choices, rather than from parameter count alone. Once the architecture is better aligned with the temporal-spatial structure of neural decoding, width and depth scaling can further improve parameter efficiency while preserving performance. Table 6 summarizes the configurations of both variants. Their shared architecture is described below.

Given an input $\mathbf{X} \in \mathbb{R}^{B\times C\times L}$, each model folds groups of three consecutive time samples into a channel-like dimension. This produces a two-dimensional representation $\mathbf{X}' \in \mathbb{R}^{B\times 1\times 3C\times \lceil L/3\rceil}$. The transformation is designed to expand the local temporal context available to the first convolutional layer while limiting early mixing across recording channels. Applied directly to the original $C \times L$ representation, a 3×3 kernel spans three adjacent recording channels but only three consecutive time samples. Folding the temporal axis allows the same kernel to cover a wider temporal range, supporting short-range temporal modeling before broader cross-channel integration in subsequent layers.

The reshaped two-dimensional representation is subsequently processed by a stem stage comprising four sequential stem blocks. Each block consists of a two-dimensional convolutional layer, followed by batch normalization and a ReLU activation function. The convolutional layers in all four blocks utilize a kernel size of 3, padding of 1, and strides of 2, 1, 1, and 2, respectively. The input and output channel dimensions for the four blocks are as follows: from 1 to 64, 64 to 32, 32 to 64, and 64 to 96, respectively.

Following the stem stage, there are four feature extraction stages. The input/output channel dimensions for these stages are $[96 \to D_{\text{stage 1}}]$, $[D_{\text{stage 1}} \to D_{\text{stage 2}}]$, $[D_{\text{stage 2}} \to D_{\text{stage 3}}]$, $[D_{\text{stage 3}} \to D_{\text{stage 4}}]$. Each stage contains d blocks with two key components:

1. **Patch Embedding:** In the second stage, every block uses a 3×3 convolution with stride 2 followed by batch normalization, resulting in two downsampling operations in NeuroSketch-Base and three in NeuroSketch-Large. In the other stages, a 3×3 convolution with stride 1 followed by batch normalization is used when the number of feature maps changes. Otherwise, an identity mapping is applied.
2. **Convolution Module:** This component applies grouped 3×3 convolutions, with the number of groups equal to G to efficiently capture local channel-wise dependencies. It is followed by batch normalization, ReLU activation, and a 1×1 convolution for feature fusion. The output is added back to the patch-embedded input via a residual connection.

After the four feature extraction stages, we obtain a tensor of shape $\mathbb{R}^{B\times D_{\text{stage 4}}\times C'\times T'}$, where C' and T' denote the downsampled channel and temporal dimensions. We then apply GeM pooling to aggregate it into a representation $\mathbb{R}^{B\times D_{\text{stage 4}}}$, which is finally passed through a linear layer to produce the class probabilities.

Table 6: **Architectural configurations of NeuroSketch-Base and NeuroSketch-Large.**

	NeuroSketch-Base	NeuroSketch-Large
# Params (M)	1.4	4.2
$D_{\text{stage 1}}$	96	96
$D_{\text{stage 2}}$	128	144
$D_{\text{stage 3}}$	160	256
$D_{\text{stage 4}}$	192	384
d	2	3
G	4	4

C.3 Baseline Architectures and Configurations

This section describes the ten baselines used in the main evaluation and their implemented configurations. For reference, parameter counts are calculated using inputs with 64 recording channels and 2,000 time samples, with 12 output classes. Counts include all parameters registered in each

model implementation, including classification heads. Under this setting, NeuroSketch-Base and NeuroSketch-Large contain approximately 1.4M and 4.2M parameters, respectively. Their stage widths and depths are specified in Table 6.

ConvFormer and **CAFormer** (Yu et al., 2023) are two variants of MetaFormer (Yu et al., 2022), employing different token mixers. ConvFormer uses depthwise separable convolution as its token mixer. With this design, ConvFormer can be regarded as a pure CNN model that does not rely on channel or spatial attention mechanisms. CAFormer uses depthwise separable convolution as the token mixer in the first two stages of the model and self-attention in the last two stages. This design enables CAFormer to capture local features while better obtaining long-range dependencies.

ConvFormer and CAFormer use their S18 backbones, with approximately 26.8M and 26.3M parameters, respectively. Both models use the same input preprocessing as NeuroSketch to construct a 2D representation of the neural recordings. GeM pooling converts the backbone outputs into 512-dimensional representations, which are passed to task-specific linear classifiers.

ModernTCN (Luo & Wang, 2024) is a modernized purely convolutional architecture. It enhances the traditional TCN by incorporating depthwise convolutions and convolutional feed-forward networks (ConvFFNs) into the 1D CNN design. Additionally, it introduces time series–specific adaptations, such as patchified variable-independent embeddings, to improve suitability for time series tasks.

ModernTCN uses three stages with 32, 64, and 128 feature channels, respectively, and one block per stage. The large temporal kernels have sizes 21, 19, and 13, while the small kernels have size 5 throughout. Each feed-forward module expands the feature width by a factor of four. The input patch length and stride are both 50 time samples. This configuration contains approximately 8.1M parameters.

