CodeBrain: Towards Decoupled Interpretability and Multi-Scale Architecture for EEG Foundation Model

Anonymous authors
Paper under double-blind review

Abstract

Electroencephalography (EEG) provides real-time insights into brain activity and supports diverse applications in neuroscience. While EEG foundation models (EFMs) have emerged to address the scalability issues of task-specific models, current approaches still yield clinically uninterpretable and weakly discriminative representations, inefficiently capture global dependencies, and neglect important local neural events. We present *CodeBrain*, a two-stage EFM designed to fill this gap. In the first stage, we introduce the **TFDual-Tokenizer**, which decouples heterogeneous temporal and frequency EEG signals into discrete tokens, quadratically expanding the representation space to enhance discriminative power and offering domain-specific representation-level interpretability by suggesting potential links to neural events and spectral rhythms. In the second stage, we propose the multi-scale **EEGSSM** architecture, which combines structured global convolution with sliding window attention to efficiently capture both sparse long-range and local dependencies, reflecting the brain's small-world topology. Pretrained on the largest public EEG corpus, CodeBrain achieves strong generalization across 8 downstream tasks and 10 datasets under distribution shifts, supported by comprehensive ablations, scaling-law analyses, and interpretability evaluations.

1 Introduction

Electroencephalography (EEG) captures brain activity via scalp electrodes (Niedermeyer & da Silva, 2005) and provides high temporal-resolution signals for neuroscience and cognitive research (da Silva, 2013). To enable automated analysis, researchers have developed various task-specific models for applications such as sleep staging (Lee et al., 2025; Ma et al., 2025), emotion recognition (Jia et al., 2020a; Liu et al., 2024a), motor imagery (Li et al., 2020; Jia et al., 2020b), and other applications (Guerra et al., 2024; Hu et al., 2024). However, building separate models from scratch for each task is resource-intensive and limits scalability, as shared knowledge across tasks cannot be effectively leveraged. Moreover, variations in channel configurations and input lengths across EEG tasks further hinder knowledge transfer. To tackle these issues, EEG foundation models (EFMs) are developed to learn universal representations for diverse downstream tasks (Zhou et al., 2025).

Inspired by masked self-supervised pretraining in natural language processing (Van Den Oord et al., 2017; Devlin et al., 2019), current EFMs commonly adopt patch-wise representation learning: EEG signals are divided into patches, encoded into latent representations, and trained to reconstruct the masked portions. While this approach offers flexibility across varying channel configurations and input lengths by adjusting the patch number and arrangement, direct raw-signal reconstruction (Wang et al., 2024a; 2025) remains challenging due to the inherent noise and variability of EEG. To mitigate this, recent studies have introduced codebook-based tokenization (Jiang et al., 2024; Pradeepkumar et al., 2025), which abstracts away low-level fluctuations and provides a more robust latent space. Despite these advances, existing EFMs still face critical limitations, calling for new architectures.

Failing to Decouple Heterogeneous EEG for Domain-Specific Interpretability. Recent EFMs adopt vector quantization for noise-robust representation (Jiang et al., 2024; Pradeepkumar et al., 2025), following the VQ-VAE framework originally designed for images, where homogeneous visual features make a single tokenizer sufficient (Van Den Oord et al., 2017; Mentzer et al., 2024). In

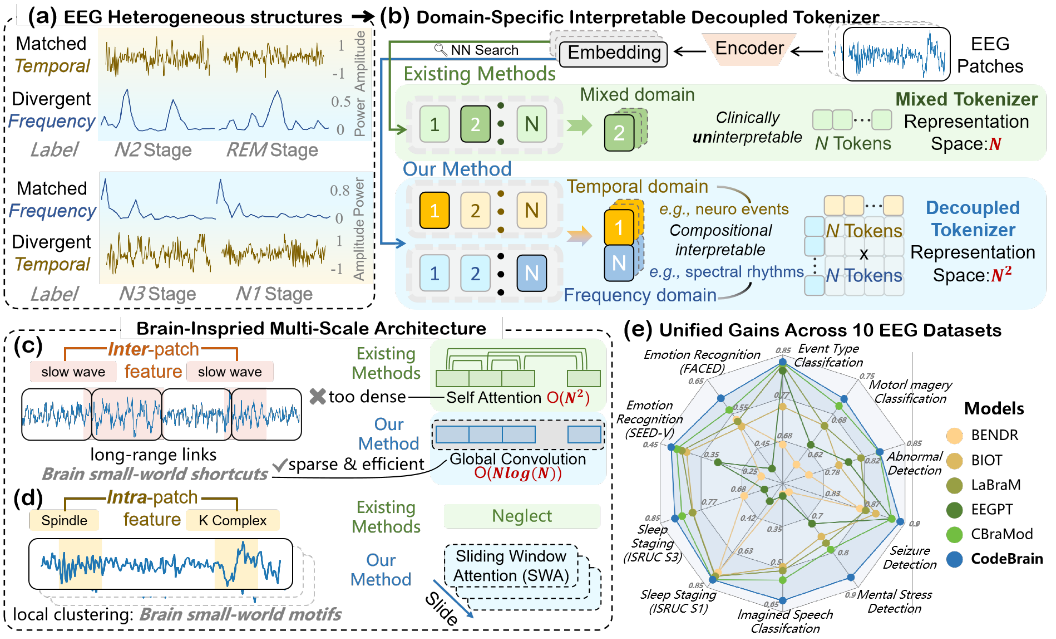


Figure 1: Rationale and overview of *CodeBrain* beyond existing EFMs. **(a)** EEG signals are heterogeneous, as patches matched in one domain may diverge in the other. **(b)** We then propose a decoupled tokenizer for domain-specific interpretable representations while expanding the representation space. **(c-d)** Inspired by the brain's small-world topology, a multi-scale architecture further captures inter-patch dependencies efficiently, while modeling overlooked intra-patch neural events. **(e)** These designs deliver performance gains across 10 EEG datasets.

contrast, EEG exhibits heterogeneous structures: temporal and frequency components reflect distinct aspects of brain activity (Miwakeichi et al., 2004). As illustrated in Fig. 1(a), signals matched in one domain may diverge in the other. Therefore, a mixed tokenizer can conflate domain-specific patterns (Liu et al., 2024b), weakening representation capacity and producing tokens difficult to align with clinically interpretable neural events or spectral rhythms.

Struggling with Efficiently Modeling Global Brain Dependencies. EEG signals exhibit sparse global dependencies and strong local correlations, reflecting the brain's small-world topology (Bullmore & Sporns, 2009; Bassett & Bullmore, 2006; He et al., 2009). Efficient modeling of such structure requires capturing relationships in a scalable way. However, most EFMs (Yang et al., 2023; Jiang et al., 2024; Wang et al., 2024a; 2025) adopt Transformer architectures with fully connected self-attention (Vaswani et al., 2017). This over-connected design is misaligned with the brain's sparse structure and struggles to efficiently capture global dependencies due to its quadratic complexity with sequence length (Tay et al., 2021; Hong et al., 2025; Tegon et al., 2025).

Neglecting Local Dependencies within EEG Patches. EEG signals exhibit rich local waveform structures over short temporal windows, reflecting crucial transient neural events (e.g., sleep waveforms in Fig. 1(d)) (Tatum IV, 2021). However, most existing EFMs represent each EEG patch as a single token and apply attention mechanisms only at the patch level (Wang et al., 2025; Jiang et al., 2024), thereby ignoring important local dependencies within patches.

To address the above challenges, we propose *CodeBrain*, a novel EEG foundation model that integrates a decoupled tokenizer for domain-specific representation-level interpretability with a brain-inspired multi-scale architecture. CodeBrain is trained in two stages. In the first stage, we introduce the **TFDual-Tokenizer** (Fig. 1(b)), which decouples temporal and frequency EEG components into discrete tokens. In the second stage, we develop **EEGSSM**, a masked self-supervised framework inspired by the brain's small-world topology. EEGSSM adopts a structured global convolution backbone, conceptually related to recent state-space sequence models (Smith et al., 2023; Li et al., 2022; Gu et al., 2022a), for sparse and efficient global modeling with a sliding window attention

(SWA) mechanism for capturing local neural events overlooked by prior studies(Fig. 1(c–d)). Our detailed contributions are summarized as follows:

- **Decoupled Tokenizer for Domain-Specific Representation-Level Interpretability.** We propose the *TFDual-Tokenizer*, which decouples temporal and frequency EEG components into discrete tokens. This design quadratically expands the representation space, and qualitative analyzes suggest that some tokens correspond to neural events and spectral rhythms. A contrastive objective is further applied to the temporal branch to stabilize training. To the best of our knowledge, this is the first tokenizer in EFMs to provide domain-specific representation-level interpretability.
- **Brain Small-World Topology Inspired Multi-Scale Architecture.** We design *EEGSSM*, a patch-wise self-supervised framework for EEG. Guided by the brain's small-world topology, it employs structured global convolution to capture sparse long-range temporal dependencies and sliding window attention to model local neural events. In addition, dynamic positional embeddings are used to flexibly learn spatial channel correlations.
- **Strong Generalization and Comprehensive Validation.** Pretrained on the largest publicly available EEG corpus, TUEG (Obeid & Picone, 2016), CodeBrain achieves strong performance on 8 downstream tasks across 10 datasets (Fig. 1(e)) with distribution shifts in cohorts and channel configurations. This suggests the model design plays a central role in generalization. Comprehensive ablations, scaling-law analyses, together with visualization and quantitative analyses, further confirm its robustness, scalability, and provide domain-specific representation-level interpretability.

2 Methodology

2.1 Model Architecture

We introduce *CodeBrain*, a two-stage pretraining framework designed to learn interpretable and universal EEG representations (Fig. 2). The model is motivated by complementary goals: 1) **domain-specific interpretability** via decoupled tokenization of heterogeneous temporal and frequency information, achieved by the proposed **TFDual-Tokenizer**, and 2) **multi-scale modeling** of EEG sequences inspired by the brain's small-world topology, addressed by the **EEGSSM** framework. This design lets Stage 1 learn a tokenizer of patch-level codes, while Stage 2 leverages it for EEG representations. We next provide a formal definition of the two stages to clarify their respective roles.

Stage 1: Decoupled Tokenization. Given a normalized EEG patch $\mathbf{x} \in \mathbb{R}^L$, where L is the patch length, our goal is to discretize $\mathbf{x}$ into temporal and frequency tokens, enabling domain-specific representation learning. Specifically, let $Vt \in \mathbb{R}^{K\times D}$ and $Vf \in \mathbb{R}^{K\times D}$ denote the temporal and frequency codebooks, where K is the vocabulary size and D is the embedding dimension of each token. The tokenizer function is defined as: $vt, vf = f_{\text{tokenizer}}(\mathbf{x}), \quad vt, vf \in \mathbb{R}^D$.

Stage 2: EEG Representation Learning. Given unlabeled EEG sequences $\mathcal{X} = \{X_m\}_{m=1}^N$, where each $X_m \in \mathbb{R}^{C\times f\times T}$ consists of C channels, sampling rate f, and T seconds. We divide each sequence into n non-overlapping patches of t seconds, so each patch length is $L = f \cdot t$. The goal is to train an encoder $f_{\text{enc}} : \mathbb{R}^{C\times n\times L} \to \mathbb{R}^{C\times n\times D}$ that produces latent representations $Z_m = f_{\text{enc}}(X_m)$.

2.2 TFDual-Tokenizer Pretraining

Our TFDual-Tokenizer includes a shared *neural encoder*, a *dual tokenizer* with separate codebooks, and two *decoders*. The *neural encoder* extracts joint time-frequency embeddings from EEG patches, which are then discretized into temporal and frequency tokens by the *dual tokenizer*. Each token stream is reconstructed by a decoder to supervise codebook learning in its respective domain.

Neural Encoder For each patch $\mathbf{x}_i \in \mathbb{R}^L$, we apply the Discrete Fourier Transform (DFT) (Cooley & Tukey, 1965) to obtain its frequency representation:

$$\mathbf{x}_i[k] = DFT(\mathbf{x}_i) \tag{1}$$

where $\mathbf{x}_i[k]$ denotes the k-th frequency component. $\mathbf{x}_i[k]$ and the $\mathbf{x}_i$ are fed into the **TFConv module**, where they are processed in parallel through stacks of convolutional, batch normalization, and ReLU layers. The temporal representation $e_i^t = TFConv(\mathbf{x}_i)$ and frequency representation $e_i^f = TFConv(\mathbf{x}_i[k])$ are concatenated to form a time-frequency embedding $e_i^p = Concat\{e_i^t, e_i^f\}$.

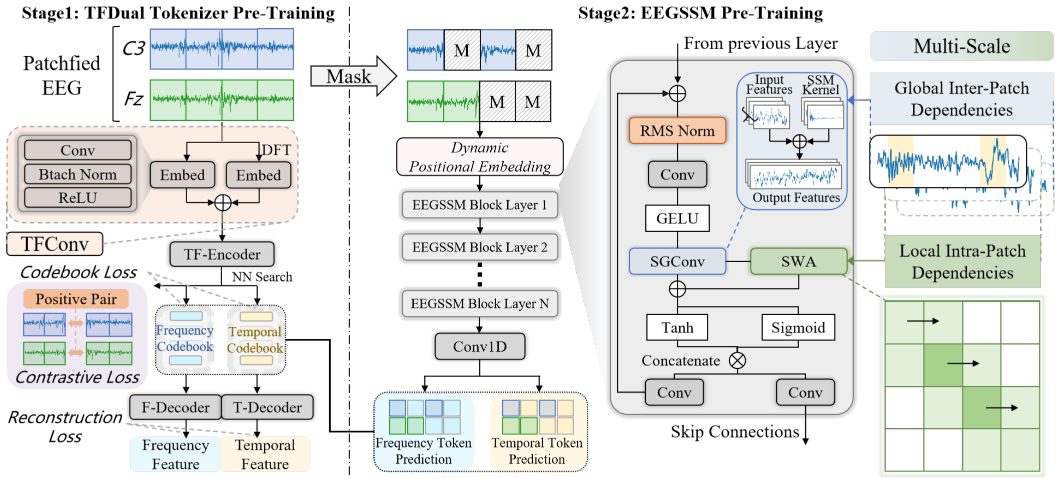


Figure 2: Overview of the *CodeBrain* framework. **Left:** *TFDual-Tokenizer* learns to discretize EEG signals into temporal and frequency tokens using two separate codebooks, by reconstructing both the temporal waveforms and the frequency-domain magnitude and phase. **Right:** *EEGSSM* learns representations by predicting the discrete tokens of masked patches generated by TFDual-Tokenizer.

To get patch representation $\tilde{e}_i$, we then add a positional embedding e_{pos} and feed it into **TF-Encoder**:

$$\tilde{e}_i = Encoder(e_i^p + e_{pos}). \tag{2}$$

We choose a Transformer encoder here since this stage is *patch-to-token*, where its ability to model local contextual relations makes it well-suited for capturing patch-level patterns. To keep the tokenizer channel-agnostic, e_{pos} is shared temporal embeddings without channel-specific identities.

Dual Tokenizer We use two separate tokenizers with distinct codebooks for the temporal and frequency domains, denoted as $vt_j, vf_j \in \mathbb{R}^D$, where $j = 1, \ldots, K$. Given the patch representation $\tilde{e}_i$ from the neural encoder, each tokenizer independently selects the nearest code from its codebook:

$$pt_i = \arg\min_j \|\tilde{e}_i - vt_j\|^2, \quad pf_i = \arg\min_j \|\tilde{e}_i - vf_j\|^2, \tag{3}$$

where pt_i and pf denote the closest positions for the embeddings in the temporal and frequency domain codebook. The effectiveness of the Dual Tokenizer is based on the following proposition:

Proposition 2.1 *Decoupling temporal and frequency codebooks yields representations that are no less effective than those from a joint codebook.*

Proof. See Appendix D.

For this Proposition, we provide empirical validation in Sections 3.4, 3.5 and analysis of Dual Tokenizer 's interpretable structure in Appendix B.

Frequency Codebook Training To train the frequency codebook, we reconstruct amplitude and phase from the code embeddings. For each EEG patch, we apply the DFT to obtain the frequency representation: $\mathbf{x}_i[k] = \text{Re}\{\mathbf{x}_i[k]\} + j \cdot \text{Im}\{\mathbf{x}_i[k]\}$ where $\text{Re}\{\mathbf{x}_i[k]\}$ and $\text{Im}\{\mathbf{x}_i[k]\}$ are the real part and imaginary part respectively, then the amplitude and phase can be calculated as:

$$A_i = \sqrt{\text{Re}(\mathbf{x}_i[k])^2 + \text{Im}(\mathbf{x}_i[k])^2}, \quad \phi_i = \arctan 2\left(\text{Im}(\mathbf{x}_i[k]), \text{Re}(\mathbf{x}_i[k])\right). \tag{4}$$

We use z-score normalization to ensure stable training. The code embedding vf_i, retrieved from the frequency codebook, is passed through the **F-Decoder**, which consists of a Transformer encoder followed by two linear layers:

$$y_i^A = Encoder(MLP(\tilde{e}_i)), \quad y_i^P = Encoder(MLP(\tilde{e}_i)). \tag{5}$$

where y_i^A and y_i^P are the predicted amplitude and phase, respectively. The frequency codebook's training objective is the mean squared error (MSE) loss:

$$\mathcal{L}_i^f = ||y_i^A - A_i||_2^2 + ||y_i^P - \phi_i||_2^2. \tag{6}$$

Temporal Codebook Training Direct reconstruction of temporal features might lead to non-convergence (Jiang et al., 2024). To address this, we combine contrastive loss with reconstruction loss to train a temporal codebook. Inspired by studies on physiological signals (Kiyasseh et al., 2021), we assume temporal dependencies exist between EEG segments, especially within the same channel. For an EEG segment $X_m \in \mathbb{R}^{C\times n\times L}$, we split it into two halves of length $n/2$. We use **TF-Encoder** in Eq. (2) to obtain the representations of these two parts $X_{m1}, X_{m2} \in \mathbb{R}^{C\times \frac{2}{n}\times L}$ separately:

$$e^h_{mi} = Encoder(X_{mi}), i \in \{1,2\}. \tag{7}$$

We encourage the latent representations of different parts within a single segment e^h_{m1} and e^h_{m2} to be similar, while making those of the same part across different segments e^h_{mi} and e^h_{sk} as distinct as possible, where $X_s \in \mathbb{R}^{C\times n\times L}$, $s \neq m, k \neq i$. SimCLR loss (Chen et al., 2020) is used for training:

$$\mathcal{L}^{CL}_m = -\log \frac{\exp\left(\text{sim}(e^h_{m1}, e^h_{m2})/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(\text{sim}(e^h_{mi}, e^h_{sk})/\tau\right)}, \tag{8}$$

where $\text{sim}(e^h_{m1}, e^h_{m2})$ is the cosine similarity, $\tau > 0$ is the temperature parameter, and $\mathbb{1}_{[k\neq i]}$ is an indicator function used to exclude itself. We introduce a **T-Decoder** to reconstruct raw signals from the temporal code embedding $\tilde{e}_i$, which consists of a Transformer encoder followed by a linear projection. Let y_t be the Transformer output from Eq. (2); the overall training objective is:

$$\mathcal{L}^t_i = \mathcal{L}_{CL} + ||y^t_i - \mathbf{x}_i||^2_2, \quad y^t_i = Encoder(MLP(\tilde{e}_i)). \tag{9}$$

Finally, the training objective for the TFDual-Tokenizer can be defined as:

$$\begin{aligned}\mathcal{L}_{\text{tokenizer}} = \sum_{X_m\in\mathcal{X}} \mathcal{L}^{CL}_m + \sum_{X_m\in\mathcal{X}} \sum_{i=0}^{n} \mathcal{L}^f_i + \mathcal{L}^t_i + \underbrace{\|\text{sg}(\tilde{e}_i) - vt_{pt_i}\|^2 + \|\text{sg}(\tilde{e}_i) - vf_{pf_i}\|^2}_{\text{codebook loss}} \\ + \underbrace{\|\tilde{e}_i - \text{sg}(vt_{pt_i})\|^2 + \|\tilde{e}_i - \text{sg}(vf_{pf_i})\|^2}_{\text{commitment loss}},\end{aligned} \tag{10}$$

where $\text{sg}(\cdot)$ denotes stop-gradient to avoid updating encoder parameters.

2.3 EEGSSM Pretraining

In this stage, we introduce a novel convolutional structured state space model framework, called **EEGSSM**, composed of multiple *EEGSSM blocks*. To adapt to unseen channels, we first learn dynamic positional embeddings using a single depthwise 2D convolution with an asymmetric kernel, following the ACPE design (Wang et al., 2025), enabling the model to learn relative inter-channel structures and generalize across heterogeneous EEG channel layouts. The resulting features are processed by EEGSSM blocks. Another 1D convolutional layer then maps the output back to the token space for reconstructing the indices of masked tokens produced by the TFDual-Tokenizer.

EEGSSM Block Our EEGSSM block is composed of several blocks, which are integrated together through a residual connection mechanism. An EEGSSM block consists of a Layer Normalization, SGConv layer, SWA layer, and a gating component. Afterward, we feed the intermediate variables into the SGConv layer to obtain a global receptive field through convolution SSM.

SGConv Layer. SGConv is a structured SSM model (see Appendix C) using convolution architecture, and its convolution structure can be represented as a DFT formula:

$$y = F^{-1}_N D_k F_N u, D_k = \text{diag}(\overline{K} F_N), \tag{11}$$

where F_N denotes the DFT matrix of size N, and the convolution can be computed in $O(N \log N)$ via FFT. As a type of convolutional SSM, SGConv improves the convolution kernel $\overline{K}$ in Eq. (22) by introducing two features: sparse parameterization and kernel decay, making SGConv easier and more efficient to compute compared to the traditional S4 kernel. Let L be the length of the input sequence. The convolution kernel $\overline{K}$ of SGConv is composed of several sub-kernels. Assuming the size of the first sub-kernel is d, with parameters $w_i \in \mathbb{R}^{d \times d}$, then the number of sub-kernels can be expressed as $N = \log_2(\frac{L}{d}) + 1$. The convolution kernel $\overline{K}$ in Eq. (22) can thus be initialized as:

$$\overline{K} = \frac{1}{Z}[k_0, k_1, ..., k_{N-1}], k_i = \alpha^i \text{Upsample}_{2^{max[i-1,0]}d}(w_i), \tag{12}$$

where α denotes the decay coefficient, usually chosen to be 0.5, inducing the decaying structure, and $\text{Upsample}(x)$ denotes upsample x to length l. We also introduce the normalization constant Z to ensure that the convolution operation does not change the scale of the input.

Sliding Window Attention Layer. We included a sliding window attention (SWA) layer to capture fine-grained local temporal dependencies. We apply a small fixed-length window, allowing the model to directly access the content within the context of the window through an attention mechanism by sliding it across the entire sequence, thereby addressing previous models' neglect of intra-patch temporal information. Furthermore, SWA maintains linear computational complexity to ensure training speed remains largely unaffected.

Gate Mechanism. we use a gating mechanism to control the output of the block. We employ a gated unit similar to Wavenet (Obeid & Picone, 2016), which can suppress useless or irrelevant features and help stabilize training in deep networks. We concatenate the global features output y_{sg} by the SGConv layer with the local features output y_{swa} by SWA and feed them into a gated unit:

$$z = tanh(W_f \times Concat(y_{sg}, y_{swa})) \odot \sigma(W_g \times Concat(y_{sg}, y_{swa})), \tag{13}$$

$$y_1 = \text{Conv}(z), y_2 = \text{Conv}(z) \tag{14}$$

where $tanh(\cdot)$ and $\sigma(\cdot)$ are tanh function and sigmoid function, $\odot$ denotes an element-wise multiplication operator, W_f and W_g are learnable convolution filters. y_1 becomes the input to the next block, while y_2 will be aggregated to the output of SSM blocks through a skip connection.

Pre-Training Objective To help the EEGSSM model learn general EEG representations, we use a Masked Autoencoder (MAE) for self-supervised pre-training. For a patched sample $X = \{x_i \mid i \in [1, 2, \ldots, C]\}$, we randomly generate a mask $\mathcal{M} = \{m_i \mid i \in [1, 2, \ldots, C]\}$ from a Bernoulli distribution of r proportion, where $m_i \in \{0, 1\}$. We reconstruct the token indices of masked EEG patches from the TFDual-Tokenizer by cross-entropy loss. Let y_i denotes the output of the EEGSSM block, the probability that the EEG signal matches the corresponding token v_i in the codebooks:

$$p(v_i|x_i) = softmax(Conv1D(y_i)). \tag{15}$$

Suppose the size of the pre-training set is N, the final cross-entropy loss is:

$$\mathcal{L}_p = -\sum_{j=0}^{N} \sum_{n \in \{m_i = 1\}}^{C} p(v_{nj}|x_{nj}). \tag{16}$$

3 Experiments

3.1 Datasets

Pre-Training. We pretrain CodeBrain on the TUH EEG Corpus (Obeid & Picone, 2016), the largest publicly available EEG dataset to date. Data processing follows a standardized pipeline: recordings shorter than 5 minutes are excluded, and the first and last minute of each segment is removed. We retain 19 commonly used EEG channels (C3, C4, Cz, F3, F4, Fp1, Fp2, F7, F8, Fz, O1, O2, P3, P4, Pz, T3, T4, T5, T6), selected based on the international 10–20 system for electrode placement (HH, 1958). We apply band-pass filtering (0.3–75 Hz), and notch filtering at 60 Hz to remove noise. The data is resampled to 200 Hz and divided into 30-second non-overlapping segments. Segments with absolute amplitudes over 100 μV are filtered out. To normalize the signals, each value is divided by 100. Each segment is split into 1-second windows, resulting in 570 EEG patches per sample. After preprocessing, 1,109,545 samples (about 9,246 hours) are retained for pretraining.

Downstream Tasks. We evaluate CodeBrain on 8 downstream tasks across 10 public EEG datasets, which span diverse applications and exhibit distribution shifts from the pretraining dataset, to assess generalizability. Detailed dataset configurations are in Table 3.1. We perform cross-subject or cross-session splits with strict separation between training, validation, and test sets. For **FACED**, we use 80 subjects for training, 20 for validation, and the remaining 23 for testing. In **SEED-V**, each session consists of 15 trials, which are evenly divided into training, validation, and test sets. **ISRUC_S3** consists of 10 subjects, for which we apply an 8:1:1 cross-subject split. **MentalArithmetic** consists of 36 subjects, and we use a 7:1:1 cross-subject split. **BCIC2020-T3** follows the official competition protocol. In **CHB-MIT**, we use recordings from 19 subjects for training, and from 2 subjects each for validation and testing. Additional dataset details are provided in the Appendix G.

Table 1: Summary of Downstream Tasks and Associated EEG Datasets.

Downstream Tasks	**Datasets**	**#Channels**	**Length**	**#Samples**	**Class**
Emotion Recognition	FACED (Chen et al., 2023)	32	10s	10,332	9-class
	SEED-V (Liu et al., 2021)	62	1s	117,744	5-class
Sleep Staging	ISRUC_S1 (Khalighi et al., 2016)	6	30s	86,320	5-class
	ISRUC_S3 (Khalighi et al., 2016)	6	30s	8,500	5-class
Imagined Speech Classification	BCIC 2020-T3 (Jeong et al., 2022)	64	3s	6,000	5-class
Mental Stress Detection	Mental Arithmetic (Mumtaz, 2016)	20	5s	1,707	2-class
Seizure Detection	CHB-MIT (Shoeb, 2009)	16	10s	326,993	2-class
Motor Imagery Classification	SHU-MI (Goldberger et al., 2000)	32	4s	11,988	2-class
Event Type Classification	TUEV (Obeid & Picone, 2016)	16	5s	112,491	6-class
Abnormal Detection	TUAB (Obeid & Picone, 2016)	16	10s	409,455	2-class

3.2 Experiment Settings

Experiment Setup. (1) Pretraining Setup. All experiments are conducted on NVIDIA 40GB A100 GPUs. The TFDual-Tokenizer is trained with temporal and frequency codebooks of 4096 codes (32 dimensions) for 20 epochs, using a batch size of 256 and a learning rate of 1e-4, across six A100 GPUs for approximately ten hours. An 8-layer EEGSSM backbone (15.17M) with a masking ratio of 0.5 is trained for 10 epochs, using a batch size of 256 on two A100 GPUs for about 24 hours.

(2) Finetuning Strategy. We evaluate the quality of the pretrained representations under full finetuning. All downstream task datasets are resampled to 200 Hz to match the pretraining configuration. A three-layer MLP is applied to aggregate channel information, compress the x-second sequence, and map the representation to the target class, with activation and dropout between layers.

Baselines. We compare our model with a comprehensive set of baseline models that include widely used task-specific models, as well as publicly available EFMs that have released pretrained weights. Among the non-foundation baselines, **EEGNet** (Lawhern et al., 2018) and **EEGConformer** (Song et al., 2022) represent compact architectures designed for efficient EEG decoding. **ContraWR** (Yang et al., 2021) is a contrastive-learning–based small model, while **ST-Transformer** (Song et al., 2021) provides a transformer backbone. These models serve as representative lightweight baselines commonly adopted across EEG classification tasks.

For EEG foundation models with publicly available pretrained weights, we include five representative methods that cover diverse pretraining paradigms. **BENDR** (Kostas et al., 2021) adopts a contrastive learning framework. **BIOT** (Yang et al., 2023) uses patch-based continuous tokenization. **LaBraM** introduces discrete neural tokens through vector quantization (Jiang et al., 2024). **EEGPT** (Wang et al., 2024a) and **CBraMod** (Wang et al., 2025) rely on the masked reconstruction of raw EEG signals.

Evaluation Metric. For multi-class classification, we report **Cohen's Kappa**, **Weighted F1 score**, and **Balanced Accuracy**, with Kappa based on validation performance for testing. For binary classification, we use Area Under the ROC Curve (**AUROC**), Area Under the Precision-Recall Curve (**AUC-PR**), and **Balanced Accuracy**, with AUROC based on validation performance for testing. Balanced Accuracy is included to mitigate potential class imbalance effects. All experiments are repeated with five random seeds, and we report the mean and standard deviation.

More details about hyperparameters, baselines, and evaluation metrics are provided in Appendix I, H.

Table 2: Comparison results of different methods on downstream tasks.

Methods	FACED (9-Class)			SEED-V (5-Class)		
	Cohen's Kappa	Weighted F1	Balanced Acc	Cohen's Kappa	Weighted F1	Balanced Acc
EEGNet	0.3342 ± 0.0251	0.4124 ± 0.0141	0.4090 ± 0.0122	0.1006 ± 0.0143	0.2749 ± 0.0098	0.2961 ± 0.0102
EEGConformer	0.3858 ± 0.0186	0.4514 ± 0.0107	0.4559 ± 0.0125	0.1772 ± 0.0174	0.3487 ± 0.0136	0.3537 ± 0.0112
ContraWR	0.4231 ± 0.0151	0.4887 ± 0.0078	0.4887 ± 0.0078	0.1905 ± 0.0188	0.3544 ± 0.0121	0.3546 ± 0.0105
ST-Transformer	0.4137 ± 0.0133	0.4795 ± 0.0096	0.4810 ± 0.0079	0.1083 ± 0.0121	0.2833 ± 0.0105	0.3052 ± 0.0072
BENDR	0.4716 ± 0.0095	0.5340 ± 0.0086	0.5320 ± 0.0083	0.0335 ± 0.0062	0.2026 ± 0.0330	0.2231 ± 0.0059
BIOT	0.4476 ± 0.0254	0.5136 ± 0.0112	0.5118 ± 0.0118	0.2261 ± 0.0262	0.3856 ± 0.0203	0.3837 ± 0.0187
LaBraM	0.4698 ± 0.0102	0.5288 ± 0.0188	0.5273 ± 0.0107	0.2386 ± 0.0209	0.3974 ± 0.0111	0.3976 ± 0.0138
EEGPT	0.4639 ± 0.0023	0.3924 ± 0.0017	0.4607 ± 0.0014	0.1323 ± 0.0062	0.3090 ± 0.0052	0.3061 ± 0.0044
CBraMod	0.5041 ± 0.0122	0.5618 ± 0.0093	0.5509 ± 0.0089	0.2569 ± 0.0143	0.4101 ± 0.0108	0.4091 ± 0.0097
CodeBrain	**0.5406** ± 0.0084	**0.5953** ±0.0113	**0.5941** ± 0.0098	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023
Methods	**ISRUC_S3 (5-Class)**			**BCIC 2020-T3 (5-Class)**		
	Cohen's Kappa	Weighted F1	Balanced Acc	Cohen's Kappa	Weighted F1	Balanced Acc
EEGNet	0.7396 ± 0.0155	0.7407 ± 0.0184	0.7121 ± 0.0134	0.4413 ± 0.0102	0.3016 ± 0.0123	0.4413 ± 0.0096
EEGConformer	0.7482 ± 0.0164	0.7501 ± 0.0211	0.7212 ± 0.0181	0.4488 ± 0.0154	0.3133 ± 0.0183	0.4506 ± 0.0133
ContraWR	0.7493 ± 0.0150	0.7513 ± 0.0185	0.7226 ± 0.0164	0.4407 ± 0.0182	0.3078 ± 0.0218	0.4257 ± 0.0162
ST-Transformer	0.7388 ± 0.0195	0.7399 ± 0.0223	0.7116 ± 0.0197	0.4247 ± 0.0138	0.2941 ± 0.0159	0.4126 ± 0.0122
BENDR	0.5995 ± 0.0151	0.6789 ± 0.0142	0.6352 ± 0.0095	0.0607 ± 0.0093	0.2379 ± 0.0165	0.2485 ± 0.0075
BIOT	0.7168 ± 0.0119	0.7834 ± 0.0096	0.7598 ± 0.0109	0.3650 ± 0.0176	0.4917 ± 0.0079	0.4920 ± 0.0086
LaBraM	0.7194 ± 0.0162	0.7843 ± 0.0189	0.7617 ± 0.0122	0.3800 ± 0.0242	0.5054 ± 0.0205	0.5060 ± 0.0155
EEGPT	0.6160± 0.0856	0.6375 ± 0.0632	0.6650 ± 0.0311	0.0567 ± 0.0164	0.2441 ± 0.0105	0.2453 ± 0.0131
CBraMod	0.7407 ± 0.0251	0.8056 ± 0.0219	0.7844 ± 0.0126	0.4216 ± 0.0163	0.5383 ± 0.0096	0.5373 ± 0.0108
CodeBrain	**0.7671** ± 0.0091	**0.8202** ± 0.0071	**0.7856** ± 0.0031	**0.5127** ± 0.0065	**0.6101** ± 0.0053	**0.6101** ± 0.0052
Methods	**Mental Arithmetic (2-Class)**			**CHB_MIT (2-Class)**		
	AUROC	AUC-PR	Balanced Acc	AUROC	AUC-PR	Balanced Acc
EEGNet	0.7321 ± 0.0108	0.5763 ± 0.0102	0.6770 ± 0.0116	0.8048 ± 0.0136	0.1914 ± 0.0182	0.5658 ± 0.0106
EEGConformer	0.7424 ± 0.0128	0.5829 ± 0.0134	0.6805 ± 0.0123	0.8226 ± 0.0170	0.2209 ± 0.0215	0.5976 ± 0.0141
ContraWR	0.7332 ± 0.0082	0.5787 ± 0.0164	0.6631 ± 0.0097	0.8103 ± 0.0144	0.2279 ± 0.0183	0.6351 ± 0.0122
ST-Transformer	0.7132 ± 0.0174	0.5672 ± 0.0259	0.6631 ± 0.0173	0.8237 ± 0.0491	0.1422 ± 0.0094	0.5915 ± 0.0195
BENDR	0.6248 ± 0.0765	0.3661 ± 0.0672	0.5681 ± 0.0448	0.8632 ± 0.0526	0.3071 ± 0.1240	0.5609 ± 0.0432
BIOT	0.7536 ± 0.0144	0.6004 ± 0.0195	0.6875 ± 0.0186	0.8761 ± 0.0284	0.3277 ± 0.0460	0.7068 ± 0.0457
LaBraM	0.7721 ± 0.0093	0.5999 ± 0.0155	0.6909 ± 0.0125	0.8679 ± 0.0199	0.3287 ± 0.0402	0.7075 ± 0.0358
EEGPT	0.7162 ± 0.0171	0.5081 ± 0.0275	0.5597 ± 0.0171	0.8892 ± 0.0066	0.3073 ± 0.0641	0.5481 ± 0.0151
CBraMod	0.7905 ± 0.0073	0.6267 ± 0.0099	0.7256 ± 0.0132	0.8892 ± 0.0154	0.3689 ± 0.0382	**0.7398** ± 0.0284
CodeBrain	**0.8707** ± 0.0209	**0.7177** ± 0.0421	**0.7514** ± 0.0203	**0.8961** ± 0.0174	**0.4377** ± 0.0288	0.7273 ± 0.0240

3.3 Comparison with Baselines

We ensure consistent data splits across all baselines. Results are reported on six representative downstream datasets, with additional results provided in Appendix J. As shown in Table 2, CodeBrain achieves consistent performance gains compared to baselines. For multi-class classification, it achieves the largest gain of +0.0911 in Cohen's Kappa (21.6%), +0.0718 in Weighted F1 score (13.3%), +0.0728 in Balanced Acc (13.5%) on *BCIC 2020-T3* over the strongest baseline (Wang et al., 2025). For binary classification, it achieves the largest gain of +0.0802 in AUROC (10.1%), +0.0910 in AUC-PR (14.5%) and +0.0258 in Balanced Acc (3.6%) on *Mental Arithmetic* over the strongest baseline. These results demonstrate the superior generalizability of CodeBrain.

3.4 Ablation Study

We conduct ablation studies on three datasets with the same five seeds as in the main experiments to evaluate the key components of CodeBrain (Table 3.3). Below is the detailed analysis:

(1) *Tokenizer configuration*: we compare the proposed *TFDual-Tokenizer* (**Dual**) with variants using a single domain codebook (**Temporal** or **Frequency**) or a shared codebook that jointly reconstructs both domains (**Mixed**). Across all datasets, the Dual codebook consistently yields superior performance.

Table 3: The results of ablation studies for tokenizer configurations and module components.