MedFormer (Wang et al., 2024) is a multi-granularity patching Transformer specifically designed for medical time series classification. It effectively captures the features of medical time series through cross-channel patching, multi-granularity embedding, and a two-stage multi-granularity self-attention mechanism.

MedFormer processes temporal patches of 5, 10, and 20 samples using 128-dimensional embeddings. Its encoder contains six layers, each with eight attention heads and a feed-forward hidden dimension of 256. This configuration contains approximately 2.3M parameters.

DeepConvNet (Schirrmeister et al., 2017) is an EEG decoding CNN. Temporal convolution first extracts frequency-related features, followed by spatial convolution across recording channels. Subsequent convolution, batch normalization, ELU, and max-pooling layers learn hierarchical representations, with dropout used during training. A final dense layer performs classification.

DeepConvNet uses four convolutional blocks with 25, 50, 100, and 200 feature channels. Temporal convolutions use kernels of length 5, and each block halves the temporal resolution through pooling. LeakyReLU activation and dropout with a rate of 0.5 are used throughout. This configuration contains approximately 0.5M parameters.

EEGNet (Lawhern et al., 2018) is a compact CNN for EEG decoding. Temporal convolutions extract temporal features, and depthwise spatial convolutions learn spatial filters across recording channels. A subsequent separable convolution combines depthwise temporal filtering with pointwise feature mixing. Average pooling reduces the temporal resolution before classification.

EEGNet uses 16 temporal filters of length 64 and two spatial filters per temporal filter. Depthwise spatial convolution spans all recording channels, producing 32 feature maps with a singleton recording-channel dimension. A depthwise temporal convolution of length 16 followed by a pointwise convolution retains 32 feature maps. The two processing blocks use batch normalization, ELU activation, average pooling with temporal factors of 4 and 8, respectively, and dropout at a rate of 0.25. Flattened features are passed to a linear classifier. Spatial filters and classifier weights use maximum-norm constraints of 1 and 0.25, respectively. This configuration contains 28,588 trainable parameters.

EEG-Conformer (Song et al., 2022) is a CNN-Transformer model for EEG signals. The convolution module learns low-level local features through one-dimensional temporal and spatial convolution layers, and the self-attention module processes the output of the convolution module to learn global temporal dependencies.

EEG-Conformer uses 40-dimensional embeddings and six Transformer encoder layers with five attention heads per layer. Its convolutional input module applies temporal pooling with a stride of 50 samples. The encoded features are averaged over time and passed through layer normalization and a linear classifier. This configuration contains approximately 0.2M parameters.

SPaRCNet (Jing et al., 2023) is a 1D CNN for seizure pattern recognition. Its structure features dense and transition blocks. Each dense block has four layers of two convolutional layers and two ELUs, and each transition block contains an ELU, a convolutional layer, and an average pooling layer.

SPaRCNet starts with 64 feature channels and uses six dense blocks, each containing four layers. Each layer adds 32 feature channels and uses a bottleneck with 128 channels. Dropout rates are 0.2 within the dense layers and 0.5 in the classifier. This configuration contains approximately 0.9M parameters.

seegnificant (Mentzelopoulos et al., 2024) is a brain foundation model for cross-subject neural decoding from SEEG. It tokenizes electrode-wise signals using convolutions, models long-term temporal dependencies with self-attention, and integrates electrode 3D spatial locations through positional encoding followed by cross-electrode attention. A unified backbone extracts global neural representations, while subject-specific task heads enable individualized decoding. Trained on multi-session SEEG data from 21 participants, seegnificant demonstrates effective behavioral response time decoding and supports few-shot transfer to new subjects, offering a path toward robust multi-subject generalization in SEEG analysis.

For seegnificant, input signals are resampled to 400 Hz to match the sampling rate used in pretraining. The configuration uses an embedding dimension of 2, with one temporal and one spatial Transformer block. Its outputs are flattened and passed to a 128-unit hidden layer with ReLU activation and dropout with a rate of 0.5. A final linear layer produces one logit per target class. The complete implementation contains approximately 6.3M parameters under the reference setting.

CBraMod (Wang et al., 2025a) is a brain foundation model for EEG decoding. The model first segments EEG signals into patches and randomly masks them. After patch encoding and asymmetric conditional position encoding, it learns spatio-temporal dependencies through the parallel spatial and temporal attention mechanisms of the cross-Transformer. Finally, it uses the reconstruction head to reconstruct the masked EEG patches, thereby learning the general representation of EEG signals.

For CBraMod, input signals are resampled to 200 Hz to match the sampling rate used in pretraining. Each recording channel is then divided into non-overlapping 200-sample patches, corresponding to one second, and encoded using 200-dimensional embeddings. The encoder has 12 layers, eight attention heads per layer, and a feed-forward hidden dimension of 800. For classification, the encoded patches are averaged over time to obtain 200 features per recording channel. These features are concatenated across channels and passed to a single linear layer with one output logit per target class. The complete implementation contains approximately 5.1M parameters under the reference setting.