Dataset	Codebook	CL	SWA	SGConv	Gate	Cohen's Kappa	Weighted F1	Balanced Accuracy
FACED 9-Class	Dual	✓	✓	✓	✓	**0.5406 ± 0.0084**	**0.5953 ± 0.0113**	**0.5941 ± 0.0098**
	Temporal	✓	✓	✓	✓	0.4618 ± 0.0072	0.5277 ± 0.0067	0.5217 ± 0.0056
	Frequency	✓	✓	✓	✓	0.5006 ± 0.0224	0.5607 ± 0.0201	0.5580 ± 0.0187
	Mixed	✓	✓	✓	✓	0.4676 ± 0.0061	0.5319 ± 0.0052	0.5281 ± 0.0049
	Dual	✗	✓	✓	✓	0.5222 ± 0.0082	0.5811 ± 0.0084	0.5765 ± 0.0074
	Dual	✓	✗	✓	✓	0.5192 ± 0.0092	0.5792 ± 0.0093	0.5736 ± 0.0075
	Dual	✓	✓	✗	✓	0.1936 ± 0.1637	0.2627 ± 0.1824	0.2858 ± 0.1467
	Dual	✓	✓	✓	✗	0.2578 ± 0.0340	0.3363 ± 0.0270	0.3431 ± 0.0316
SEED-V 5-Class	Dual	✓	✓	✓	✓	**0.2735 ± 0.0032**	**0.4235 ± 0.0022**	**0.4137 ± 0.0023**
	Temporal	✓	✓	✓	✓	0.2633 ± 0.0116	0.4152 ± 0.0092	0.4068 ± 0.0074
	Frequency	✓	✓	✓	✓	0.2665 ± 0.0208	0.4186 ± 0.0177	0.4098 ± 0.0147
	Mixed	✓	✓	✓	✓	0.2708 ± 0.0047	0.4214 ± 0.0044	0.4124 ± 0.0032
	Dual	✗	✓	✓	✓	0.2589 ± 0.0065	0.4129 ± 0.0056	0.4029 ± 0.0042
	Dual	✓	✗	✓	✓	0.2561 ± 0.0051	0.4106 ± 0.0042	0.4019 ± 0.0034
	Dual	✓	✓	✗	✓	0.2062 ± 0.0099	0.3707 ± 0.0075	0.3620 ± 0.0083
	Dual	✓	✓	✓	✗	0.2212 ± 0.0076	0.3826 ± 0.0057	0.3757 ± 0.0052
ISRUC_S3 5-Class	Dual	✓	✓	✓	✓	**0.7671 ± 0.0091**	**0.8202 ± 0.0071**	**0.7856 ± 0.0031**
	Temporal	✓	✓	✓	✓	0.7314 ± 0.0210	0.7916 ± 0.0181	0.7565 ± 0.0244
	Frequency	✓	✓	✓	✓	0.7390 ± 0.0601	0.7986 ± 0.0514	0.7728 ± 0.0361
	Mixed	✓	✓	✓	✓	0.7400 ± 0.0217	0.7999 ± 0.0171	0.7673 ± 0.0157
	Dual	✗	✓	✓	✓	0.7558 ± 0.0333	0.8130 ± 0.0264	0.7801 ± 0.0132
	Dual	✓	✗	✓	✓	0.7359 ± 0.0324	0.7950 ± 0.0259	0.7621 ± 0.0311
	Dual	✓	✓	✗	✓	0.6218 ± 0.0427	0.6956 ± 0.0279	0.6664 ± 0.0316
	Dual	✓	✓	✓	✗	0.5478 ± 0.0302	0.6429 ± 0.0307	0.6258 ± 0.0448

(2) *Contrastive learning* (CL): We evaluate the impact of contrastive learning in TFDual-Tokenizer pretraining. It leads to consistent gains, indicating a better capture of temporal patterns. Moreover, the CL facilitates the convergence of the temporal codebook, with detailed analyzes provided in Appendix F.

(3) *Components of EEGSSM*: Evaluating the **SWA**, **SGConv**, and **Gate** modules, which demonstrate improvements in EEG representation learning. Including the SWA module consistently improves performance, confirming its regularization effect in capturing local dependencies. The gating mechanism also shows large impact, as it effectively stabilizes fine-tuning and prevents overfitting.

(4) *Scaling Laws*: Prior works (Wang et al., 2025; Jiang et al., 2024) explored scaling with 1-1000 hours of EEG data for pretraining. We extend to 1k-9k hours and 3M-150M models, ranging from a 3.86M 3-layer model with a hidden size of 128 to a 146.75M 12-layer model with hidden size of 384, enabling a systematic exploration of scaling laws across depth and width. As shown in Figure 3, Kappa consistently improves with more data and parameters. These results confirm that larger models yield consistent but diminishing returns. Detailed results and efficiency analysis are provided in Appendix N. These findings indicate that the 8-layer (15.17M) model is a balanced choice between performance and computational efficiency.

We further investigate several key design choices, including mask ratio, SWA window size, codebook size, patch size, SGConv kernel parameters, and subband contributions in Appendix K. We report robustness experiments in Appendix M and computational efficiency in Appendix L. We also provide low-resource comparisons (Appendix O) and pretraining curves (Appendix E), showing stable VQ convergence and efficient EEGSSM training.

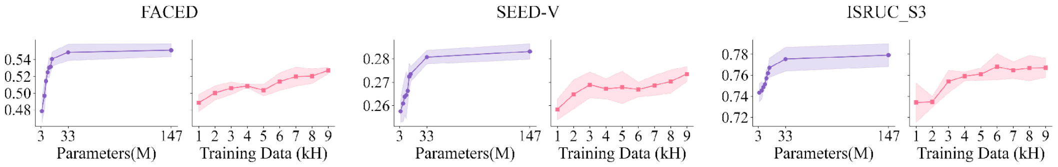


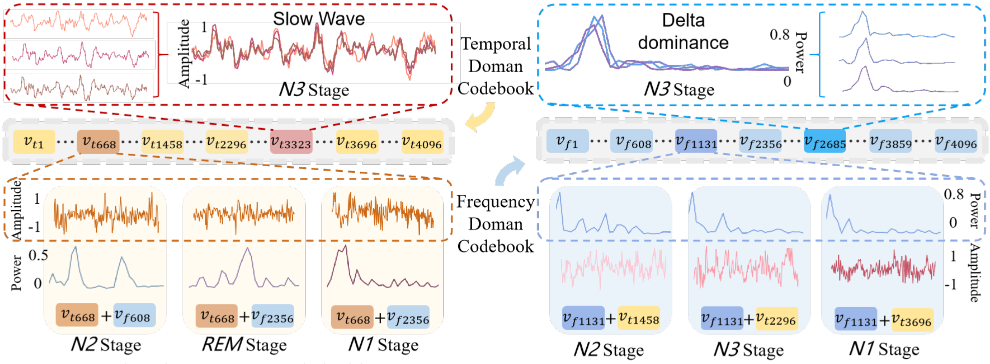
Figure 3: Model and training data scaling laws of CodeBrain across three datasets on Cohen's Kappa.

3.5 Vector Visualization

To demonstrate how the **TFDual-Tokenizer** models heterogeneous EEG, we visualize its learned temporal and frequency codes on the ISRUC_S3 dataset by mapping individual code indices back to their corresponding raw signals. As shown in Figure 4(a)(b), each domain-specific codebook captures meaningful representation-level structures: temporal codes align with neural events (e.g., slow waves), while frequency codes highlight spectral rhythms such as dominant delta activity, both of which are informative for sleep staging. However, in many cases, neither domain alone is sufficient, and richer structure emerges only from their composition, as in Figure 4(c)(d), where the same temporal code can pair with different frequency codes, and vise versa, to yield complementary representations. This decoupled design expands the representation space and enhances representation-level interpretability, with additional meaningful tokens to neurophysiological features and quantitative analyzes provided in Appendix B.

(a) Patterns Captured by Temporal Code No. 3323

(b) Patterns Captured by Frequency Code No. 2685



(c) Temporal Code No. 668 Coupled with Frequency Codes Reflects Different Task Labels

(d) Frequency Code No. 1131 Coupled with Temporal Codes Reflects Different Task Labels

Figure 4: Decoupled time-frequency codes visualization on ISRUC_S3 dataset.

4 Conclusion

In this paper, we present *CodeBrain*, an EEG foundation model that unifies interpretable tokenization with a brain-inspired multi-scale architecture. The TFDual-Tokenizer decouples heterogeneous EEG signals, expanding the representation space while suggesting domain-specific representation-level interpretability, and the EEGSSM architecture integrates structured global convolution with sliding-window attention to efficiently capture both long-range and local dependencies. Pretrained on the large-scale TUEG corpus, CodeBrain demonstrates strong generalization across 8 tasks and 10 datasets under distribution shifts, with comprehensive ablations, scaling-law analyses confirming robustness and scalability. These results establish CodeBrain as a strong foundation for neural time-series representation learning.

Ethics statement

This research adheres to the ICLR Code of Ethics and Responsible Research Guidelines. We confirm that all aspects of the study conform to these guidelines. The work preserves anonymity requirements and does not deviate from the ethical principles set by ICLR. This paper does not involve human subjects, crowdsourcing, or sensitive user data. No Institutional Review Board (IRB) approval was required. The work does not release models or datasets with a high risk of misuse, and it poses no privacy, security, or legal compliance concerns. Furthermore, it does not contain potentially harmful insights, discrimination or bias issues, or conflicts of interest. All experiments are conducted on publicly available EEG datasets, following research integrity and responsible research practices as required by the ICLR Code of Ethics. In addition, disclosures regarding the use of LLM are in the Appendix R.

Reproducibility statement

We have made extensive efforts to ensure the reproducibility of our work. The complete implementation and training scripts are provided in the anonymous code repository (Appendix Q). For theoretical results, all assumptions and complete proofs of the claims are included in Appendix D. For experimental reproducibility, we provide a detailed description of the datasets, preprocessing procedures, and evaluation protocols in Section G, as well as comprehensive hyperparameter settings in Appendix I. These materials together ensure that both the theoretical and empirical results can be independently verified.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Danielle Smith Bassett and ED Bullmore. Small-world brain networks. *The neuroscientist*, 12(6): 512–523, 2006.

Richard B Berry, Rita Brooks, Charlene E Gamaldo, Susan M Harding, Carole Marcus, Bradley V Vaughn, et al. The aasm manual for the scoring of sleep and associated events. *Rules, Terminology and Technical Specifications, Darien, Illinois, American Academy of Sleep Medicine*, 176(2012):7, 2012.

Ed Bullmore and Olaf Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature reviews neuroscience*, 10(3):186–198, 2009.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Chi-Sheng Chen, Ying-Jung Chen, and Aidan Hung-Wen Tsai. Large cognition model: Towards pretrained eeg foundation model. *arXiv preprint arXiv:2502.17464*, 2025a.

Jingjing Chen, Xiaobin Wang, Chen Huang, Xin Hu, Xinke Shen, and Dan Zhang. A large finer-grained affective computing eeg dataset. *Scientific Data*, 10(1):740, 2023.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020.

Zheng Chen, Yasuko Matsubara, Yasushi Sakurai, and Jimeng Sun. Long-term eeg partitioning for seizure onset detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 14221–14229, 2025b.

James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

Fernando Lopes da Silva. Eeg and meg: relevance to neuroscience. *Neuron*, 80(5):1112–1128, 2013.

Tri Dao, Daniel Y Fu, Khaled K Saab, Armin W Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Daniel Y Fu, Elliot L Epstein, Eric Nguyen, Armin W Thomas, Michael Zhang, Tri Dao, Atri Rudra, and Christopher Ré. Simple hardware-efficient long convolutions for sequence modeling. In *International Conference on Machine Learning*, pp. 10373–10391. PMLR, 2023.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33: 1474–1487, 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in Neural Information Processing Systems*, 35:35971–35983, 2022a.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations*, 2022b.

Michele Guerra, Roberto Milanese, Michele Deodato, Madalina G Ciobanu, and Fausto Fasano. Exploring the diagnostic potential of llms in schizophrenia detection through eeg analysis. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 6812–6819. IEEE, 2024.

Yiyu Gui, MingZhi Chen, Yuqi Su, Guibo Luo, and Yuchao Yang. Eegmamba: Bidirectional state space model with mixture of experts for eeg multi-task classification, 2024. URL `https://arxiv.org/abs/2407.20254`.

Yong He, Jinhui Wang, Liang Wang, Zhang J Chen, Chaogan Yan, Hong Yang, Hehan Tang, Chaozhe Zhu, Qiyong Gong, Yufeng Zang, et al. Uncovering intrinsic modular organization of spontaneous brain activity in humans. *PloS one*, 4(4):e5226, 2009.

JASPER HH. The ten-twenty electrode system of the international federation. *Electroenceph clin Neurophysiol*, 10:367–380, 1958.

Jiazhen Hong, Geoffrey Mackellar, and Soheila Ghane. Eegm2: An efficient mamba-2-based self-supervised framework for long-sequence eeg modeling. *arXiv preprint arXiv:2502.17873*, 2025.

Yongquan Hu, Shuning Zhang, Ting Dang, Hong Jia, Flora D Salim, Wen Hu, and Aaron J Quigley. Exploring large-scale language models to evaluate eeg-based multimodal data for mental health. In *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 412–417, 2024.

Ji-Hoon Jeong, Jeong-Hyun Cho, Young-Eun Lee, Seo-Hyun Lee, Gi-Hwan Shin, Young-Seok Kweon, José del R Millán, Klaus-Robert Müller, and Seong-Whan Lee. 2020 international brain–computer interface competition: A review. *Frontiers in human neuroscience*, 16:898300, 2022.

Ziyu Jia, Youfang Lin, Xiyang Cai, Haobin Chen, Haijun Gou, and Jing Wang. Sst-emotionnet: Spatial-spectral-temporal based attention 3d dense network for eeg emotion recognition. In *Proceedings of the 28th ACM international conference on multimedia*, pp. 2909–2917, 2020a.

Ziyu Jia, Youfang Lin, Jing Wang, Kaixin Yang, Tianhang Liu, and Xinwang Zhang. Mmcnn: A multi-branch multi-scale convolutional neural network for motor imagery classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 736–751. Springer, 2020b.

Ziyu Jia, Youfang Lin, Jing Wang, Ronghao Zhou, Xiaojun Ning, Yuanlai He, and Yaoshuai Zhao. Graphsleepnet: Adaptive spatial-temporal graph convolutional networks for sleep stage classification. In *Ijcai*, volume 2021, pp. 1324–1330, 2020c.

Weibang Jiang, Liming Zhao, and Bao-liang Lu. Large brain model for learning generic representations with tremendous eeg data in bci. In *The Twelfth International Conference on Learning Representations*, 2024.

Weibang Jiang, Yansen Wang, Bao-liang Lu, and Dongsheng Li. Neurolm: A universal multi-task foundation model for bridging the gap between language and eeg signals. In *The Thirteenth International Conference on Learning Representations*, 2025.

Sirvan Khalighi, Teresa Sousa, José Moutinho Santos, and Urbano Nunes. Isruc-sleep: A comprehensive public dataset for sleep researchers. *Computer methods and programs in biomedicine*, 124: 180–192, 2016.

Dani Kiyasseh, Tingting Zhu, and David A Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In *International Conference on Machine Learning*, pp. 5606–5615. PMLR, 2021.

Demetres Kostas, Stephane Aroca-Ouellette, and Frank Rudzicz. Bendr: Using transformers and a contrastive self-supervised learning task to learn from massive amounts of eeg data. *Frontiers in Human Neuroscience*, 15:653659, 2021.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, 2018.

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.

Hyojin Lee, You Rim Choi, Hyun Kyung Lee, Jaemin Jeong, Joopyo Hong, Hyun-Woo Shin, and Hyung-Sin Kim. Explainable vision transformer for automatic visual sleep staging on multimodal psg signals. *npj Digital Medicine*, 8(1):55, 2025.

Yuhong Li, Tianle Cai, Yi Zhang, Deming Chen, and Debadeepta Dey. What makes convolutional models great on long sequence modeling? *arXiv preprint arXiv:2210.09298*, 2022.

Zhenqi Li, Jing Wang, Ziyu Jia, and Youfang Lin. Learning space-time-frequency representation with two-stream attention based 3d network for motor imagery classification. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 1124–1129. IEEE, 2020.

Chenyu Liu, Xinliang Zhou, Zhengri Zhu, Liming Zhai, Ziyu Jia, and Yang Liu. Vbh-gnn: Variational bayesian heterogeneous graph neural networks for cross-subject emotion recognition. In *The Twelfth International Conference on Learning Representations*, 2024a.

Qijiong Liu, Xiaoyu Dong, Jiaren Xiao, Nuo Chen, Hengchang Hu, Jieming Zhu, Chenxu Zhu, Tetsuya Sakai, and Xiao-Ming Wu. Vector quantization for recommender systems: a review and outlook. *arXiv preprint arXiv:2405.03110*, 2024b.

Wei Liu, Jie-Lin Qiu, Wei-Long Zheng, and Bao-Liang Lu. Comparing recognition performance and robustness of multimodal deep learning models for multimodal emotion recognition. *IEEE Transactions on Cognitive and Developmental Systems*, 14(2):715–729, 2021.

Jingying Ma, Qika Lin, Ziyu Jia, and Mengling Feng. St-usleepnet: A spatial-temporal coupling prominence network for multi-channel sleep staging. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 4182–4190, 2025.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. In *The Twelfth International Conference on Learning Representations*, 2024.

Fumikazu Miwakeichi, Eduardo Martınez-Montes, Pedro A Valdés-Sosa, Nobuaki Nishiyama, Hiroaki Mizuhara, and Yoko Yamaguchi. Decomposing eeg data into space–time–frequency components using parallel factor analysis. *NeuroImage*, 22(3):1035–1045, 2004.

Navid Mohammadi Foumani, Geoffrey Mackellar, Soheila Ghane, Saad Irtza, Nam Nguyen, and Mahsa Salehi. Eeg2rep: enhancing self-supervised eeg representation through informative masked inputs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5544–5555, 2024.

Wajid Mumtaz. MDD Patients and Healthy Controls EEG Data (New). Figshare, November 2016. URL `https://figshare.com/articles/dataset/EEG_Data_New/4244171`.

Ernst Niedermeyer and FH Lopes da Silva. *Electroencephalography: basic principles, clinical applications, and related fields*. Lippincott Williams & Wilkins, 2005.

Iyad Obeid and Joseph Picone. The temple university hospital eeg data corpus. *Frontiers in neuroscience*, 10:196, 2016.

Jathurshan Pradeepkumar, Xihao Piao, Zheng Chen, and Jimeng Sun. Single-channel eeg tokenization through time-frequency modeling. *arXiv preprint arXiv:2502.16060*, 2025.

Aniruddh Raghu, Payal Chandak, Ridwan Alam, John Guttag, and Collin M. Stultz. Sequential multi-dimensional self-supervised learning for clinical time series, 2023.

Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *54th Annual Meeting of the Association for Computational Linguistics*, pp. 1715–1725. Association for Computational Linguistics (ACL), 2016.

Ali Hossam Shoeb. *Application of machine learning to epileptic seizure onset detection and treatment*. PhD thesis, Massachusetts Institute of Technology, 2009.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. In *ICLR*, 2023.

Yonghao Song, Xueyu Jia, Lie Yang, and Longhan Xie. Transformer-based spatial-temporal feature learning for eeg decoding. *arXiv preprint arXiv:2106.11170*, 2021.

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.

William O Tatum IV. *Handbook of EEG interpretation*. Springer Publishing Company, 2021.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *The Ninth International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=qVyeW-grC2k`.

Anna Tegon, Thorir Mar Ingolfsson, Xiaying Wang, Luca Benini, and Yawei Li. Femba: Efficient and scalable eeg analysis with a bidirectional mamba foundation model. *arXiv preprint arXiv:2502.06438*, 2025.

Xuan-The Tran, Linh Le, Quoc Toan Nguyen, Thomas Do, and Chin-Teng Lin. Eeg-ssm: Leveraging state-space model for dementia detection. *arXiv preprint arXiv:2407.17801*, 2024.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in neural information processing systems*, volume 30, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017.

Guangyu Wang, Wenchao Liu, Yuhong He, Cong Xu, Lin Ma, and Haifeng Li. Eegpt: Pretrained transformer for universal and reliable representation of eeg signals. In *Advances in Neural Information Processing Systems*, volume 37, pp. 39249–39280, 2024a.

Jiquan Wang, Sha Zhao, Haiteng Jiang, Yangxuan Zhou, Zhenghe Yu, Tao Li, Shijian Li, and Gang Pan. Caresleepnet: a hybrid deep learning network for automatic sleep staging. *IEEE Journal of Biomedical and Health Informatics*, 2024b.

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 *The Third International Conference on Learning Representations*, 2025.

Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14408–14419, 2023.

Chaoqi Yang, Danica Xiao, M Brandon Westover, and Jimeng Sun. Self-supervised eeg representation learning for automatic sleep staging. *arXiv preprint arXiv:2110.15278*, 2021.

Chaoqi Yang, M Westover, and Jimeng Sun. Biot: Biosignal transformer for cross-data learning in the wild. In *Advances in Neural Information Processing Systems*, volume 36, pp. 78240–78260, 2023.

Zhizhang Yuan, Daoze Zhang, Junru Chen, Gefei Gu, and Yang Yang. Brant-2: Foundation model for brain signals. *CoRR*, 2024.

Daoze Zhang, Zhizhang Yuan, Yang Yang, Junru Chen, Jingjing Wang, and Yafeng Li. Brant: Foundation model for intracranial neural signal. *Advances in Neural Information Processing Systems*, 36:26304–26321, 2023.