D Training and Evaluation Protocols

This section describes the training and evaluation procedures used in Sections 2 and 3.

D.1 Data Splitting and Evaluation

For Section 2, each subject's data is split into training, validation, and test sets in a 3:1:1 ratio. For each configuration, the checkpoint with the highest validation accuracy is selected for testing.

For Section 3, 20% of each subject's data is reserved as a fixed test set, and the remaining 80% is divided into three folds for training and validation. In each iteration, one fold is used for validation and the other two for training. For each task, test accuracy is first averaged across subjects within each fold, the reported mean and standard deviation are then computed across the three fold-level averages.

D.2 Data Augmentation

We apply the following augmentations during training:

Random Shift. We define a maximum shift range proportional to the input sequence length. For each training instance, a shift step is randomly sampled from this range. A positive value indicates a forward shift of the sequence, whereas a negative value corresponds to a backward shift. This approach enhances the model's robustness to uncertainty in stimulus onset times.

Noise. We generate noise from a standard normal distribution matching the shape of the original data. The noise is scaled by a predefined standard deviation and added to the input to produce noisy data. This method improves the model's resilience to signal perturbations.

Channel Masking. During training, a mask is applied to each channel with the specified probability, zeroing out the corresponding channel values. This technique reduces over-reliance on specific channels and promotes better integration of multi-channel information.

Time Masking. Similar to channel masking, we apply masks along the temporal dimension. This encourages the model to extract features robustly across various time segments and enhances generalization.

Mixup. A mixing coefficient λ is sampled from a Beta distribution parameterized by a hyperparameter $\alpha = 0.4$. The original sample and a randomly chosen sample are linearly combined using λ, and their corresponding labels are mixed accordingly. This augmentation method exposes the model to a broader range of data combinations, thereby improving generalization.

D.3 Training Hyperparameters

For the two foundation-model baselines, seegnificant and CBraMod, the encoders are initialized with official pretrained weights and paired with newly initialized classification heads. All encoder and classification-head weights are jointly fine-tuned on each downstream task using the shared training settings in Table 7.

Table 7: **Training hyperparameters.**

Hyperparameters	Settings
Epochs	100 (Chisco), 500 (others)
Batch size	64
Seed	42
Optimizer	AdamW
Learning rate	1e-3
Weight decay	5e-2
Scheduler	Cosine
Warmup ratio	0.1
Early stop ratio	0.2
Random shift probability	0.5
Random shift ratio	0.2
Add noise probability	0.1
Channel masking probability	0.5
Time masking probability	0.5
Mixup probability	0.5

The shared training and augmentation hyperparameters for all ten baselines and both NeuroSketch instantiations are summarized in Table 7. We also tested learning rates of 5×10^{-4} and 10^{-4}, but retained 10^{-3} for its better decoding performance.

Across multiple datasets, we observed that decoding performance typically improves with an increased number of training epochs. Based on this observation, we set the number of training epochs to 500 in most experiments to fully optimize the model performance. However, for the *Chisco* dataset, the accuracy reaches a plateau at around 80 epochs and shows no further improvement. Therefore, we limit training on *Chisco* to 100 epochs.

The checkpoint with the highest validation accuracy is selected for test evaluation. Data splitting and augmentation procedures are described in Appendices D.1 and D.2, respectively.

D.4 Compute Resources

We use an AMD EPYC 7663 56-core processor and eight NVIDIA A100 GPUs, each with 80 GB of memory. Section 2 outlines 21 architectural configurations, each evaluated across 84 subject–task combinations: 12 for Du-IN, 5 each for Chisco-R and Chisco-I, 14 for FacesHouses, 10 for ThingsEEG, 20 for SEED-DV, and 9 each for OpenMIIR-P and OpenMIIR-I. In total, this results in 1,764 experiments. Each experiment is executed on an A100 GPU and requires, on average, one hour of training time. Section 3 comprises 2,742 training runs: eight non-foundation baselines and the two NeuroSketch instantiations cover all 84 subject–task combinations across three folds ($10\times84\times3 = 2{,}520$), seegnificant covers Du-IN and FacesHouses ($(12+14)\times3 = 78$), and CBraMod covers Chisco-R, Chisco-I, SEED-DV, OpenMIIR-P, and OpenMIIR-I ($(5+5+20+9+9)\times3 = 144$). On average, each run requires approximately 50 minutes of training time. The roadmap study and main evaluation therefore account for 4,506 training runs in total. This count excludes supplementary analyses and additional hyperparameter trials.

E Recipe and Model Analyses

This section evaluates the recipe after scaling down, examines component and hyperparameter choices, and analyzes model scaling and learned representations.

E.1 Revisiting the Recipe After Scaling Down

The roadmap establishes the design choices before reducing model widths and depths. We revisit these choices after scaling down to assess whether the recipe retains its effectiveness.

Experimental setup. Taking NeuroSketch-Large as an example, we modify one design choice at a time while retaining three blocks per stage. For feature-map expansion, Step uses stage widths of $[96, 144, 256, 384]$, whereas Leap sets the stem output and all four stage widths to 248. For downsampling, Pagoda applies a stride-2 convolution in each of the three blocks in stage 2. Pyramid instead applies one stride-2 convolution in the first block of each of stages 2, 3, and 4. We also replace grouped convolutions with either vanilla or separable convolutions. We evaluate these variants on Du-IN, OpenMIIR perception and imagery, and Chisco reading and imagination. All variants share the same inputs and data splits, with checkpoints selected by validation accuracy.

Table 8: **Revisiting the recipe after scaling down.** Accuracy is reported as mean ± sample standard deviation across three folds, after averaging across participants within each fold. The highest mean accuracy for each task is shown in bold.

Configuration	Du-IN	OpenMIIR-P	OpenMIIR-I	Chisco-R	Chisco-I
Reference recipe	$\mathbf{0.652} \pm 0.002$	$\mathbf{0.982} \pm 0.009$	$\mathbf{0.977} \pm 0.002$	$\mathbf{0.116} \pm 0.001$	$\mathbf{0.095} \pm 0.005$
Step → Leap	0.550 ± 0.016	0.974 ± 0.011	0.964 ± 0.001	0.111 ± 0.007	0.093 ± 0.004
Pagoda → Pyramid	0.572 ± 0.012	0.970 ± 0.010	0.960 ± 0.005	0.110 ± 0.007	0.091 ± 0.007
Group → Vanilla	0.637 ± 0.007	0.967 ± 0.007	0.958 ± 0.002	0.108 ± 0.002	0.089 ± 0.003
Group → Separable	0.640 ± 0.005	0.964 ± 0.006	0.955 ± 0.005	0.110 ± 0.003	0.092 ± 0.006

Results. The reference recipe achieves the highest mean accuracy across all five tasks, with each replacement reducing performance (Table 8). The largest reductions occur on Du-IN, where replacing Step and Pagoda lowers accuracy by 10.2 and 8.0 percentage points, respectively. These consistent trends support the effectiveness of the recipe after scaling down, linking the roadmap's architectural findings to a compact model instantiation.

E.2 Component-wise Ablation Study

To systematically evaluate the independent contributions of key architectural components in the model instantiations, we conduct ablation experiments on the pooling strategy and residual connections using NeuroSketch-Base.

Pooling strategy. We compare three pooling methods: Generalized mean pooling (GeM) (used in the final model), average pooling, and max pooling. Table 9 summarizes the results.

Table 9: **Ablation study on pooling strategies.** We compare GeM pooling (used in NeuroSketch) with average and max pooling using NeuroSketch-Base. Results are reported as mean and standard deviation across three runs.

Pooling	Chisco-R	Chisco-I	OpenMIIR-P	OpenMIIR-I	ThingsEEG
GeM (Ours)	**0.111 ± 0.001**	**0.103 ± 0.002**	**0.981 ± 0.010**	**0.970 ± 0.002**	**0.196 ± 0.002**
Average	0.103 ± 0.002	0.095 ± 0.003	0.977 ± 0.013	0.969 ± 0.002	0.192 ± 0.001
Max	0.085 ± 0.002	0.077 ± 0.006	0.972 ± 0.007	0.946 ± 0.001	0.170 ± 0.003

Table 10: **Ablation study on residual connections.** We evaluate NeuroSketch-Base with and without residual connections. Results are reported as mean and standard deviation across three runs.

Residual	Chisco-R	Chisco-I	OpenMIIR-P	OpenMIIR-I	ThingsEEG
w/ (Ours)	**0.111 ± 0.001**	**0.103 ± 0.002**	**0.981 ± 0.010**	**0.970 ± 0.002**	**0.196 ± 0.002**
w/o	0.084 ± 0.002	0.073 ± 0.006	0.910 ± 0.009	0.923 ± 0.004	0.110 ± 0.001

GeM pooling yields consistent improvements over average pooling and substantially outperforms max pooling across all datasets. The learnable parameter in GeM allows adaptive aggregation of temporal and spatial features, which is beneficial for capturing discriminative patterns.

Residual connections. Table 10 compares NeuroSketch-Base with and without residual connections. Removing residual connections reduces accuracy across the evaluated tasks, supporting their role in the final model.

E.3 Sensitivity to Kernel Size and Group Count

Since both NeuroSketch instantiations use the CNN-2D backbone, convolutional operations play a central role in their performance. To better understand their impact, we conducted a hyperparameter analysis focusing on two key factors: the kernel size and the number of groups in group convolutions.

Kernel size. The size of the convolutional kernel directly affects the receptive field and therefore controls the extent of local temporal–spatial dependencies that the model can capture. In NeuroSketch-Large, we adopt a kernel size of 3 as the default setting, since smaller kernels are generally more effective in capturing fine-grained dynamics while keeping the parameter overhead low. To further examine the effect of kernel size, we additionally evaluate larger kernels of 5 and 7 on the Du-IN dataset. The results are summarized in Table 11.