Daoze Zhang, Zhizhang Yuan, Junru Chen, Kerui Chen, and Yang Yang. Brant-x: A unified physiological signal alignment framework. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4155–4166, 2024.

Ruizhe Zheng, Daoze Zhang, Zhizhang Yuan, Junru Chen, Yang Yang, et al. Beatrix: Out-of-distribution generalization of large eeg model via invariant contrastive fine-tuning.

Ruizhe Zheng, Jun Li, Yi Wang, Tian Luo, and Yuguo Yu. Scatterformer: locally-invariant scattering transformer for patient-independent multispectral detection of epileptiform discharges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 148–158, 2023.

Ruizhe Zheng, Lingyan Mao, Dingding Han, Tian Luo, Yi Wang, Jing Ding, and Yuguo Yu. Fapex: Fractional amplitude-phase expressor for robust cross-subject seizure prediction. *arXiv preprint arXiv:2511.03263*, 2025.

Linqi Zhou, Michael Poli, Winnie Xu, Stefano Massaroli, and Stefano Ermon. Deep latent state space models for time-series generation. In *International Conference on Machine Learning*, pp. 42625–42643. PMLR, 2023.

Xinliang Zhou, Chenyu Liu, Zhisheng Chen, Kun Wang, Yi Ding, Ziyu Jia, and Qingsong Wen. Brain foundation models: A survey on advancements in neural signal processing and brain discovery. *arXiv preprint arXiv:2503.00580*, 2025.

Appendix Contents

A Related Work

EEG Foundation Models. Inspired by foundation models in vision and language (Wang et al., 2023; Achiam et al., 2023), EEG research is shifting from task-specific models (Jia et al., 2020c; Wang et al., 2024b; Chen et al., 2025b) (Zheng et al., 2023) to EFMs for learning expressive representations. Current EFMs can be divided into two categories. 1) Contrastive learning-based (CL) (Chen et al., 2025a): BENDER (Kostas et al., 2021) first showed the ability of CL for EEG representations. Then, the Brant series (Zhang et al., 2023; Yuan et al., 2024; Zhang et al., 2024) enables joint representation learning across physiological signals using CL. 2) Reconstruction-based: BIOT (Yang et al., 2023) pioneers cross-modal pretraining for biosignals, including EEG. Subsequent models focus specifically on EEG, learning representations by predicting masked discrete tokens (Jiang et al., 2024; 2025) or reconstructing raw signals (Wang et al., 2024a; Mohammadi Foumani et al., 2024; Wang et al., 2025). Most existing EFMs adopt Transformer architecture, which is suboptimal for EEG due to poor handling of sparse dependencies, quadratic complexity, and ignoring local dependencies by treating each patch as a token.

EEG Tokenization. Tokenization has been key in NLP for generating generalizable and interpretable input representations (Sennrich et al., 2016; Kudo & Richardson, 2018). Inspired by this, early EFMs used patch-based continuous tokenization (Yang et al., 2023; Yuan et al., 2024) to handle EEG noise and variability, but without quantization, leading to unbounded and less interpretable representations. LaBraM (Jiang et al., 2024) introduced vector quantization to learn discrete EEG tokens, following the VQ-VAE design from vision tasks (Van Den Oord et al., 2017). However, this direct transfer overlooks EEG's heterogeneous structure, limiting representation capacity. Moreover, the tokenizer of LaBraM is trained only on frequency-domain reconstruction due to convergence issues with raw signals reconstruction. Later efforts (Pradeepkumar et al., 2025; Jiang et al., 2025) (Zheng et al.) adopted a single codebook under a frequency-dominant pretraining paradigm, limiting the representation space and interpretability.

State Space Model. Recent works have focused on enhancing classic State-Space Models (SSM) to more efficiently model sequential data using deep learning. For example, Rangapuram et al. (2018) used recurrent neural networks to learn parameters in SSM. However, the most significant progress came from Gu et al. (2021) with the introduction of the structured state space model (S4), which reduces the computational complexity of SSM in modeling long sequences using a special state transition matrix (Gu et al., 2020). These low-rank and normal matrices enable SSM to compute global convolution kernels efficiently through fast Fourier transform across the entire sequence. Subsequently, some works have further improved the shortcomings of S4 in areas such as model architecture (Smith et al., 2022) and convolution (Raghu et al., 2023), and have started applying it to tasks such as natural language processing (Dao et al., 2022) and time series analysis (Zhou et al., 2023). Recently, some works have also been applied to the EEG data, Tran et al. (Tran et al., 2024) leveraging SSMs to detect dementia. They extract temporal information from EEG signals through the Mamba architecture and combine it with frequency domain features to better manage the complexity of multivariate EEG. In the work of Gui et al. (2024) Zheng et al. (2025), SSM has also become the backbone network of the EEG foundation model. This further highlights the fast reasoning speed and efficient memory usage of the SSM model when processing EEG signals.

B Interpretability Analyses of TFDual-Tokenizer Representations

We conduct representation-level interpretability analyses of the decoupled TFDual-Tokenizer. As case studies, we visualize selected tokens alongside well-established physiological patterns, illustrating that frequency codes reflect spectral rhythms and temporal codes align with characteristic waveform events. These qualitative examples show that the tokenizer does not discretize signals arbitrarily but organizes them into domain-relevant structures. We further complement these visualizations with a quantitative analysis of class-specific token usage across four datasets, demonstrating that the learned codebooks induce structured representation patterns.

B.1 Exploring Frequency Token Patterns in Relation to Spectral Rhythms

To illustrate the interpretable structure of the learned frequency tokens, we take sleep staging as an example downstream dataset. We focus on N2 and N3 stages, since they are characterized by the most distinctive spectral rhythms in clinical sleep scoring: *spindle* in the sigma band (11-16 Hz) for N2, and *slow wave* in the delta band (0.5-4 Hz) for N3. Using token-activation statistics from the ISRUC_S3 test set, we select the most frequently activated class-specific frequency tokens.

As shown in Figure 5, frequency code No. 2298 tends to capture a sigma bump, with prominent peaks localized in the spindle range. This is similar to N2 spindles (Berry et al., 2012). Quantitatively, this code exhibited a clear sigma peak in approximately 15.4% of its assigned patches. Similarly, Figure 6 shows that frequency code No. 32 predominantly encodes delta dominance, a typical frequency feature of N3 sleep. This code displayed clear delta dominance in about 18.3% of its assigned segments. Taken together, these findings suggest that our frequency-branch tokenizer helps establish a frequency vocabulary for EEG.

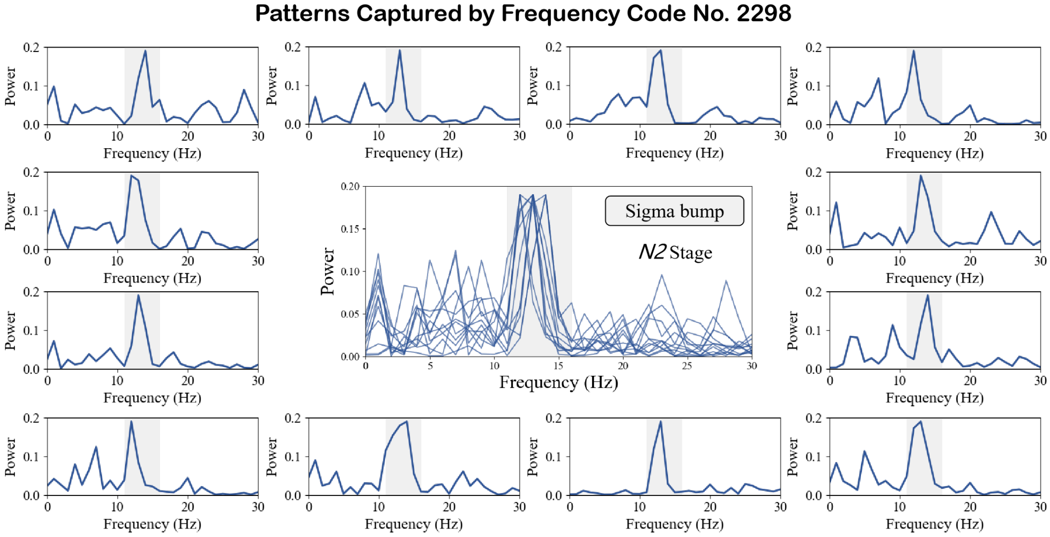


Figure 5: Frequency code capturing sigma bump, a typical spectral rhythm of N2 stage.

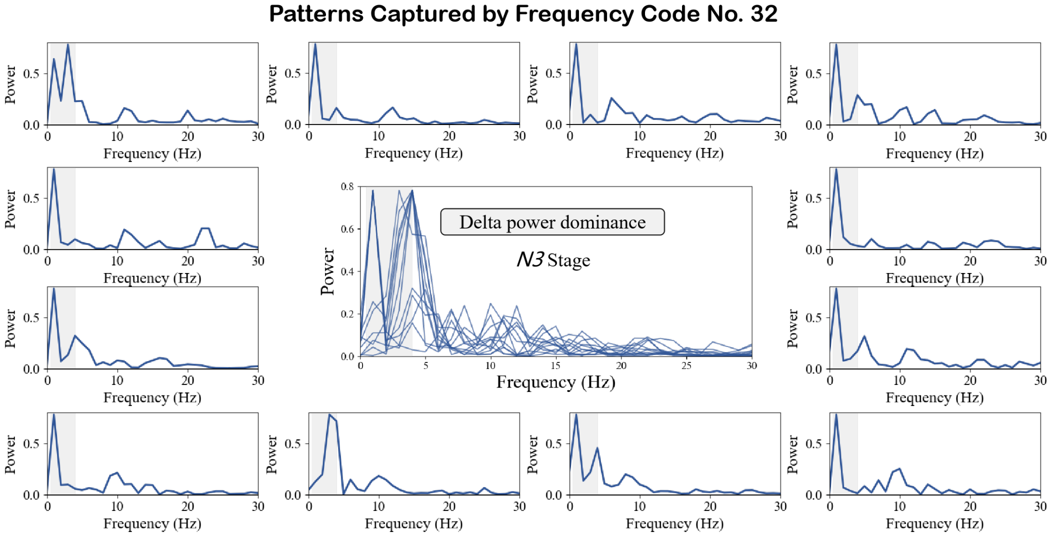


Figure 6: Frequency code capturing delta dominance, a typical spectral rhythm of N3 stage.

B.2 Exploring Temporal Token Patterns in Relation to Neural Events

In addition to spectral rhythms, we applied the same token-activation–based selection procedure to the temporal codes on ISRUC_S3 to examine whether the tokenizer captures coherent and clinically recognizable structures in the raw EEG waveforms. Figure 8 and Figure 7 show that certain codes align with K-complexes and sleep spindles, the hallmark waveforms of N2 sleep. Quantitatively, $\approx 5.34\%$ of the assigned patches contained K-complexes and $\approx 7.01\%$ contained spindles. These rates are consistent with clinical expectations (1–2 K-complexes and 2–5 spindles per minute in N2 Berry et al. (2012)) and, when considering the overall prevalence of N2 in five-class sleep staging, still substantially exceed what would be expected from random token usage.

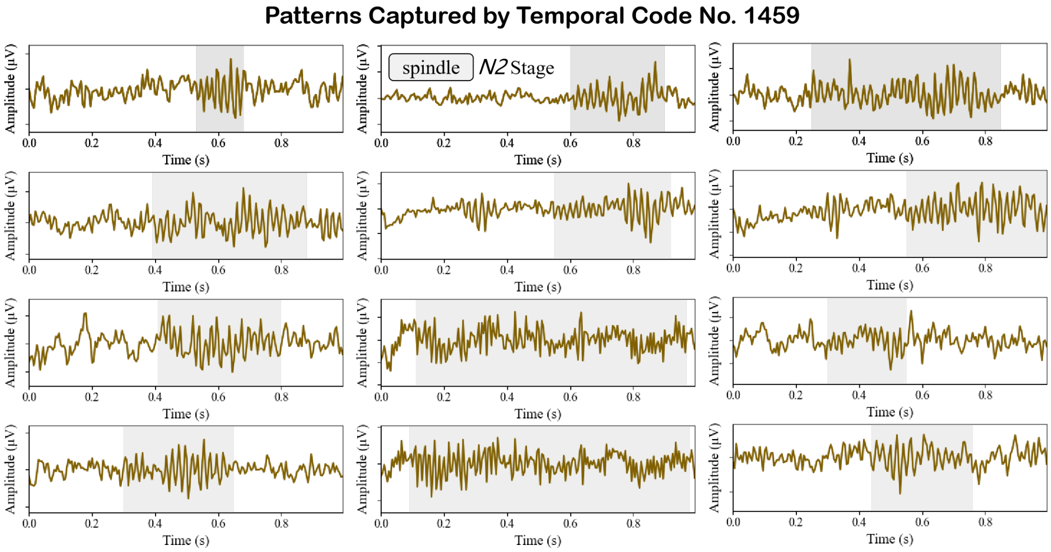


Figure 7: Temporal code capturing sleep spindle, a typical neural events of N2 stage.

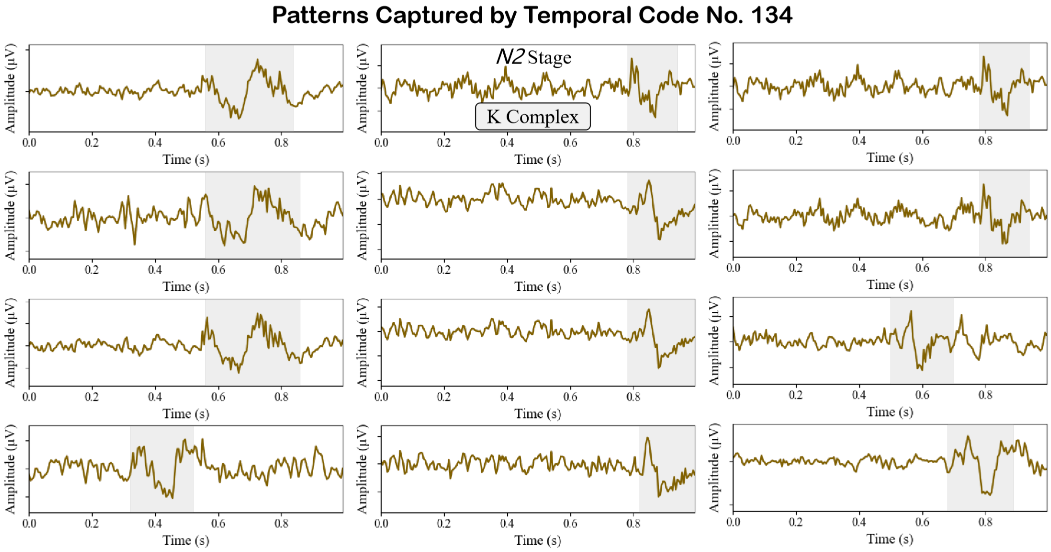


Figure 8: Temporal code capturing K complex, a typical neural events of N2 stage.

Figure 9 shows that temporal code No. 1537 corresponds to slow waves, the defining feature of N3 sleep, occurring in $\approx 40.4\%$ of its assigned patches. This exceeds the 20% per-epoch criterion for N3 scoring Berry et al. (2012) and, given the overall prevalence of N3, indicates that this token provides a meaningful link to neural events.

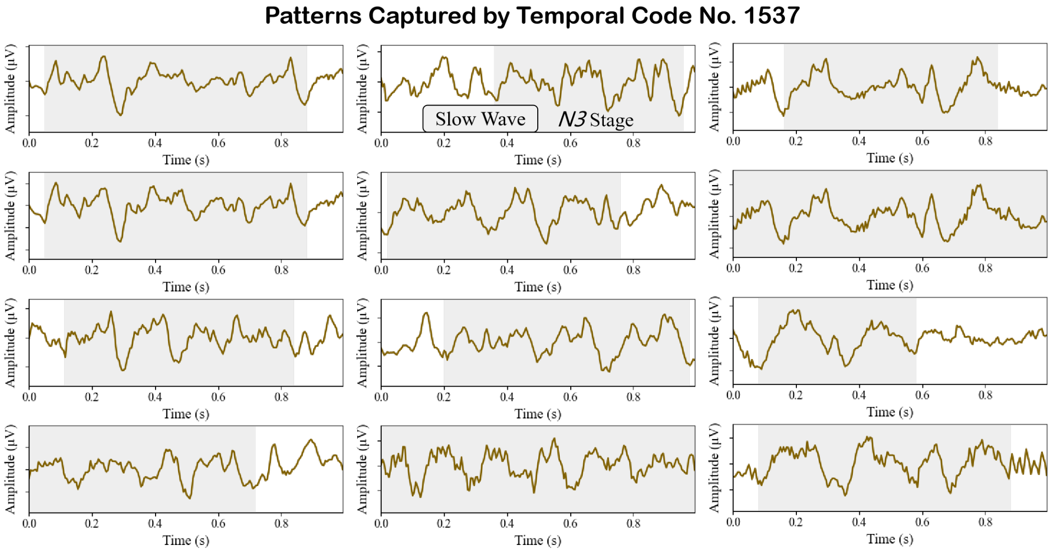


Figure 9: Temporal code capturing slow wave, a typical neural events of N3 stage.

Taken together, these observations suggest that the temporal branch of our tokenizer contributes to establishing a vocabulary of neurophysiological events, complementing the frequency-domain findings. While temporal waveforms are often noisier and harder to model than spectral rhythms (Jiang et al., 2024), our decoupled design allows temporal tokens to emerge with meaningful associations to critical neural events. This indicates that the temporal branch offers useful insights to clinically relevant waveforms and demonstrates the effectiveness of our approach in capturing complementary structure.

B.3 Class-Specific Token Ratio Analysis for TFDual-Tokenizer

To further characterize the representation structure learned by the TFDual-Tokenizer, we analyze the distribution of token usage across classes on unseen downstream datasets. This analysis aims to examine whether the decoupled tokenizer induces more structured and class-consistent token activation patterns, which would support our motivation for separating temporal and frequency domains at the representation level.

A token (code) is considered class-specific if it predominantly appears in samples of a single class. Formally, for a given token c, let $N_c^{(y)}$ denote the number of times c appears in class y. The dominance ratio is defined as:

$$\text{Dominance}(c) = \frac{\max_y N_c^{(y)}}{\sum_y N_c^{(y)}} \tag{17}$$

If Dominance$(c) \geq \tau$ (we use $\tau = 1$), the token is deemed class-specific. The class-specific ratio for a codebook is then computed as:

$$\text{Class-Specific Token Ratio} = \frac{\#\text{ class-specific tokens}}{\text{Total tokens in codebook}} \tag{18}$$

Figure 10 illustrates the proportion of class-specific tokens derived from three configurations: using only the temporal codebook, only the frequency codebook, and a combination of both (TF-Decoupled). We employ two independent codebooks to capture complementary information via the proposed TFDual-Tokenizer module.

Across all four datasets, including two for emotion recognition (FACED and SEED-V) and two for sleep staging (ISRUC_S3 and ISRUC_S1), the decoupled codebook consistently achieves the highest class-specific token ratios, reaching 54.7% on ISRUC_S3 and 46.4% on FACED. These results confirm that decoupling temporal and frequency domain information significantly enhances the model's ability to capture structured representation.

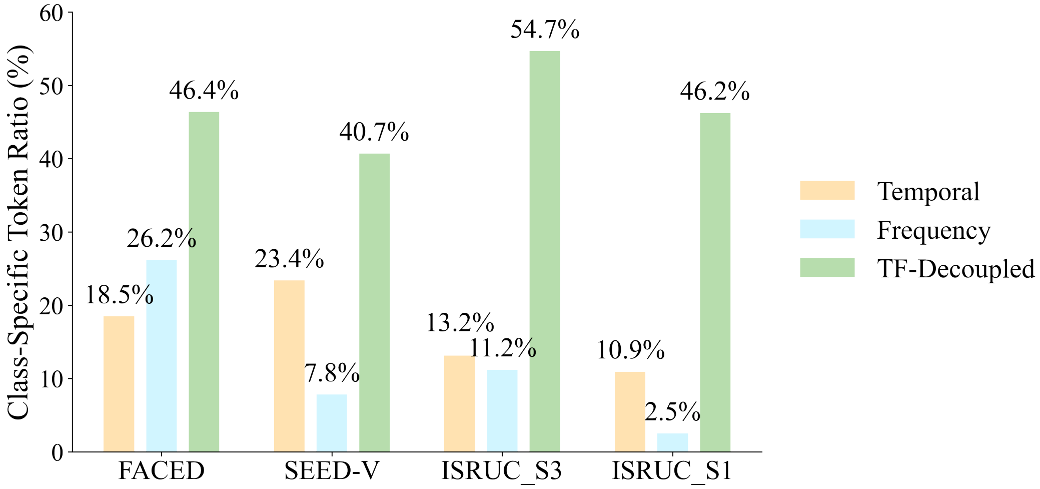


Figure 10: Class-Specific Code Ratio Across Different Codebooks.

B.4 Additional Ablations on Codebook Contribution

To further examine the role of the tokenizer in pretraining, we provide additional exploratory evidence on how the the codebooks influence representation learning. These analyses extend the comparisons in Table 3.3, where the decoupled design outperforms temporal-only, frequency-only, and mixed codebooks. Specifically, we compare the following two ablations on the ISRUC_S3 dataset:

- **Raw-signal reconstruction.** We remove the tokenizer entirely and train the EEGSSM backbone directly to reconstruct raw waveforms, thereby eliminating the discretization step.
- **Masked codebook.** We identify the top 50% most frequently activated tokens across both codebooks on ISRUC_S3. During pretraining, whenever a segment is assigned one of these tokens, we replace it with a placeholder, effectively masking half of the vocabulary and preventing the model from relying on these high-activation codes.

Table 4 reports downstream performance on ISRUC_S3 under these settings.

Table 4: Ablations of the codebook contribution on ISRUC_S3.

Setting	Cohen's κ	Weighted F1	Balanced Acc
TFDual_tokenizer (ours)	**0.7671** ± 0.0091	**0.8202** ± 0.0071	**0.7856** ± 0.0031
Raw-signal Reconstruction	0.7503 ± 0.0087	0.8014 ± 0.0079	0.7763 ± 0.0048
Masked Codebook	0.7426 ± 0.0102	0.7931 ± 0.0084	0.7690 ± 0.0063

These experiments do not suggest that specific interpretable codes directly determine task outcomes. However, they provide supporting evidence that the decoupled tokenizer imparts useful structure during pretraining, and that entirely removing or partially disabling the codebooks leads to consistently degraded downstream performance. These observations reinforce the contribution of the tokenizer at the representation level.

C Structure State Space Model

The state-space model is a classic model in control theory, and it represents the operational state of a system using first-order differential equations (ODE). A continuous state-space model can be defined in the following form:

$$x'(t) = Ax(t) + Bu(t), y(t) = Cx'(t) + Du(t), \tag{19}$$

where $u(t)$ is a vector that represents the input of the system, while $y(t)$ is a vector that represents the output of the system. $x(t)$ and its derivative $x'(t)$ represent the latent states of the system, typically in the form of an N-D vector. And A, B, C, D here are the state, input, output, and feedforward matrices, defining the relationship between the input, output, and state vector. Following Gu et al. (Gu et al., 2022b), $\bar{D}$ is set equal to 0 since it can be replaced by the residual connection. Now, Equation 20 resembles an architecture similar to RNN, allowing us to recurrently compute x_k. Let the initial state be $x_{k-1} = 0$, and we can unroll Eq 20 as follows:

$$y_k = \overline{CA^kB}u_0 + \overline{CA^{k-1}B}u_1 + ... + \overline{CAB}u_{k-1} + \overline{CB}u_k \tag{20}$$

$$y = \overline{K}u, \quad \overline{K} = (\overline{CB}, \overline{CA^1B}, ..., \overline{CA^kB}). \tag{21}$$

Therefore, SSM can be transformed from the form of a recurrent neural network to a convolutional neural network. During training, the $\overline{K}$ can be considered as a 1-D globe convolution kernel, so the y can be calculated via the "long" convolution, allowing us to use the fast Fourier transform to efficiently compute the SSM convolutional kernel $\overline{K}$. However, directly computing the convolution in Equation 21 can be very expensive for long sequences. We can use the Fast Fourier Transform to accelerate it. The form of this convolution can be written as:

$$y = F_N^{-1} D_k F_N u, D_k = \text{diag}(\overline{K}F_N), \tag{22}$$

where F_N denotes the DFT matrix of size N. This FFT convolution has a computational complexity of $O(nlog(n))$. Following (Li et al., 2022; Fu et al., 2023), we hope to parameterize K directly rather than through $\{A, B, C\}$ because we can eliminate complex parameterization and accelerate the entire convolution.

While Eq. 22 shows that the SSM can be computed using FFT-based convolution, it is important to formalize the complexity guarantee. We now state the following proposition.

Proposition 1. *Let (A, B, C, D) denote the discretized state-space matrices of an S4 layer, with input sequence $u \in \mathbb{R}^N$. The output $y \in \mathbb{R}^N$ can be written as a linear convolution $y = k * u$ with kernel*

$$k_0 = D, \quad k_n = CA^{n-1}B, \quad n \geq 1.$$

Using the convolution theorem and the FFT, this convolution can be computed in time $\Theta(N \log N)$.

Proof. This proof follows the proof in *Lemma C.2* of Gu et al. (2021):

Expanding the recurrence

$$x_{n+1} = Ax_n + Bu_n, \qquad y_n = Cx_n + Du_n, \tag{23}$$

yields

$$y_n = Du_n + \sum_{t=0}^{n-1} CA^{n-1-t}B\, u_t = \sum_{t=0}^{n} k_{n-t}\, u_t, \tag{24}$$

where $k_0 = D$ and $k_n = CA^{n-1}B$. Thus $y = k * u$.

Let $\mathcal{F}_N$ denote the N-point DFT. By the convolution theorem,

$$\mathcal{F}_N(y) = \mathcal{F}_N(k) \odot \mathcal{F}_N(u), \tag{25}$$

so that

$$y = \mathcal{F}_N^{-1}(\mathcal{F}_N(k) \odot \mathcal{F}_N(u)) \,. \tag{26}$$

The cost of forward FFT, element-wise product, and inverse FFT is $\Theta(N \log N)$.

The S4 parameterization ensures A admits a diagonal-plus-low-rank (DPLR) structure, so the kernel takes the form

$$k_n = \sum_{j=1}^{r} \alpha_j\, \lambda_j^{n-1}, \qquad r \ll N. \tag{27}$$

Each exponential sequence can be generated recursively in $\mathcal{O}(N)$, yielding all $k_0, \ldots, k_{N-1}$ in linear time.

Combining the above,

$$T(N) = \underbrace{\Theta(N \log N)}_{\text{FFT convolution}} + \underbrace{\mathcal{O}(N)}_{\text{kernel generation}} = \Theta(N \log N). \tag{28}$$

D Theoretical Analysis of Decoupled Codebook Training

Proposition 2.1. *Let $X = (X_t, X_f)$ denote the temporal and frequency representations of an EEG segment, assumed approximately independent. Consider an additive reconstruction distortion $d(x, \hat{x}) = d_t(x_t, \hat{x}_t) + d_f(x_f, \hat{x}_f)$. Under a fixed total codebook size $K = 2^R$, a product codebook $\mathcal{C}_t \times \mathcal{C}_f$ with $|\mathcal{C}_t| = 2^{R_t}$ and $|\mathcal{C}_f| = 2^{R_f}$, $R_t + R_f = R$, achieves a minimum expected distortion*

$$D^\star_{\textit{prod}}(R) = \min_{R_t+R_f=R} \Big(D^\star_t(R_t) + D^\star_f(R_f) \Big), \tag{29}$$

which satisfies

$$D^\star_{\textit{mix}}(R) \;\geq\; D^\star_{\textit{prod}}(R), \tag{30}$$

where $D^\star_{\textit{mix}}(R)$ is the minimum distortion of a single mixed codebook $\mathcal{C}_{\textit{mix}} \subset \mathbb{R}^{d_t+d_f}$ of size 2^R.

Proof. The argument combines the separability of the rate–distortion (R–D) function for independent sources under additive distortion and the high-rate quantization approximation to the Shannon R–D limit.

Let $X = (X_t, X_f)$ with $X_t \perp X_f$, and distortion

$$d(x, \hat{x}) \;=\; d_t(x_t, \hat{x}_t) + d_f(x_f, \hat{x}_f). \tag{31}$$

Then the Shannon R–D function satisfies

$$R(D) \;=\; \min_{D_t+D_f=D} \Big(R_t(D_t) + R_f(D_f) \Big), \tag{32}$$

where $R_t(\cdot)$ and $R_f(\cdot)$ are the marginal R–D functions for X_t and X_f. By convex duality, the optimal test channel factorizes:

$$p(\hat{x}_t, \hat{x}_f \mid x_t, x_f) \;=\; p(\hat{x}_t \mid x_t)\, p(\hat{x}_f \mid x_f), \tag{33}$$

and the optimal distortion allocation solves

$$\min_{D_t, D_f} \; R_t(D_t) + R_f(D_f) \quad \text{s.t. } D_t + D_f = D, \tag{34}$$

with KKT condition $R'_t(D^\star_t) = R'_f(D^\star_f)$.

At rate R_t bits for X_t and R_f bits for X_f (with $R_t + R_f = R$), the minimal achievable distortions are $D^\star_t(R_t)$ and $D^\star_f(R_f)$. In the high-rate regime, practical vector quantizers approach these Shannon limits, yielding

$$D_{\text{prod}}(R_t, R_f) \;\approx\; D^\star_t(R_t) + D^\star_f(R_f). \tag{35}$$

Optimizing over all feasible splits gives

$$D^\star_{\text{prod}}(R) \;=\; \min_{R_t+R_f=R} \big(D^\star_t(R_t) + D^\star_f(R_f) \big). \tag{36}$$

Any mixed codebook $\mathcal{C}_{\text{mix}} \subset \mathbb{R}^{d_t+d_f}$ with $|\mathcal{C}_{\text{mix}}| = 2^R$ cannot beat the Shannon R–D limit, so

$$D^\star_{\text{mix}}(R) \;\geq\; D^\star(R). \tag{37}$$

But from (34)– (35),

$$D^\star(R) \;=\; \min_{R_t+R_f=R} \big(D^\star_t(R_t) + D^\star_f(R_f) \big) \;=\; D^\star_{\text{prod}}(R). \tag{38}$$

Therefore,

$$D^\star_{\text{mix}}(R) \;\geq\; D^\star_{\text{prod}}(R), \tag{39}$$

Remark. We acknowledge that theoretically, X_t and X_f are coupled via the Fourier Transform. However, within the context of Neural Vector Quantization, they behave as heterogeneous sources of information. For example, topological features in waveforms (e.g., K-complexes) and frequency densities in frequency (e.g., Alpha rhythms) impose orthogonal constraints on the codebook optimization landscape. Therefore, the ”approximate independence” in Proposition 2.1 should be interpreted as the functional independence of semantic distortions in the latent representation space, rather than the statistical independence of the raw signals.

E Pretraining Results

Our model follows a two-stage pretraining framework. In the first stage, we train the TFDual-Tokenizer, which independently tokenizes EEG signals in both the temporal and frequency domains. This tokenizer is optimized to reconstruct the original raw EEG signals, amplitude, and phase components, thereby producing discrete code representations with structural interpretability. In the second stage, we pretrain the EEGSSM encoder using a masked modeling objective: given the original EEG signals as input, the model learns to predict the corresponding masked tokens generated from the TFDual-Tokenizer.

This section reports the pretraining results of both stages, including loss convergence, reconstruction dynamics, and codebook utilization patterns.

E.1 TFDual-Tokenizer Pretraining Results

Total Training Loss The total pretraining loss curve of the TFDual-Tokenizer is shown in Figure 11. The model demonstrates a rapid initial decrease in loss during the first few epochs, followed by a slower but consistent decline.

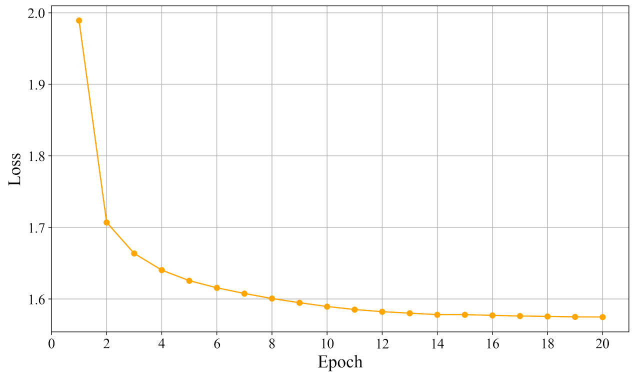


Figure 11: Pretraining Loss Curve of TFDual-Tokenizer.

Reconstruction Loss. We report the pretraining reconstruction loss of the TFDual-Tokenizer in Figure 12. The temporal codebook is trained to reconstruct raw EEG signals in the time domain, while the frequency codebook is trained to reconstruct the corresponding amplitude and phase components in the frequency domain. All three loss curves exhibit a sharp initial decline, followed by a gradual convergence, indicating stable optimization.

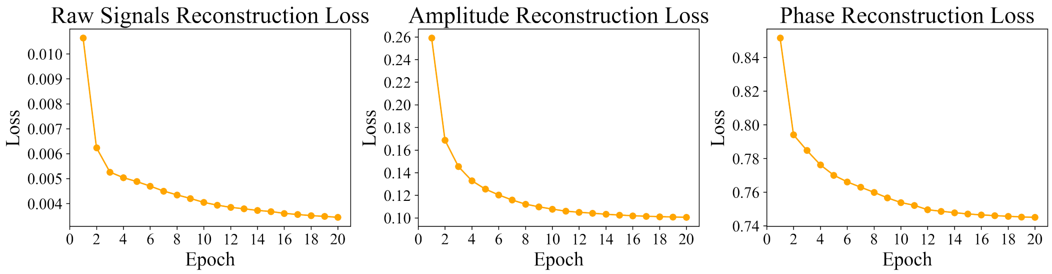


Figure 12: Pretraining Loss Curve of TFDual-Tokenizer.

Unused Codes Analysis. During pretraining, we track the number of unused codes in both the temporal and frequency codebooks of the TFDual-Tokenizer, each with a size of 4096. As shown in Figure 13, the frequency codebook demonstrates a rapid decrease in unused codes, while the temporal codebook shows a slower and more incremental reduction. A more detailed analysis of the temporal-frequency complementarity is provided in Section B.3.

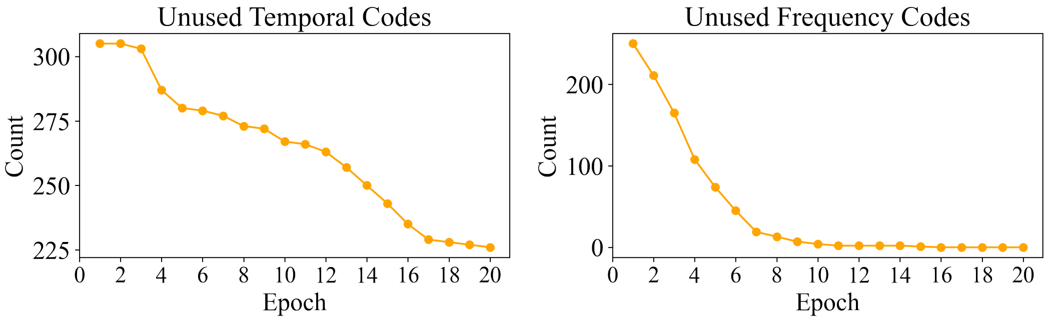


Figure 13: Unused code dynamics of the TFDUAL-TOKENIZER.

E.2 EEGSSSM PRETRAINING RESULTS

We plot the pretraining loss curve of EEGSSM in Figure 14. We select epoch 10 as the checkpoint for downstream fine-tuning. We observe that the pretraining loss of EEGSSM decreases rapidly from epoch 1 to 6 (9.04 → 6.39), then flattens gradually after epoch 10 (6.01 → 5.66). Using epoch 10 for fine-tuning is a balance between representation strength and generalization. Overtraining on EEG, prone to noise and inter-subject variability, can reduce transferability. Epoch 10 serves as a conservative yet effective checkpoint. This practice is consistent with trends in foundation models from NLP (Devlin et al., 2019) and vision (Caron et al., 2021), where mid-training checkpoints often lead to better downstream performance than final ones due to reduced overfitting to the pretext task.

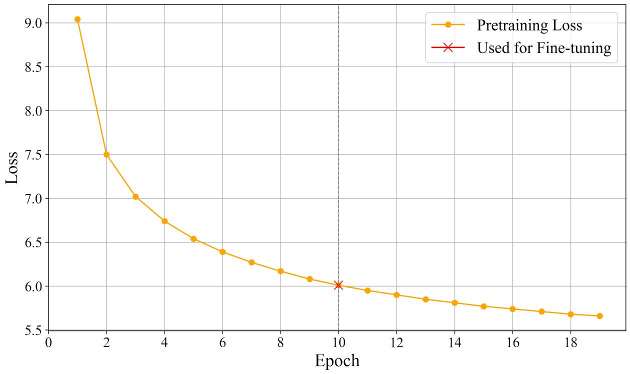


Figure 14: Pretraining Loss Curve of EEGSSM.

F IMPROVING TEMPORAL CODEBOOK LEARNING VIA CONTRASTIVE LOSS

To improve the learning of the temporal codebook in our TFDUAL-TOKENIZER, we introduce a contrastive loss as one of the objectives during pretraining. This design is motivated by observations from prior work LaBraM (Jiang et al., 2024), where the authors report that reconstructing raw EEG signals leads to unconvergence, and thus omit the temporal reconstruction objective entirely.

To better understand this limitation, we first implemented a baseline reconstruction of raw signals within the LaBraM framework and observed that the training loss plateaued at a high value (between 0.128 and 0.131), showing no convergence over training epochs. To mitigate this, we introduce a *TF-Conv* module before the Transformer encoder, designed to extract temporal-frequency representations before tokenization. While this stabilizes training to some extent, we still observe significant issues with code utilization and loss convergence. Therefore, we incorporate a lightweight contrastive loss, applied over temporal representations before quantization, to encourage the model to organize similar input patterns closer in the latent space. As shown in Figure 15, this improves the optimization of the reconstruction loss and reduces the number of unused temporal codes during training.

These results demonstrate that contrastive regularization acts as an effective prior for stabilizing discrete token learning, particularly when reconstructing raw signals. It both improves convergence and mitigates codebook collapse in the temporal branch of the tokenizer.

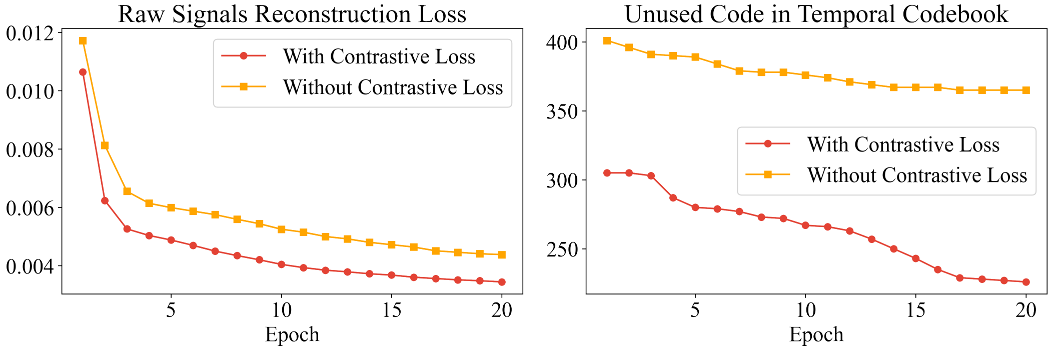


Figure 15: Effect of Contrastive Loss on Temporal Codebook Learning.