Table 11: **Results of different kernel sizes of NeuroSketch on the Du-IN dataset.** Results are reported as mean and standard deviation, computed across three distinct cross-validation folds.

Kernel Size	Accuracy	Precision	F1 Score
3	0.652 ± 0.002	0.666 ±0.003	0.648 ±0.002
5	0.557 ± 0.005	0.577 ± 0.006	0.550 ± 0.005
7	0.427 ± 0.002	0.456 ± 0.002	0.420 ± 0.003

These results indicate that larger kernel sizes lead to lower decoding accuracy, which aligns with our claim in Section 1 that neural decoding signals exhibit transient temporal dynamics. Larger kernels

tend to capture longer-range temporal features, which may dilute short-term patterns that are critical for accurate decoding in this context.

Number of groups. The number of groups in group convolution determines how the feature channels are partitioned and processed, which in turn affects both the computational cost and the representational capacity. Increasing the number of groups reduces the computational cost, since fewer channels are convolved together within each group. In NeuroSketch-Large, we set the default group number to 4. To further examine the effect of this parameter, we additionally evaluate group numbers of 2, 8, and 16 on the Du-IN dataset. The results are summarized in Table 12.

Table 12: **Effect of the number of convolution groups on NeuroSketch performance on Du-IN.** Results are reported as mean and standard deviation, computed across three distinct cross-validation folds.

#Group	Accuracy	Precision	F1 Score
2	0.651 ± 0.005	0.667 ±0.003	0.647 ±0.003
4	0.652 ± 0.002	0.666 ±0.003	0.648 ±0.002
8	0.642 ± 0.002	0.658 ±0.003	0.638 ±0.002
16	0.642 ± 0.001	0.656 ±0.003	0.637 ±0.002

The results indicate that setting the group number to 4 provides the most favorable configuration. Compared with using 2 groups, this setting achieves competitive decoding performance while reducing the computational cost. At the same time, it clearly outperforms larger group numbers of 8 and 16 in terms of decoding accuracy, highlighting that 4 groups strike an effective balance between efficiency and representational capacity.

E.4 Scaling Behavior

In this section, we analyze scaling behavior across the two NeuroSketch instantiations. As shown in Table 4, NeuroSketch-Large achieves a clear performance improvement over NeuroSketch-Base on the Du-IN dataset, whereas the two models perform comparably on the other datasets. To further investigate this phenomenon, we examine subject-level performance on the Du-IN dataset, with the results summarized in Table 13. We observe that NeuroSketch-Large outperforms NeuroSketch-Base across all subjects, although the extent of improvement varies. To examine how scaling gains vary across subjects, we plot relative accuracy improvement against NeuroSketch-Base accuracy for each subject (Figure 4). Subjects with lower Base accuracy tend to show larger relative gains. Notably, for subjects with NeuroSketch-Base accuracy below 5%, NeuroSketch-Large yields relative improvements of over 400%. This indicates that increasing model capacity within the recipe effectively addresses more challenging tasks.

E.5 Representation Analysis of CNN-2D and Transformer-Based Architectures

Given the empirical performance gap between CNN-based and Transformer-based architectures in Sections 2.2 and 3.2, we further analyze their representations. This analysis aims to uncover the underlying reasons for this gap.

Specifically, we examine model representations from a rank-based perspective, following the framework proposed by Yu et al. (2025). The rank of a feature representation reflects its expressive capacity. A higher rank indicates richer and more diverse features, while a lower rank implies excessive compression and loss of independent information. In a well-organized hierarchy, rank is expected to remain high or increase with depth, as deeper layers capture more abstract yet independent features. In our evaluation, the CNN-2D-based and Transformer-based models achieve the best and worst overall performance, respectively. We therefore analyze NeuroSketch-Large (CNN-2D-based) and MedFormer (Transformer-based) to examine how their layer-wise rank ratios evolve with network depth.

For any layer of a given model, let $\mathbf{O} \in \mathbb{R}^{s \times d}$ denote the output representation of a sample. Here, s represents the sequence length and d the feature dimension. We perform singular value decomposition (SVD) on $\mathbf{O}$ as $\mathbf{O} = \mathbf{U}\Sigma\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices. The diagonal matrix Σ

Table 13: **Scaling comparison of NeuroSketch-Base and NeuroSketch-Large on the Du-IN dataset.** We report mean accuracy with standard deviation (computed over three cross-validation folds). RI denotes the relative improvement in accuracy of NeuroSketch-Large compared to NeuroSketch-Base.

Subject	NeuroSketch-Base	NeuroSketch-Large	RI
1	0.626 ±0.018	0.791 ±0.027	26.37%
2	0.761 ±0.025	0.798 ±0.007	4.95%
3	0.081 ±0.025	0.508 ±0.016	529.34%
4	0.422 ±0.052	0.765 ±0.033	81.36%
5	0.774 ±0.012	0.882 ±0.013	13.90%
6	0.209 ±0.021	0.465 ±0.016	122.49%
7	0.409 ±0.019	0.561 ±0.021	37.28%
8	0.525 ±0.012	0.630 ±0.021	20.00%
9	0.540 ±0.030	0.817 ±0.005	51.36%
10	0.034 ±0.008	0.183 ±0.003	433.01%
11	0.721 ±0.013	0.758 ±0.011	5.09%
12	0.569 ±0.050	0.668 ±0.010	17.39%

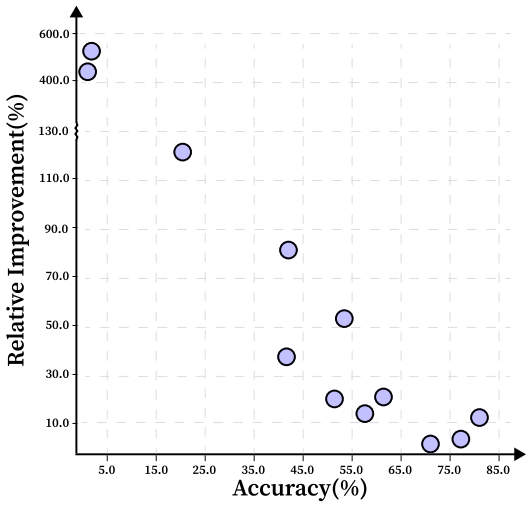


Figure 4: Scatter plot comparing the two NeuroSketch instantiations on the Du-IN dataset. The x axis denotes the accuracy of NeuroSketch-Base for each subject and the y axis represents the relative improvement achieved by NeuroSketch-Large.

contains the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min\{s,d\}}$. Although algebraic rank, defined as the number of nonzero singular values, provides a strict measure of rank, real-world data are often noisy. We therefore adopt a numerical rank measure that is more practical for these data. For a tolerance $\varepsilon > 0$, the ε-rank counts singular values that are significant relative to the largest one:

$$\varepsilon\text{-rank}\,(\mathbf{O}) = \left| \left\{ i | \, \frac{\sigma_i\,(\mathbf{O})}{\sigma_1\,(\mathbf{O})} > \varepsilon \right\} \right|. \tag{1}$$

Since the maximum rank, $\min(s, d)$, varies across layers, we further compute the ε-rank ratio by normalizing the ε-rank to this maximum:

$$\varepsilon\text{-rank ratio} = \frac{\varepsilon\text{-rank}\,(\mathbf{O})}{\min\,(s, d)}. \tag{2}$$

This metric allows a more intuitive comparison of layer-wise rank ratios.

For MedFormer, we extract the ε-rank ratios of the embedding layer and each of its six encoder layers at patch lengths of 5, 10, and 20. This comparison examines how patch length affects the representational rank ratio. For NeuroSketch-Large, we analyze the outputs of all four stem blocks and all blocks across the four feature-extraction stages. We flatten the height and width dimensions into a single sequence dimension for consistent rank analysis.

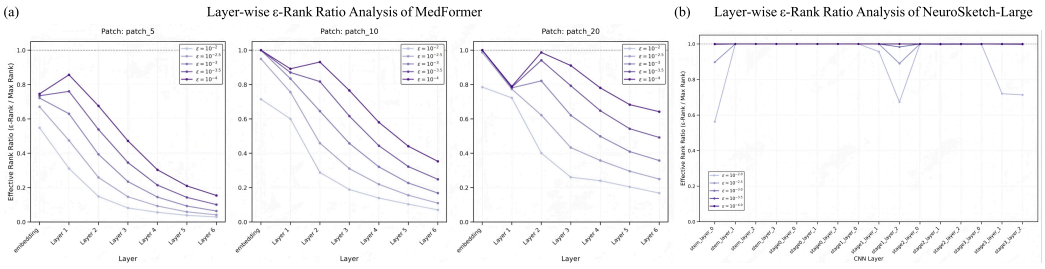


Figure 5: Layer-wise ε-rank ratio analysis of MedFormer and NeuroSketch-Large.

Figure 5 shows two patterns in the evolution of the models' ε-rank ratios.

(1) For multichannel EEG signals, embeddings are not as low-rank as those typically observed in conventional time series (Liang et al., 2025; Yu et al., 2025). When $\varepsilon \leq 10^{-2.5}$, the ε-rank ratios of MedFormer's embedding layer and NeuroSketch-Large's first stem block remain above 0.6. This indicates that both models retain high-dimensional representations rather than collapsing into a low-rank subspace.

Moreover, increasing MedFormer's patch length from 5 to 20 raises the maximum ε-rank ratio, suggesting that longer temporal patches promote richer representations. We attribute these two phenomena to the information compression ratio during the initial embedding. For example, MedFormer's embedding stage applies a 2D convolution with kernel shape $[C, P]$, where C is the number of input channels and P the patch length. The convolution output is projected into an embedding space of dimension D. Consequently, the effective compression ratio is approximately $\frac{C \times P}{D}$. Because brain signals typically involve many channels, often 64 or more, this ratio is much higher than in univariate time-series data. Increasing P further amplifies the compression, producing embeddings with higher ε-rank ratios, consistent with our observations.