G Dataset Description

We evaluate the CodeBrain across eight diverse downstream tasks covering ten publicly available EEG datasets. Notably, these datasets exhibit substantial variability in channel configurations (ranging from 6 to 64), sequence lengths (from 1s to 30s), and task complexities (2-class to 9-class), highlighting the versatility and robustness of CodeBrain across different EEG applications. The following sections describe each dataset in detail, including its task objective and data split strategy. A comprehensive analysis of each dataset is provided below.

Emotion Recognition. We conduct emotion recognition experiments on two widely used EEG datasets: FACED (Chen et al., 2023) and SEED-V (Liu et al., 2021).
The **FACED** dataset (Finer-Grained Affective Computing EEG Dataset) is a large-scale EEG dataset proposed by Chen et al. (Chen et al., 2023) for emotion recognition tasks. It consists of 32-channel EEG recordings sampled at 250 Hz from 123 participants, each exposed to 28 video clips designed to elicit nine distinct emotional states: *amusement*, *inspiration*, *joy*, *tenderness*, *anger*, *fear*, *disgust*, *sadness*, and *neutral*. These cover both positive and negative affective categories. Each EEG trial is 10 seconds long and is subsequently resampled to 200 Hz, resulting in a total of 10,332 clean EEG segments. For fair comparison, we adopt the same subject-wise split as in (Wang et al., 2025): subjects 1–80 are used for training, 81–100 for validation, and 101–123 for testing, ensuring no subject overlap across splits and enabling evaluation of cross-subject generalization.
SEED-V (Liu et al., 2021) is an EEG dataset designed for emotion recognition, covering five emotional categories: *happy*, *sad*, *neutral*, *disgust*, and *fear*. It consists of 62-channel EEG recordings collected at 1000 Hz from 16 subjects, each participating in three sessions. Each session includes 15 trials, which are evenly divided into training, validation, and test sets (5 trials each). The EEG signals are segmented into 1-second windows, yielding a total of 117,744 samples, and resampled to 200 Hz for consistency. The dataset provides rich temporal structure and inter-subject variability, making it a strong benchmark for evaluating generalization in emotion-related EEG modeling.

Sleep Staging. We use two datasets, **ISRUC_S1** and **ISRUC_S3** (Khalighi et al., 2016), for sleep stage classification. Both datasets are annotated according to the American Academy of Sleep Medicine (AASM) standard (Berry et al., 2012), with five sleep stages: *Wake*, *NREM1 (N1)*, *NREM (N2)*, *NREM (N3)*, and *REM*. Each EEG segment corresponds to a 30-second epoch.
ISRUC_S1 includes EEG recordings from 100 subjects using six channels at a sampling rate of 200 Hz. We adopt a subject-wise split, with 80 subjects for training, 10 for validation, and 10 for testing. As the transition rules between sleep stages carry important temporal patterns, we follow

prior work (Wang et al., 2024a; 2025) and insert a Transformer layer on top of the projection head during fine-tuning to better capture sequence-level dependencies. We set the input sequence length to 20, and discard segments that cannot be evenly divided. In total, 86,320 labeled samples are retained. **ISRUC_S3** is a smaller dataset comprising recordings from 10 subjects, also sampled at 200 Hz with six channels, totaling 8,500 labeled segments. We follow an 8:1:1 subject-wise split for training, validation, and testing.

Imagined Speech Classification. The **BCIC2020-T3** dataset (Jeong et al., 2022) was released as part of the 2020 International Brain–Computer Interface Competition and focuses on imagined speech decoding. It contains EEG recordings from 15 participants who were instructed to silently imagine speaking five specific words or phrases, *"hello"*, *"help me"*, *"stop"*, *"thank you"*, and *"yes"*. EEG signals were collected using 64 scalp channels at a sampling rate of 256 Hz and were subsequently resampled to 200 Hz for preprocessing consistency. Each subject completed 80 trials per class, resulting in a total of 6,000 trials. The dataset provides predefined training, validation, and test splits, with 60, 10, and 10 trials per class, respectively, facilitating fair model evaluation like existing baselines (Wang et al., 2025).

Mental Stress Detection. The **Mental Arithmetic** dataset (Mumtaz, 2016) supports the task of mental stress detection using EEG signals. It contains recordings from 36 subjects under two distinct cognitive conditions: *resting* and *active engagement* in mental arithmetic. EEG data labeled as “no stress” correspond to resting periods prior to the task, while “stress” labels are assigned to recordings during task performance. The signals were acquired using 20 electrodes placed according to the international 10–20 system, with an original sampling rate of 500 Hz. For consistency, the signals are resampled to 200 Hz and band-pass filtered between 0.5–45 Hz to suppress noise. Each recording is segmented into 5-second windows, yielding a total of 1,707 samples. We adopt a subject-wise split for fair evaluation with existing baselines (Wang et al., 2025): subjects 1–28 for training, 29–32 for validation, and 33–36 for testing.

Seizure Detection. The **CHB-MIT** dataset (Shoeb, 2009) is a widely used benchmark for seizure detection from EEG signals. It contains long-term EEG recordings from 23 patients diagnosed with intractable epilepsy, collected at the Children’s Hospital Boston. The subjects underwent continuous monitoring over several days, during which seizures were recorded following the tapering of anti-epileptic medications. EEG signals were acquired using the international 10–20 system and originally sampled at 256 Hz. In our setting, we adopt 16 channels commonly used in prior work (Yang et al., 2023; Wang et al., 2025), resample all signals to 200 Hz, and segment them into 10-second non-overlapping windows, yielding 326,993 labeled samples across seizure and non-seizure classes. We follow a subject-wise split: subjects 1–19 for training, 20–21 for validation, and 22–23 for testing. Notably, this dataset is highly imbalanced, with seizure events constituting only a small fraction of the total samples, posing significant challenges for model training and evaluation.

Motor Imagery Classification. The **SHU-MI** dataset (Goldberger et al., 2000) is designed for binary motor imagery classification, where participants are instructed to imagine movements of either the *left or right hand*. EEG signals were recorded from 25 subjects using a 32-channel setup at an original sampling rate of 250 Hz. To ensure consistency with the pre-training setting, all signals are resampled to 200 Hz and segmented into 4-second non-overlapping windows, resulting in 11,988 labeled samples. A subject-wise split is applied for fair model evaluation like existing baselines (Wang et al., 2025), with subjects 1–15 used for training, 16–20 for validation, and 21–25 for testing. This dataset supports the development of BCI systems that decode motor intentions from brain activity without actual movement.

Event Type Classification. The **TUEV** dataset (Obeid & Picone, 2016) is a clinically annotated EEG corpus used for multi-class event type classification. It includes six event categories: *spike and sharp wave (SPSW)*, *generalized periodic epileptiform discharges (GPED)*, *periodic lateralized epileptiform discharges (PLED)*, *eye movements (EYEM)*, *artifacts (ARTF)*, and *background activity (BCKG)*. EEG signals were originally recorded at 256 Hz using 23 channels. In line with prior work (Wang et al., 2025), we preprocess the data by selecting 16 bipolar montage channels based on the international 10–20 system. The signals are band-pass filtered between 0.3–75 Hz to suppress low- and high-frequency noise, and a 60 Hz notch filter is applied to eliminate power line interference. All recordings are resampled to 200 Hz and segmented into 5-second windows, yielding 112,491 labeled samples. We follow the official train-test split and further divide the training subjects into training and validation sets in an 8:2 ratio, consistent with established benchmarks.

Abnormal Detection. The **TUAB** dataset (Obeid & Picone, 2016) is employed for binary abnormal EEG detection, where each EEG recording is labeled as either *normal* or *abnormal* based on clinical interpretation. Originally recorded at 256 Hz using 23 channels, the dataset provides large-scale EEG recordings suitable for evaluating diagnostic models. To ensure fair comparison with prior work (Wang et al., 2025), we follow a similar preprocessing protocol. Specifically, we select 16 bipolar montage channels following the international 10–20 system, apply band-pass filtering between 0.3–75 Hz to eliminate low- and high-frequency artifacts, and remove 60 Hz power line interference using a notch filter. The EEG signals are then resampled to 200 Hz and segmented into 10-second windows, resulting in 409,455 labeled samples. We follow the official train-test split and further divide the training set into training and validation subsets using an 8:2 subject-wise ratio, consistent with existing benchmarks.

H Baselines and Metrics Description

H.1 Metrics

To comprehensively evaluate our model, we compare it with a set of strong baselines commonly used in EEG analysis. These baselines are evaluated using metrics tailored for class-imbalanced scenarios, which are prevalent in EEG datasets. The metrics include:

- **Balanced Accuracy**, which averages the recall across all classes and is particularly suitable for imbalanced multi-class classification tasks.
- **AUROC** and **AUC-PR**, which assess the performance of binary classifiers under different thresholds. While AUROC measures the trade-off between sensitivity and specificity, AUC-PR focuses on precision-recall trade-offs, especially informative under severe class imbalance.
- **Cohen's Kappa**, which quantifies inter-class agreement beyond chance and is employed as the primary metric for multi-class classification.
- **Weighted F1 Score**, which combines precision and recall while adjusting for class support, ensuring fair performance measurement across imbalanced datasets.

For model selection and comparison, AUROC is used as the main evaluation metric for binary classification tasks, and Cohen's Kappa is used for multi-class scenarios.

H.2 Baselines

We compare our CodeBrain model against a comprehensive set of baseline models that include widely used task-specific models, as well as publicly available EEG foundation models.

EEGNet (Lawhern et al., 2018): EEGNet is a compact convolutional neural network specifically designed for EEG-based BCI tasks. It adopts depthwise-separable convolutions to disentangle temporal filtering and spatial pattern learning, enabling efficient parameter usage while preserving discriminative EEG features.

EEGConformer (Song et al., 2022): EEGConformer integrates convolutional front-ends with Transformer blocks to jointly capture local temporal dynamics and longer-range dependencies. Its convolution modules extract short-term EEG patterns, while the attention mechanism models cross-channel and cross-time interactions.

ContraWR (Yang et al., 2021): ContraWR is a self-supervised representation learning framework that uses contrastive learning with a weakly-supervised relational task. By contrasting EEG segments from the same versus different contexts, the model learns invariant temporal representations without relying on explicit labels.

ST-Transformer (Song et al., 2021): ST-Transformer applies Transformer attention to EEG by factorizing spatial and temporal modeling. It processes EEG as a structured sequence across both dimensions, where attention layers capture inter-channel relationships as well as time-varying dependencies.

We also compare CodeBrain against 5 publicly available EFMs that have released pre-trained weights, covering a diverse set of pretraining strategies to evaluate the effectiveness of different foundation model designs and pretraining paradigms under comparable settings.

BENDR (Kostas et al., 2021): We adopted **BENDR (Bert-inspired Neural Data Representations)** as our baseline model, as introduced by Kostas et al. BENDR is a pioneering deep learning architecture for Electroencephalography (EEG) data, leveraging transformers and a contrastive self-supervised learning task. This approach enables the model to learn meaningful representations from vast amounts of unlabeled EEG data.

BIOT (Yang et al., 2023): **BIOT (Biosignal Transformer for Cross-data Learning in the Wild)** is a transformer-based architecture designed to handle cross-dataset EEG signal classification under domain shifts. It leverages a domain-invariant attention mechanism and contrastive representation learning to enhance generalization across different recording conditions and subject populations.

LaBraM(Jiang et al., 2024): **LaBraM (Large Brain Model)** proposes a scalable transformer-based framework designed to learn generic EEG representations from large-scale brain signal datasets. By pretraining on a diverse corpus of EEG recordings, the model captures rich temporal and spatial features that transfer effectively to various downstream BCI tasks. The architecture incorporates efficient self-attention mechanisms and task-specific adapters to support flexible fine-tuning.

EEGPT (Wang et al., 2024a): **EEGPT** employs a dual self-supervised learning strategy that combines masked autoencoding with spatial-temporal representation alignment, enhancing feature quality by focusing on high signal-to-noise ratio (SNR) representations rather than raw signals. The model's hierarchical architecture decouples spatial and temporal processing, improving computational efficiency and adaptability to various brain-computer interface (BCI) applications.

CBraMod (Wang et al., 2025): **CBraMod (Criss-Cross Brain Foundation Model)** is a transformer-based EEG foundation model that addresses the heterogeneous spatial and temporal dependencies inherent in EEG signals. It introduces a criss-cross transformer architecture comprising parallel spatial and temporal attention mechanisms, enabling separate yet simultaneous modeling of spatial and temporal relationships.

I Hyperparameter Setting

We provide detailed hyperparameter configurations for the two-stage pretraining of our CodeBrain model and the fine-tuning settings across ten downstream tasks.

I.1 Pretraining Settings

The pretraining process consists of two stages:

1. Training the TFDual-Tokenizer
2. Training the EEGSSM

The hyperparameters used in each stage are summarized in Table 5 and Table 6, respectively.

I.2 Parameters of EEGSSM

The model architecture parameters used in EEGSSM during pre-training are shown in Table 7.

I.3 Fine-tuning Settings on Downstream Tasks

The CodeBrain model is fine-tuned on ten downstream EEG classification tasks, each with task-specific hyperparameters. Following the general strategy adopted by prior EFMs, we adopt a lightweight three-layer MLP as the probe head for all downstream tasks and fine-tune the entire model end-to-end. Table 8 lists the fine-tuning configurations including learning rate, weight decay, dropout

Table 5: Hyperparameters for TFDual-Tokenizer.

Hyperparameters	**Values**
TFConv	
Input channels	{1, 8, 4}
Output channels	{8, 4, 4}
Kernel size	{(1, 15), (1, 3), (1, 3)}
Stride	{(1, 8), (1, 1), (1, 1)}
Padding	{(0, 7), (0, 1), (0, 1)}
Transformer encoder layers	12
Transformer decoder layers	3
Hidden size	200
MLP size	800
Attention head number	8
Temporal Codebook size	4096 × 32
Frequency Codebook size	4096 × 32
Codebook initialization	Random init + L^2 normalization
Batch size	256
Peak learning rate	1e-4
Minimal learning rate	1e-5
Learning rate scheduler	Cosine
Optimizer	AdamW
Adam β	(0.9, 0.99)
Weight decay	1e-4
Warm-up steps	5
Total epochs	20
Data stride	200
Contrastive temperature (τ)	0.5

Table 6: Hyperparameters of Pre-training.

Hyperparameters	**Values**
Epochs	10
Batch size	256
Dropout	0.1
Optimizer	Adam
Learning rate	1e-4
Adam β	(0.9, 0.999)
Adam ϵ	1e-8
Weight decay	5e-3
Scheduler	CosineAnnealingLR
Minimal learning rate	1e-5
Clipping gradient norm	5

rate, and batch size for each task. For sleep staging tasks, due to their strong temporal structure, we follow prior work (Wang et al., 2024a; 2025) and insert an additional Transformer encoder on top of the projection head to jointly model the sequence of 20 consecutive EEG segments. This enables the model to capture inter-epoch transitions critical to sleep stage classification.

J Additional Evaluation on Other BCI Tasks

We report the performance of CodeBrain on four additional EEG datasets not included in the main text, covering diverse domains of sleep staging, motor imagery, event detection, and abnormality

Table 7: Configuration of EEGSSM

Parameters	Values
Input size	200
Hidden dimension	200
Output size	200
Number of layers	8
Max sequence length	570
SGConv state	64
SGConv bidirectional	True
Layer normalization	True
Sliding window attention length	1s

Table 8: Fine-tuning Hyperparameters for Downstream Tasks.

Dataset	Learning Rate	Weight Decay	Dropout	Batch Size
FACED	5e-5	5e-4	0.1	16
SEED-V	5e-5	1e-2	0.1	64
ISRUC_S1	1e-4	1e-1	0.2	48
ISRUC_S3	1e-4	1e-1	0.2	48
BCIC2020-T3	5e-5	5e-2	0.1	32
Mental Arithmetic	3e-5	1e-3	0.1	32
CHB-MIT	3e-5	1e-2	0.4	64
SHU-MI	5e-5	5e-3	0.3	64
TUEV	2e-5	5e-4	0.3	64
TUAB	1e-5	5e-5	0.4	512

classification in Tables 9 to 12. These allow us to assess the cross-domain generalization ability of our pretrained model beyond the main text.

We note that both TUAB and TUEV originate from the TUH EEG corpus (Obeid & Picone, 2016), which overlaps with our pretraining source (TUEG). To avoid overfitting to this distribution and promote generalization, we stop pretraining at epoch 10 as discussed in Section E.2. While this may limit gains on TUH datasets compared to previous EFM, such as CBraMod (trained for 40 epochs in the same pretraining dataset) (Wang et al., 2025), CodeBrain still achieves superior or competitive results.

ISRUC_S1 As shown in Table 9, CodeBrain achieves state-of-the-art performance on ISRUC_S1 in terms of Cohen's Kappa (0.7476) and Weighted F1 (0.8020), slightly surpassing CBraMod (Wang et al., 2025) by +0.34 and +0.09 points, respectively. Its Balanced Accuracy of 0.7835 is also competitive, trailing the best result by only -0.30. These results highlight the model's ability to capture temporal dependencies and learn discriminative representations for 5-class sleep staging under a cross-subject setting.

SHU-MI As shown in Table 10, CodeBrain achieves the best overall performance on SHU-MI across all three metrics. It obtains an AUROC of 0.7124 and an AUC-PR of 0.7166, slightly improving over the previous best by +1.36 and +0.27 points, respectively. For Balanced Accuracy, it reaches 0.6431 (+0.61), with notably lower variance. These results underscore its strong generalization to motor imagery decoding under a cross-subject protocol.

TUEV As shown in Table 11, CodeBrain achieves the highest Cohen's Kappa (0.6912, an improvement of +0.0140 over the best baseline (Wang et al., 2025)) and Weighted F1 (0.8362) on TUEV. Although its Balanced Accuracy is lower than CBraMod (Wang et al., 2025), we attribute this to reduced sensitivity on the rare *SPSW* class. Since TUEV shares distributional overlap with our pretraining source (TUEG), we stop pretraining at epoch 10 to prevent overfitting, unlike CBraMod's 40-epoch training as discussed in subsection E.2. We also report LaBraM's results based on its original

Table 9: Performance Comparison on the ISURC_S1 (5-Class) dataset.

Methods	Cohen's Kappa	Weighted F1	Balanced Accuracy
EEGNet	0.7040 ± 0.0173	0.7513 ± 0.0124	0.7154 ± 0.0121
EEGConformer	0.7143 ± 0.0162	0.7634 ± 0.0151	0.7400 ± 0.0133
ContraWR	0.7178 ± 0.0156	0.7610 ± 0.0137	0.7402 ± 0.0126
ST-Transformer	0.7013 ± 0.0352	0.7681 ± 0.0175	0.7381 ± 0.0205
BENDRKostas et al. (2021)	0.6956 ± 0.0053	0.7569 ± 0.0049	0.7401 ± 0.0056
BIOTYang et al. (2023)	0.7192 ± 0.0231	0.7790 ± 0.0146	0.7527 ± 0.0121
LaBraMJiang et al. (2024)	0.7231 ± 0.0182	0.7810 ± 0.0133	0.7633 ± 0.0102
EEGPTWang et al. (2024a)	0.2223 ± 0.0227	0.3111 ± 0.0110	0.4012 ± 0.0177
CBraModWang et al. (2025)	0.7442 ± 0.0152	0.8011 ± 0.0099	**0.7865** ± 0.0110
CodeBrain	**0.7476** ± 0.0040	**0.8020** ± 0.0018	0.7835 ± 0.0033

Table 10: Performance Comparison on the SHU-MI (2-Class) dataset.

Methods	AUROC	AUC-PR	Balanced Accuracy
EEGNet	0.6283 ± 0.0152	0.6311 ± 0.0142	0.5889 ± 0.0177
EEGConformer	0.6351 ± 0.0101	0.6370 ± 0.0093	0.5900 ± 0.0107
ContraWR	0.6273 ± 0.0113	0.6315 ± 0.0105	0.5873 ± 0.0128
ST-Transformer	0.6431 ± 0.0111	0.6394 ± 0.0122	0.5992 ± 0.0206
BENDRKostas et al. (2021)	0.5863 ± 0.0280	0.5853 ± 0.0268	0.5573 ± 0.0227
BIOTYang et al. (2023)	0.6609 ± 0.0127	0.6770 ± 0.0119	0.6179 ± 0.0183
LaBraMJiang et al. (2024)	0.6604 ± 0.0091	0.6761 ± 0.0083	0.6166 ± 0.0192
EEGPTWang et al. (2024a)	0.6241 ± 0.0071	0.6266 ± 0.0133	0.5778 ± 0.0162
CBraModWang et al. (2025)	0.6988 ± 0.0068	0.7139 ± 0.0088	0.6370 ± 0.0151
CodeBrain	**0.7124** ± 0.0050	**0.7166** ± 0.0106	**0.6431** ± 0.0066

23-channel setting (Jiang et al., 2024), while CodeBrain follows the 16-channel configuration used in CBraMod. Similarly, EEGPT (Wang et al., 2024a) does not adopt a linear fine-tuning protocol but applies two convolutional layers before entering the foundation model, followed by an MLP head. While such architectural choices may enhance performance, we follow their respective fine-tuning settings. In addition, following the experimental setup of CBraMod, we also report the results of experiments conducted by removing all TUEV and TUAB samples from the TUEG dataset. It can be seen that although our model's performance slightly declined on TUEV, it still surpassed other baselines. In such cases, CodeBrain still outperforming both LaBraM and EEGPT under their own fine-tuning settings clearly demonstrates the robustness of our approach.

TUAB As shown in Table 12, CodeBrain achieves the highest Balanced Accuracy (0.8294) on TUAB, slightly outperforming CBraMod (Wang et al., 2025). Similar to TUEV, TUAB is part of the TUH EEG corpus family (Obeid & Picone, 2016) and thus closely aligned with our pretraining source (TUEG). As discussed in subsection E.2, we adopt an early stopping strategy at epoch 10 to mitigate overfitting to this distribution, which may partly account for the slightly lower AUROC and AUC-PR compared to CBraMod, trained for 40 epochs on the same dataset. While LaBraM leverages a 23-channel montage (Jiang et al., 2024) and EEGPT (Wang et al., 2024a) employs two convolutional layers before the foundation model, we retain their respective fine-tuning protocols for comparison. In addition, the impact of duplicate data in the pre-training set on our model is smaller on the TUAB dataset, and in some experiments it can even surpass situations without leaking. This may be due to the larger size of the TUAB dataset. Despite these potentially stronger configurations, CodeBrain still exceeds both models under their own settings, highlighting its strong and consistent generalization.

Table 11: Performance Comparison on the TUEV (6-Class) dataset.

Methods	Cohen's Kappa	Weighted F1	Balanced Accuracy
EEGNet	0.3577 ± 0.0155	0.6539 ± 0.0120	0.3876 ± 0.0143
EEGConformer	0.3967 ± 0.0195	0.6983 ± 0.0152	0.4074 ± 0.0164
ContraWR	0.3912 ± 0.0237	0.6893 ± 0.0136	0.4384 ± 0.0349
ST-Transformer	0.3765 ± 0.0306	0.6823 ± 0.0190	0.3984 ± 0.0228
BENDRKostas et al. (2021)	0.4271 ± 0.0238	0.6755 ± 0.0216	0.4363 ± 0.0245
BIOTYang et al. (2023)	0.5273 ± 0.0249	0.7492 ± 0.0082	0.5281 ± 0.0225
LaBraMJiang et al. (2024)	0.6637 ± 0.0093	0.8312 ± 0.0052	0.6409 ± 0.0065
EEGPTWang et al. (2024a)	0.6351 ± 0.0134	0.8187 ± 0.0063	0.6232 ± 0.0114
CBraModWang et al. (2025)	0.6772 ± 0.0096	0.8342 ± 0.0064	**0.6671** ± 0.0107
CodeBrain (Excluding)	0.6838 ± 0.0291	0.8293 ± 0.0163	0.6375 ± 0.0182
CodeBrain	**0.6912** ± 0.0101	**0.8362** ± 0.0048	0.6428 ± 0.0062

Table 12: Performance Comparison on the TUAB (2-Class) dataset.

Methods	Balanced Accuracy	AUC-PR	AUROC
EEGNet	0.7642 ± 0.0036	0.8299 ± 0.0043	0.8412 ± 0.0031
EEGConformer	0.7758 ± 0.0049	0.8427 ± 0.0054	0.8445 ± 0.0038
ContraWR	0.7746 ± 0.0041	0.8421 ± 0.0104	0.8456 ± 0.0074
ST-Transformer	0.7966 ± 0.0023	0.8521 ± 0.0026	0.8707 ± 0.0019
BENDRKostas et al. (2021)	0.7714 ± 0.0248	0.8412 ± 0.0215	0.8426 ± 0.0237
BIOTYang et al. (2023)	0.7959 ± 0.0057	0.8792 ± 0.0023	0.8815 ± 0.0043
LaBraMJiang et al. (2024)	0.8140 ± 0.0019	0.8965 ± 0.0016	0.9022 ± 0.0009
EEGPTWang et al. (2024a)	0.8038 ± 0.0040	0.8891 ± 0.0018	0.8811 ± 0.0015
CBraModWang et al. (2025)	0.8289 ± 0.0022	**0.9258** ± 0.0008	**0.9227** ± 0.0011
CodeBrain (Excluding)	0.8288 ± 0.0064	0.9061 ± 0.0039	0.9012 ± 0.0020
CodeBrain	**0.8294** ± 0.0013	0.9100 ± 0.0006	0.9030 ± 0.0009

K Ablation on Design Choices

K.1 Ablation on Mask Ratio

We conduct an ablation study to investigate the effect of the mask ratio in the EEGSSM pretraining framework. As shown in Tables 13–15 and Figure 16, downstream performance consistently exhibits a U-shaped trend with respect to the masking ratio across all three datasets: FACED, SEED-V, and ISRUC_S3. Moderate masking (e.g., ratios around 0.4–0.6) leads to optimal performance, whereas excessively low (e.g., 0.1) or high (e.g., 0.9) ratios degrade generalization.

To further illustrate this pattern, we visualize the training loss curves across different mask ratios in Figure 17. Interestingly, higher mask ratios result in slower convergence and higher final training loss, which is expected due to the increased difficulty of the reconstruction task. In contrast, lower mask ratios lead to faster and smoother loss reduction, but do not necessarily yield better downstream performance. This observation suggests a possible *optimization-vs-generalization trade-off*: easier pretext tasks (low mask ratio) are more optimizable but may encourage the model to learn shortcut solutions with limited generalizability, while overly difficult tasks (high mask ratio) may hinder effective representation learning due to insufficient learning signal. Moderate masking strikes a balance by being sufficiently challenging to promote abstraction, while still being learnable, thereby facilitating better generalization across downstream tasks.

Table 13: Performance of CodeBrain on FACED Dataset under Different Mask Ratios.

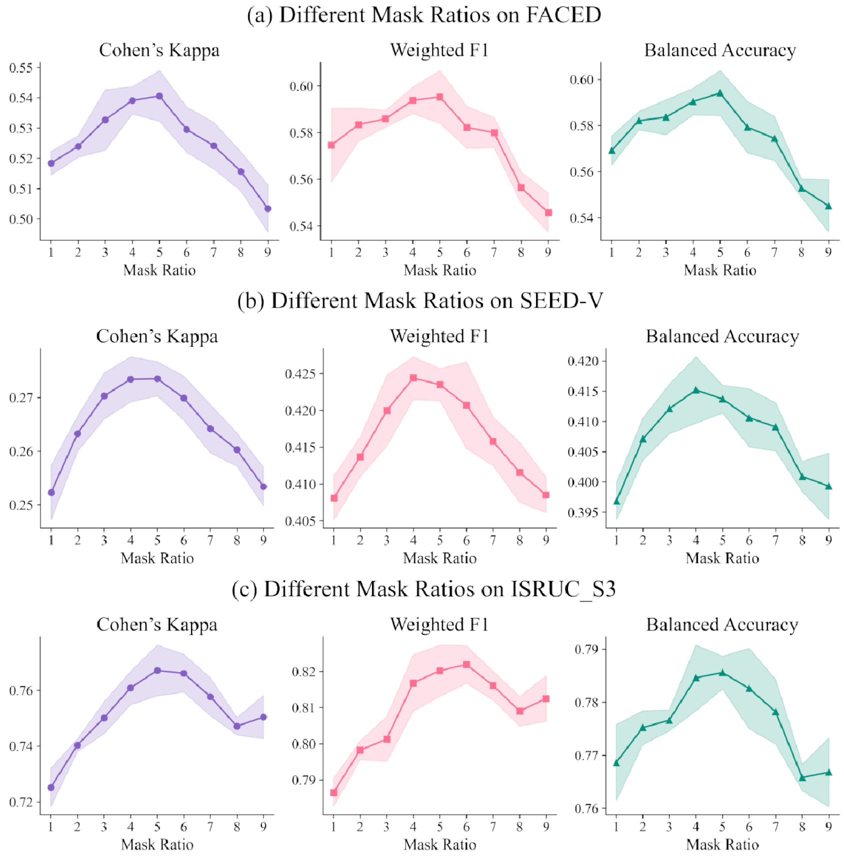
Mask Ratio	Cohen's Kappa	Weighted F1	Balanced Accuracy
0.1	0.5184 ± 0.0039	0.5746 ± 0.0158	0.5692 ± 0.0064
0.2	0.5239 ± 0.0036	0.5834 ± 0.0071	0.5821 ± 0.0040
0.3	0.5327 ± 0.0100	0.5859 ± 0.0036	0.5836 ± 0.0076
0.4	0.5391 ± 0.0045	0.5938 ± 0.0057	0.5904 ± 0.0058
0.5	**0.5406 ± 0.0084**	**0.5953 ± 0.0113**	**0.5941 ± 0.0098**
0.6	0.5295 ± 0.0075	0.5822 ± 0.0090	0.5793 ± 0.0112
0.7	0.5242 ± 0.0077	0.5800 ± 0.0065	0.5744 ± 0.0096
0.8	0.5157 ± 0.0065	0.5564 ± 0.0067	0.5528 ± 0.0040
0.9	0.5034 ± 0.0078	0.5457 ± 0.0084	0.5451 ± 0.0114

Table 14: Performance of CodeBrain on SEED-V Dataset under Different Mask Ratios.

Mask Ratio	Cohen's Kappa	Weighted F1	Balanced Accuracy
0.1	0.2523 ± 0.0051	0.4081 ± 0.0030	0.3968 ± 0.0031
0.2	0.2633 ± 0.0033	0.4137 ± 0.0027	0.4071 ± 0.0035
0.3	0.2703 ± 0.0043	0.4200 ± 0.0048	0.4121 ± 0.0041
0.4	0.2734 ± 0.0042	**0.4244** ± 0.0029	**0.4142** ± 0.0055
0.5	**0.2735** ± 0.0032	0.4235 ± 0.0022	0.4137 ± 0.0023
0.6	0.2699 ± 0.0041	0.4207 ± 0.0059	0.4106 ± 0.0048
0.7	0.2642 ± 0.0045	0.4158 ± 0.0033	0.4091 ± 0.0040
0.8	0.2603 ± 0.0031	0.4116 ± 0.0041	0.4009 ± 0.0025
0.9	0.2534 ± 0.0036	0.4085 ± 0.0024	0.3993 ± 0.0055

Table 15: Performance of CodeBrain on ISRUC_S3 Dataset under Different Mask Ratios.

Mask Ratio	Cohen's Kappa	Weighted F1	Balanced Accuracy
0.1	0.7252 ± 0.0069	0.7865 ± 0.0039	0.7686 ± 0.0072
0.2	0.7403 ± 0.0023	0.7982 ± 0.0026	0.7752 ± 0.0032
0.3	0.7501 ± 0.0057	0.8013 ± 0.0062	0.7766 ± 0.0020
0.4	0.7608 ± 0.0060	0.8168 ± 0.0078	0.7846 ± 0.0062
0.5	**0.7671** ± 0.0091	0.8202 ± 0.0071	**0.7856** ± 0.0031
0.6	0.7661 ± 0.0068	**0.8219** ± 0.0052	0.7826 ± 0.0076
0.7	0.7577 ± 0.0068	0.8161 ± 0.0040	0.7782 ± 0.0061
0.8	0.7471 ± 0.0031	0.8090 ± 0.0041	0.7658 ± 0.0025
0.9	0.7504 ± 0.0078	0.8125 ± 0.0063	0.7668 ± 0.0065


(a) Different Mask Ratios on FACED

(b) Different Mask Ratios on SEED-V

(c) Different Mask Ratios on ISRUC_S3

Figure 16: Performance Across Different Mask Ratios on FACED, SEED-V, and ISRUC_S3.

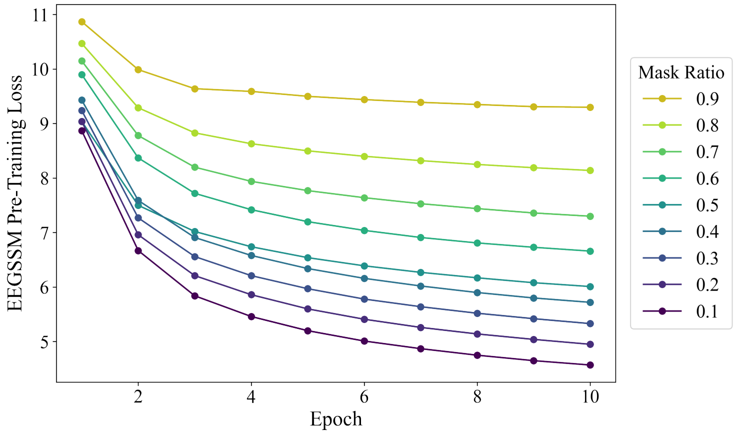


Figure 17: EEGSSM Pre-Training Loss Curve for Different Mask Ratios.

K.2 Ablation on SWA Window Size

We conduct an ablation study to investigate the effect of the SWA window size in the EEGSSM framework. The window size of SWA means the length of the segment observed by the attention mechanism in the SWA mechanism. Usually, the window size of SWA is an odd number because the model generally focuses not only on a single segment but also needs to observe features in adjacent segments. Therefore, its size is usually $2n + 1$, where n represents the length of SWA. As shown in Figure 18 and Figure 19, SWA window size = 1 achieved the best performance on both datasets among the 5 SWA window sizes. With the increase of the SWA window size, the SWA is close to self-attention. Therefore, as the SWA window size increases, the model's performance actually improves because more data will be involved in the attention calculation. When the SWA window size is 1, it is a special case where SWA only calculates for one second. This is equivalent to the model performing attention mechanism calculations within one second. Overall, SWA can help the model achieve a certain performance improvement.

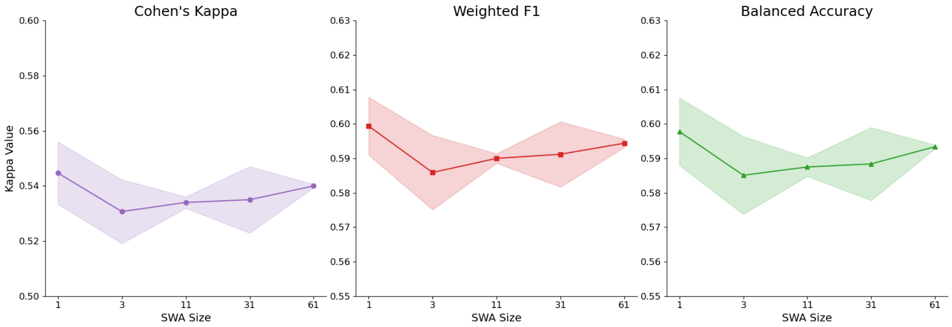


Figure 18: Performance of different SWA window sizes on the FACED dataset.

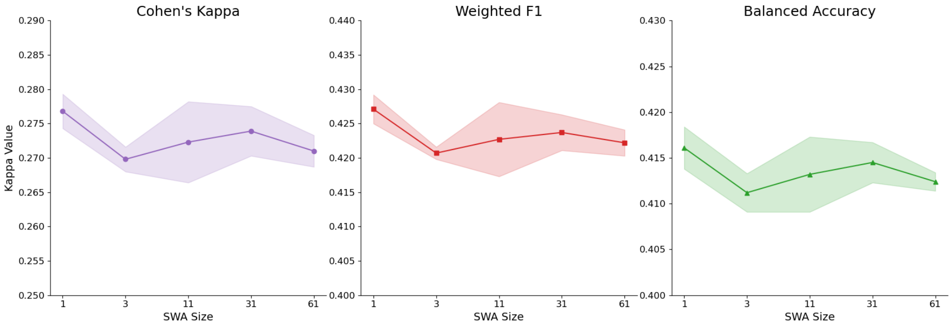


Figure 19: Performance of different SWA window sizes on the SEED-V dataset.

K.3 Ablation on Codebook Size

The size of the codebook is an important parameter; a codebook that is too large may lead to unstable training, while a codebook size that is too small may result in mixed information. Ideally, the codebook should maintain a small amount of unused codes but not be 0. We tested several combinations of different time-domain and frequency-domain codebook sizes to observe their unused codes during Tokenizer training.

Table 16 shows the unused temporal codes and unused frequency codes under different codebook size combinations. When both the Temporal codebook size and Frequency codebook size are set to 2048, the unused frequency code is 0, indicating that there are duplicate frequency codes in the current codebook. When choosing a codebook size of 8192, the unused temporal codes reached 3401 and the unused frequency codes reached 1260, indicating that a large number of codes in the completed training codebook were not used. We ultimately selected 4096 for both. Although the frequency domain often yields richer representations, enlarging its codebook may increase reliance on it, so we kept temporal and frequency codebooks equal. Considering capacity and utilization, 4096–4096 offers the best trade-off.

Table 16: Temporal and frequency codebook statistics

Temporal Codebook Size	**Frequency Codebook Size**	**Unused Temporal Codes**	**Unused Frequency Codes**
2048	2048	12	0
2048	4096	116	0
4096	**4096**	225	165
4096	8192	321	931
8192	8192	3401	1260

K.4 Ablation on Patch Size

The size of the patch window is also an important adjustable parameter, which affects temporal resolutions and masking strategies. To explore the impact of patch window size on the model, we used window sizes ranging from 0.5s to 5s for complete two-stage pre-training and full-parameter fine-tuning. Note that for some datasets where the patch size is larger than the channel length, such as SEED-V, we pad the portion exceeding the available data to match the patch size in this experiment.

Table 17 and Table 18 show the performance of our method with different patch sizes on the SEED-V dataset and ISRUC_3 dataset. Notably, in SEED-V, patch lengths longer than 1s require heavy padding, causing large performance drops; in ISRUC_S3, the shortest 0.5s patches achieve the worst performance, likely because they fragment key waveforms in sleep staging (e.g., Spindle ≥0.5s). From these results, the 1s setting is supported by two key considerations:

Broad compatibility with downstream task. 1s is a divisor of most downstream sequence lengths (1–30s), minimizing padding and ensuring transferability. For example, on the SEED-V dataset, if the patch size chosen by the model is greater than 1s, some methods (such as padding) need to be adopted to enable model training. These methods may usually impair the model's performance because they introduce additional noise or increase computational load. EEG datasets with durations less than 1 second are relatively rare, as most datasets have at least 1 second of data. If the data does not exactly match the whole seconds, the cost of processing such a dataset is also relatively small. Prior EEG foundation models (e.g., LaBram (Jiang et al., 2024), CBraMod (Wang et al., 2025)) also adopt 1s patches for this reason.

Semantic integrity. Choosing a 1s patch length preserves the natural structure of EEG waveforms and prevents semantic fragmentation. Many physiologically meaningful EEG events have characteristic durations: for example, K-complexes are typically around 1s, spindles last 0.5–2s, and event-related potentials such as P300 occur in the range of 0.3–0.6s. If patches are shorter than these characteristic scales, the temporal branch of the TFDual-Tokenizer may only capture partial fragments of these waveforms, leading to loss of semantic context.