(2) NeuroSketch-Large and MedFormer exhibit distinct trends in their layer-wise ε-rank ratios. As network depth increases, MedFormer's rank ratio progressively decreases, indicating excessive compression of input information into a lower-dimensional subspace. NeuroSketch-Large exhibits the opposite trend, with its rank ratio gradually increasing and approaching one in deeper layers.

These contrasting trends imply that the two architectures differ fundamentally in how they organize their representation spaces. In neural decoding tasks, the CNN-2D architecture learns more robust and information-rich representations than the Transformer architecture.

F Extended Evaluation

This section extends the evaluation to additional foundation models, unseen participants, and reduced training data.

F.1 Additional Foundation-Model Evaluation

To further assess how NeuroSketch compares with recent neural foundation models, we include CSBrain (Zhou et al., 2025), CodeBrain (Ma et al., 2026), and LUNA-Base (Döner et al., 2025) in an additional evaluation across four tasks from OpenMIIR and Chisco.

Data preparation. We retain the preprocessing, labels, and participant-specific partitions described in Appendices B.2 and D.1. OpenMIIR uses 64-channel, 600-sample windows at 512 Hz. Chisco retains 122 EEG channels at 500 Hz after excluding three non-EEG channels. OpenMIIR and Chisco have 12 and 39 target classes, respectively. The evaluation covers all nine OpenMIIR participants and all five Chisco participants.

Resampling and amplitude adaptation. We use MNE's FFT-based resampling with anti-aliasing, preserving window duration up to sampling-grid rounding. CSBrain and CodeBrain use 200 Hz inputs, scaled by 10^4 from volts to microvolts divided by 100, without additional z-score normalization. LUNA-Base uses 256 Hz inputs, converted to microvolts and then standardized channel-wise as a project-specific adaptation. For each participant, task, and fold, channel means and standard deviations are fitted across training windows and time samples and reused for validation and testing.

Electrode adaptation. Electrode names and coordinates follow the retained signal-channel order. Chisco coordinates come from the officially released preprocessed FIF files, and OpenMIIR coordinates from the BioSemi montage in the released FIF recordings. Coordinates are converted to meters and represent dataset templates rather than individually digitized electrode locations.

For CSBrain, official electrode-to-region assignments cover 52 of 122 Chisco channels and 58 of 64 OpenMIIR channels. Each remaining channel inherits the region of the nearest officially mapped electrode in the dataset montage, measured after normalizing template coordinates to the unit sphere. The resulting region assignments and channel permutation are supplied to the encoder. This project-specific extension assigns regions without interpolating signals. LUNA-Base receives the three-dimensional coordinates aligned with the input channels. CodeBrain retains the dataset channel order and does not receive coordinates or region assignments.

Initialization and fine-tuning. Encoders are initialized from official pretrained checkpoints. Task-specific output modules are replaced by a common head comprising token mean pooling, layer normalization, and a linear classifier. As in the main experiments, we compared learning rates of 10^{-4}, 5×10^{-4}, and 10^{-3}. The rate of 5×10^{-4} yielded the highest decoding accuracy and was used to jointly fine-tune all encoder and classifier parameters. Models were trained for 500 epochs on OpenMIIR and 100 epochs on Chisco with AdamW, a weight decay of 0.05, an effective batch size of 64, and cosine decay after a 10% warmup. For each fold, we selected the checkpoint with the highest validation accuracy for testing.

Results. As shown in Table 14, the two NeuroSketch instantiations collectively achieve the highest mean accuracy across all four tasks despite being trained from scratch. NeuroSketch-Large outperforms the strongest additional foundation model by 6.3, 5.1, and 1.6 percentage points on OpenMIIR-P, OpenMIIR-I, and Chisco-R, respectively. On Chisco-I, NeuroSketch-Base remains competitive, achieving an accuracy of 0.103 compared with 0.098 for CodeBrain. These results support the effectiveness of the proposed architectural recipe in achieving strong decoding performance without relying on large-scale pretraining.

F.2 Cross-Subject Evaluation

The cross-subject evaluation examines whether NeuroSketch retains its decoding performance when applied to participants unseen during training. NeuroSketch-Base is compared with EEG-Conformer and CBraMod on PhysioNet-MI and SEED-DV, using disjoint subjects for training and testing.

PhysioNet-MI is a four-class motor-imagery task derived from the publicly available EEG Motor Movement/Imagery Dataset (Schalk, 2009). The classes correspond to imagined movements of the

Table 14: Test accuracy (mean ± sample standard deviation of the three fold-level participant averages). Bold indicates the highest mean in each column.

Model	OpenMIIR-P	OpenMIIR-I	Chisco-R	Chisco-I
CSBrain	0.919 ± 0.004	0.926 ± 0.007	0.075 ± 0.003	0.072 ± 0.005
CodeBrain	0.870 ± 0.012	0.865 ± 0.003	0.098 ± 0.005	0.098 ± 0.002
LUNA-Base	0.852 ± 0.015	0.786 ± 0.012	0.100 ± 0.003	0.095 ± 0.005
NeuroSketch-Base	0.981 ± 0.010	0.970 ± 0.002	0.111 ± 0.001	$\mathbf{0.103} \pm 0.002$
NeuroSketch-Large	$\mathbf{0.982} \pm 0.009$	$\mathbf{0.977} \pm 0.002$	$\mathbf{0.116} \pm 0.001$	0.095 ± 0.005

left fist, right fist, both fists, and both feet. We use imagery runs 04, 06, 08, 10, 12, and 14. The 64-channel EEG recordings, originally sampled at 160 Hz, are resampled to 200 Hz and segmented into 4-second inputs containing 800 time samples per channel. Participants are assigned to disjoint training, validation, and test sets by sorting the subject directory names and selecting the first 70 participants for training, the next 19 for validation, and the next 20 for testing. This subject partition remains fixed across five training runs with random seeds 42, 43, 44, 45, and 46. We report the mean and standard deviation of the five test accuracies, reflecting variation across training seeds under the fixed subject partition.

For SEED-DV, we evaluate 40-class visual concept classification using all 20 participants. Each input contains 62 EEG channels sampled at 200 Hz over 2 seconds, giving 400 time samples per channel. Four participants (20%) are randomly held out for testing using a split seed of 42. The remaining 16 participants are partitioned into three groups for training and validation: each group serves as validation once, while the other two are used for training. All samples from a participant remain in the same partition, and the four test participants are fixed across the three folds. For each fold, the checkpoint with the highest validation accuracy is evaluated on the pooled samples of the four test participants. We report the mean and standard deviation of these three test accuracies.

Table 15: **Cross-subject decoding accuracy** (mean ± standard deviation). Higher values indicate better performance.

Model	PhysioNet-MI	SEED-DV
EEG-Conformer	0.537 ± 0.004	0.032 ± 0.005
CBraMod	0.613 ± 0.010	0.025 ± 0.000
NeuroSketch-Base	$\mathbf{0.621} \pm 0.008$	$\mathbf{0.044} \pm 0.011$

Table 15 shows that NeuroSketch-Base achieves the highest mean accuracy among the three models on both tasks. On PhysioNet-MI, its accuracy reaches 62.1%, compared with 61.3% for CBraMod and 53.7% for EEG-Conformer. On SEED-DV, the corresponding accuracies are 4.4%, 2.5%, and 3.2%, respectively. The gains over the strongest baseline are therefore 0.8 and 1.2 percentage points on PhysioNet-MI and SEED-DV. These results extend the evaluation of NeuroSketch to subject shifts in both motor-imagery and visual decoding. Nevertheless, the low absolute accuracies on SEED-DV indicate that cross-subject visual decoding remains challenging in this setting.

F.3 Robustness to Reduced Training Data

To examine sensitivity to the amount of labeled training data, we evaluate NeuroSketch-Base and SPaRCNet on OpenMIIR perception and imagery. Both tasks classify the twelve musical stimuli described in Appendix B.1, using the inputs defined in Appendix B.2. For each task, we retain 50%, 75%, or 100% of the training split while keeping the validation and test sets unchanged. For each participant, the reduced training sets are randomly sampled using a fixed seed of 42. The 100% setting provides the full-training-data reference for interpreting the reduced-data results. These percentages apply only to the training split, so a smaller fraction does not reduce the evaluation set. Nine participants are evaluated using the three-fold protocol in Appendix D.1. Accuracy is averaged across participants within each fold, and the reported mean and standard deviation are calculated across the three fold-level averages.

Table 16: **OpenMIIR accuracy with reduced training data** (%, mean ± standard deviation). The percentage in the first column applies only to the training split.

	Perception		Imagery	
Training data	SPaRCNet	NeuroSketch-Base	SPaRCNet	NeuroSketch-Base
50%	86.4 ± 5.2	$\mathbf{86.8} \pm 5.2$	80.7 ± 0.7	$\mathbf{83.2} \pm 0.8$
75%	94.8 ± 2.9	$\mathbf{96.0} \pm 2.4$	91.3 ± 1.0	$\mathbf{93.2} \pm 0.9$
100%	96.9 ± 0.7	$\mathbf{98.1} \pm 1.0$	94.8 ± 0.5	$\mathbf{97.0} \pm 0.2$

Table 16 shows that accuracy increases with the training fraction for both models on both tasks. NeuroSketch-Base achieves higher mean accuracy than SPaRCNet at every evaluated fraction. With 50% of the training data, its accuracies reach 86.8% for perception and 83.2% for imagery. These exceed SPaRCNet by 0.4 and 2.5 percentage points, respectively. At 75%, the corresponding gains are 1.2 and 1.9 percentage points. The larger gains on imagery are also present with the full training split, where NeuroSketch-Base leads by 2.2 percentage points. Thus, its advantage over SPaRCNet persists under reduced training data, although both models benefit substantially from additional labeled examples.