Table 17: Performance on SEED-V with different patch sizes

Patch Length	Cohen's Kappa	Weighted F1	Balanced Accuracy
0.5s	0.2640±0.0035	0.4035±0.0025	0.3841±0.0025
1s	**0.2735**±0.0032	**0.4235**±0.0022	**0.4137**±0.0023
2s	0.1545±0.0046	0.3313±0.0036	0.3279±0.0049
5s	0.1557±0.0043	0.3340±0.0029	0.3271±0.0045

K.5 Ablation on SGConv Kernel Parameters

To justify the choice of the SGConv decay coefficient, we conduct a sensitivity analysis on three representative downstream datasets (FACED, SEED-V, ISRUC_S3). The results are shown in Table 19

Table 18: Performance on ISRUC_3 with different patch sizes

Patch Length	Cohen's Kappa	Weighted F1	Balanced Accuracy
0.5s	0.7405±0.0102	0.7950±0.0097	0.7420±0.0081
1s	**0.7671**±0.0091	**0.8202**±0.0071	**0.7856**±0.0031
2s	0.7592±0.0079	0.8113±0.0081	0.7753±0.0087
5s	0.7601±0.0075	0.8096±0.0074	0.7791±0.0096

Table 19: Sensitivity analysis of the SGConv decay coefficient α across three downstream datasets.

SEED-V	Cohen's Kappa	Weighted F1	Balanced Accuracy
$\alpha = 0.5$ **(default)**	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023
$\alpha = 0.1$	0.2629 ± 0.0017	0.4159 ± 0.0006	0.4044 ± 0.0017
$\alpha = 0.9$	0.2318 ± 0.0074	0.3851 ± 0.0141	0.3819 ± 0.0046
$\alpha = 2.0$	0.2332 ± 0.0077	0.3851 ± 0.0090	0.3809 ± 0.0069
FACED	Cohen's Kappa	Weighted F1	Balanced Accuracy
$\alpha = 0.5$ **(default)**	**0.5406 ± 0.0084**	**0.5953** ± 0.0113	**0.5941** ± 0.0090
$\alpha = 0.1$	0.5295 ± 0.0080	0.5853 ± 0.0074	0.5839 ± 0.0069
$\alpha = 0.9$	0.4681 ± 0.0069	0.5270 ± 0.0065	0.5314 ± 0.0050
$\alpha = 2.0$	0.4782 ± 0.0037	0.5282 ± 0.0068	0.5360 ± 0.0035
ISRUC_S3	Cohen's Kappa	Weighted F1	Balanced Accuracy
$\alpha = 0.5$ **(default)**	**0.7671** ± 0.0091	**0.8202** ± 0.0071	**0.7856** ± 0.0031
$\alpha = 0.1$	0.7132 ± 0.0646	0.7834 ± 0.0523	0.7575 ± 0.0621
$\alpha = 0.9$	0.6073 ± 0.0230	0.6966 ± 0.0176	0.6564 ± 0.0058
$\alpha = 2.0$	0.5600 ± 0.0492	0.6572 ± 0.0430	0.6227 ± 0.0427

The decay parameter α controls how rapidly spatial kernel weights diminish with topological distance and, therefore, plays a central role in balancing locality preservation and kernel sparsification.

Across all datasets, $\alpha = 0.5$ consistently yields the highest performance. Larger values (e.g., $\alpha = 0.9$) moderately weaken spatial locality, while very small values (e.g., $\alpha = 0.1$) oversparsify the kernel and markedly reduce accuracy. Increasing α towards and beyond 1 (e.g., $\alpha = 2.0$) weakens the decay and assigns relatively larger weights to distant sub-kernels, which harms spatial locality and leads to clear performance drops, especially on ISRUC_S3. In addition, excessively large decay coefficients may amplify gradients during backpropagation and introduce training instability. For these reasons, we recommend α to ≤ 1 and adopt $\alpha = 0.5$ as a stable and well-performing default.

K.6 Ablation on Subband

To better understand the frequency dependencies encoded by CodeBrain, we conduct a systematic subband ablation study by masking each of the five canonical EEG frequency ranges: δ (0.5–4 Hz), θ (4–8 Hz), α (8–13 Hz), β (13–30 Hz), and γ (¿30 Hz). Unlike random dropout, this setting allows us to examine how the model allocates representational importance across physiologically meaningful frequency components. Experiments are performed on both the emotion recognition dataset (SEED-V) and the sleep-staging dataset (ISRUC_S3), covering two distinct neurophysiological tasks.

K.6.1 SEED-V (62 Channels): Subband Contributions in Emotion Recognition

Several key findings could be observed in this experiments:

Table 20: Subband ablation results on the SEED-V dataset.

Removed Band	Cohen's Kappa	Weighted F1	Balanced Accuracy
None	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023
δ	0.0297 ± 0.0027	0.2075 ± 0.0105	0.2201 ± 0.0004
θ	0.0142 ± 0.0050	0.1780 ± 0.0176	0.2098 ± 0.0039
α	0.0082 ± 0.0070	0.1306 ± 0.0536	0.2058 ± 0.0049
β	0.0442 ± 0.0120	0.2377 ± 0.0068	0.2360 ± 0.0095
γ	0.1372 ± 0.0081	0.3141 ± 0.0086	0.3092 ± 0.0044

- **Dominance of low-frequency structure.** Removing δ, θ, or α bands produces near-collapse of performance, indicating that emotional states are primarily encoded in slow and mid-range oscillations. This matches prior affective neuroscience findings showing that emotional arousal and valence strongly modulate rhythms below 13 Hz.
- **Residual robustness at higher frequencies.** Ablating β and especially γ reduces performance but does not catastrophically impair decoding. This suggests that CODEBRAIN leverages high-frequency activity as complementary contextual cues rather than primary discriminative features.
- **Contrast with raw-signal models.** Compared with prior end-to-end CNN/RNN models, the degradation patterns reveal that our decoupled tokenizer and cross-scale encoder preserve structured frequency dependencies rather than relying disproportionately on one frequency range.

These observations highlight that the learned frequency representation is aligned with known emotional knowledge while still maintaining robustness across a broad range of frequencies.

K.6.2 ISRUC_S3 (6 CHANNELS): SUBBAND CONTRIBUTIONS IN SLEEP STAGING

Table 21: Subband ablation results on the ISRUC_S3 dataset.

Removed Band	Cohen's Kappa	Weighted F1	Balanced Accuracy
None	**0.7671** ± 0.0091	**0.8202** ± 0.0071	**0.7856** ± 0.0031
δ	0.0225 ± 0.0391	0.1751 ± 0.0843	0.2043 ± 0.0237
θ	0.0728 ± 0.0557	0.1386 ± 0.0341	0.2812 ± 0.0749
α	0.1048 ± 0.0221	0.1762 ± 0.0186	0.3226 ± 0.0293
β	0.2131 ± 0.0660	0.2867 ± 0.1185	0.3835 ± 0.0446
γ	0.3618 ± 0.0012	0.4831 ± 0.0291	0.4764 ± 0.0082

Results on ISRUC_S3 sleep staging dataset reveal a qualitatively different frequency profile from SEED-V:

- **Critical dependence on δ and θ.** Removing slow-wave components almost eliminates information, consistent with their central role in NREM transitions and slow oscillations during deep sleep.
- **Higher-frequency bands remain more robust.** Masking β and γ still reduces performance but to a lesser extent, reflecting the fact that spindle- or arousal-related faster bursts are less dominant in the 6-channel ISRUC montage.
- **Physiology-specific feature reliance.** The frequency sensitivity patterns differ from SEED-V, demonstrating that CODEBRAIN adapts its feature allocation depending on task demands rather than relying on a fixed frequency prior.

The subband ablation results across two very different datasets show that CODEBRAIN could capture frequency structure, leveraging low-frequency dynamics for both emotion and sleep tasks while maintaining robustness to higher-band perturbations.

L Backbone Efficiency Comparison

To evaluate the computational efficiency of our proposed SGConv module, we conduct an ablation study by replacing it with three common sequence modeling modules: CNN, LSTM, and Transformer in EEGSSM block. We compare their model sizes, floating-point operations (FLOPs), and iteration times, as shown in Figure 20. Specifically, the CNN variant uses a 3-layer depthwise separable convolution block, while the LSTM and Transformer variants use a single layer of standard LSTM and Transformer Encoder (implemented by Pytorch), respectively. In terms of parameter count, SGConv contains 15.17M parameters, fewer than CNN (16.22M), LSTM (17.35M), and Transformer (21.2M). For FLOPs, SGConv also achieves the lowest computational cost at 8.74G, compared to Transformer's 27.79G. Regarding iteration time, SGConv is slightly slower than Transformer and CNN models in terms of training speed, but it outperforms the LSTM model. In summary, SGConv effectively reduces the number of parameters while maintaining computational complexity, which helps the model to be trained and inferred on smaller GPUs.

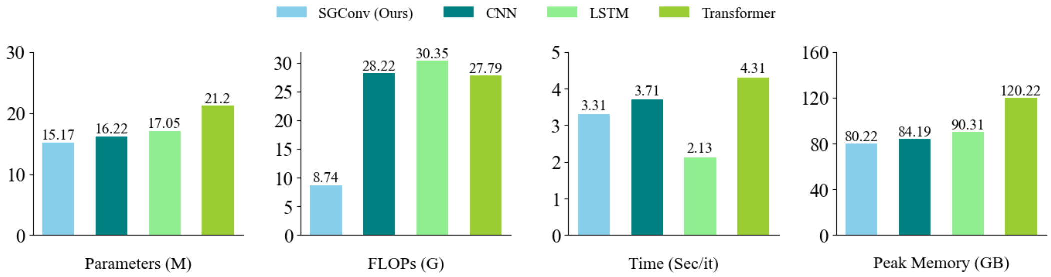


Figure 20: Computational Overhead of Using Different Backbones in the EEGSSM Module.

To further contextualize the computational efficiency of CodeBrain, we provide a comparison against widely used EEG foundation-model baselines. Table 22 summarizes model parameters, multiply–accumulate operations (MACs), and floating-point operations (FLOPs), offering stable and hardware-agnostic metrics across architectures. Overall, CodeBrain achieves a favorable balance between computational cost and representational capacity. Its FLOPs and MACs remain substantially lower than large-scale models such as BENDR and EEGPT, while maintaining higher parameter efficiency than CBraMod and LaBraM. Notably, CodeBrain occupies a middle ground in model size—significantly smaller than EEGPT while offering richer representational power than compact baselines like BIOT. This balanced compute–performance trade-off aligns with our design goal of building an efficient yet high-capacity EEG foundation model suitable for both research and deployment on modest hardware.

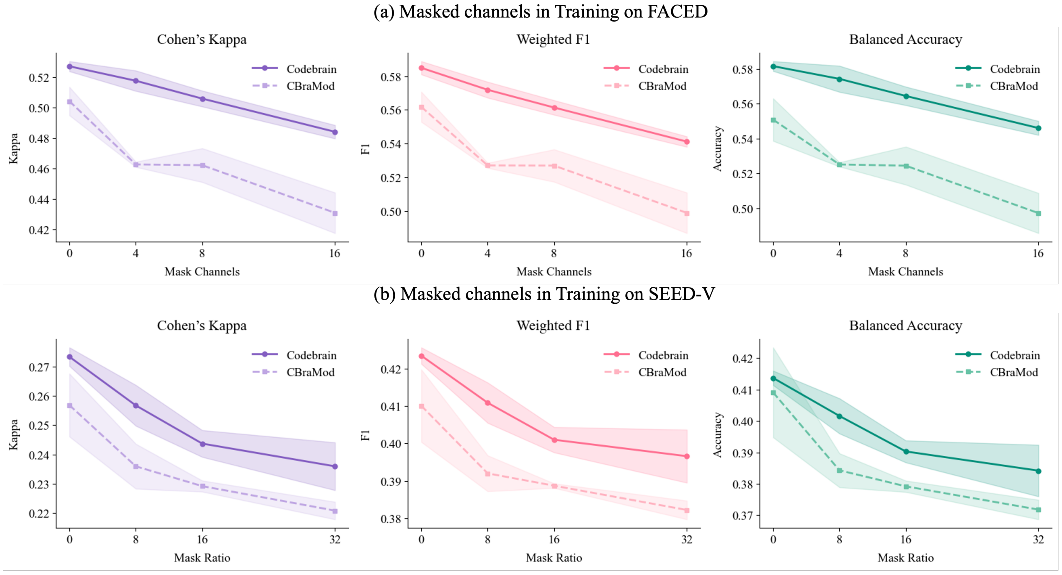
Table 22: Compute comparison between CodeBrain and representative EEG foundation-model baselines.

Model	MACs	Params	FLOPs
BENDR	12.51G	959.84M	25.02G
BIOT	0.255G	3.20M	0.510G
LaBraM	0.67G	6.02M	1.34G
CBraMod	0.64G	4.03M	1.29G
EEGPT	4.89G	25.24M	9.79G
CodeBrain	**4.37G**	**15.17M**	**8.74G**

M Model Robustness

M.1 Random Channel Dropout

In real-world scenarios, the collection of EEG often encounters situations where channels are missing, especially when using machines from different manufacturers. To test the model's performance on datasets with missing channel data, we randomly mask some channels in the training data for full parameter fine-tuning. We selected the FACED and SEED-V datasets for experiments because they represent short-sequence and long-sequence cases respectively, and their numbers of channels are relatively complete.



(a) Masked channels in Training on FACED

(b) Masked channels in Training on SEED-V

Figure 21: Performance after randomly masking different numbers of channels during the full parameter fine-tuning stage.

We evaluate the performance of the CodeBrain model and CBraMod model in scenarios with missing channels. We conducted three different experiments, randomly masking 12.5%, 25%, and 50% of the channels in each experiment, respectively. Figure 21 shows results of the experiment. It can be seen that our CodeBrain model outperforms the CBraMod model in all channel masking scenarios. On the FACED dataset, our model's performance after masking 25% of the channels is still close to that of CBraMod without masking. The performance decline of the CBraMod model is also faster than that of our model. This trend is even more pronounced on the SEED-V dataset. Through this experiment, we can demonstrate that the channel robustness of CodeBrain is stronger than that of the CBraMod model, retaining most of its performance even in cases of channel failure.

M.2 Brain-Region Channel Dropout

While random channel masking simulates incidental electrode failures, it does not capture the structured spatial organization of the EEG montage. To provide a neuroscientifically meaningful robustness evaluation, we conduct region-based masking on both the high-density SEED-V dataset (62 channels) and the minimal-montage ISRUC_S3 dataset (6 channels). These experiments simulate clinically relevant scenarios such as reference-electrode failures, lobe-specific dropout, and hemisphere-level signal loss. We also compare CodeBrain with the strongest baseline, CBraMod.

M.2.1 Region-Based Ablations on the High-Density SEED-V Dataset (62 Channels)

We design nine anatomically meaningful masking patterns, including (1) reference/hemisphere failures (midline-only, left-hemisphere masked, right-hemisphere masked), and (2) lobe-level dropout over occipital, frontal, temporal, central, frontocentral, and parietal regions. The results are summarized in Table 23.

Table 23: Region-masking results on the SEED-V dataset (62 channels).

Mask Setting	**Model**	**Cohen's Kappa**	**Weighted F1**	**Balanced Accuracy**
Baseline (no mask)	CBraMod	0.2569 ± 0.0143	0.4101 ± 0.0108	0.4091 ± 0.0097
	CodeBrain(Ours)	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023
Only midline	CBraMod	0.0114 ± 0.0098	**0.1740 ± 0.0121**	0.2077 ± 0.0018
	CodeBrain(Ours)	**0.0201** ± 0.0098	0.1610 ± 0.0087	**0.2169** ± 0.0067
Left hemisphere	CBraMod	0.0672 ± 0.0260	0.2505 ± 0.0227	0.2541 ± 0.0226
	CodeBrain(Ours)	**0.0956** ± 0.0030	**0.2779** ± 0.0051	**0.2750** ± 0.0030
Right hemisphere	CBraMod	0.0586 ± 0.0171	0.2500 ± 0.0214	0.2466 ± 0.0149
	CodeBrain(Ours)	**0.0678** ± 0.0195	**0.2540** ± 0.0066	**0.2569** ± 0.0166
Occipital	CBraMod	0.1316 ± 0.0217	0.3066 ± 0.0167	0.3068 ± 0.0174
	CodeBrain(Ours)	**0.2258** ± 0.0067	**0.3851** ± 0.0040	**0.3818** ± 0.0090
Frontal	CBraMod	0.0632 ± 0.0222	0.2486 ± 0.0177	0.2491 ± 0.0189
	CodeBrain(Ours)	**0.1255** ± 0.0090	**0.3025** ± 0.0080	**0.2976** ± 0.0046
Temporal	CBraMod	0.1571 ± 0.0284	0.3309 ± 0.0222	0.3265 ± 0.0227
	CodeBrain(Ours)	**0.2311** ± 0.0008	**0.3922** ± 0.0010	**0.3824** ± 0.0004
Central	CBraMod	0.1562 ± 0.0419	0.3305 ± 0.0336	0.3247 ± 0.0321
	CodeBrain(Ours)	**0.2410** ± 0.0078	**0.3997** ± 0.0067	**0.3894** ± 0.0055
Frontocentral	CBraMod	**0.1304** ± 0.0304	**0.3077 ± 0.0263**	**0.3039** ± 0.0244
	CodeBrain(Ours)	0.1254 ± 0.0090	0.3025 ± 0.0080	0.2976 ± 0.0047
Parietal	CBraMod	0.1571 ± 0.0284	0.3309 ± 0.0222	0.3265 ± 0.0227
	CodeBrain(Ours)	**0.2311** ± 0.0008	**0.3922** ± 0.0010	**0.3824** ± 0.0004

Across nearly all masking conditions, CodeBrain maintains stronger performance than CBraMod, particularly under lobe-level dropout (occipital, temporal, central), suggesting that its spatial-temporal modeling is less reliant on any single anatomical region. The severe degradation under midline-only signals further highlights the importance of distributed multi-lobe information in emotion-related EEG.

M.2.2 Region-Based Ablations on the Minimal-Montage ISRUC_S3 Dataset (6 Channels)

To test robustness under extreme spatial sparsity, we perform structured region masking on the 6-channel A1/A2-referenced ISRUC_S3 montage. We evaluate reference-electrode failures (masking A1 or A2) and lobe-specific removal (frontal, central, occipital). Results are shown in Table 24.

Even under this sparse spatial setup, CodeBrain consistently maintains higher performance than CBraMod across all mask types. This indicates that the learned representations are robust to structured regional dropout and remain stable even when half or more channels are removed.

Across both high-density and minimal-montage datasets, region-based ablation demonstrates that CodeBrain preserves strong predictive performance under structured channel dropout, highlighting its spatial robustness and reliable modeling of cross-regional EEG dependencies.

Table 24: Region-masking results on the ISRUC_S3 dataset (6 channels).

Mask Setting	**Model**	**Cohen's Kappa**	**Weighted F1**	**Balanced Accuracy**
Baseline (no mask)	CBraMod CodeBrain(Ours)	0.7407 ± 0.0251 **0.7671 ± 0.0091**	0.8056 ± 0.0219 **0.8202 ± 0.0071**	0.7844 ± 0.0126 **0.7856 ± 0.0031**
Right hemisphere	CBraMod CodeBrain(Ours)	0.6616 ± 0.0263 **0.7430** ± 0.0131	0.7392 ± 0.0292 **0.8047** ± 0.0086	0.6706 ± 0.0212 **0.7760** ± 0.0279
Left hemisphere	CBraMod CodeBrain(Ours)	0.6198 ± 0.1127 **0.7318** ± 0.0252	0.6964 ± 0.0958 **0.7989** ± 0.0196	0.6838 ± 0.0815 **0.7819** ± 0.0133
Occipital	CBraMod CodeBrain(Ours)	0.7089 ± 0.0412 **0.7447** ± 0.0226	0.7598 ± 0.0321 **0.8065** ± 0.0187	0.7340 ± 0.0269 **0.7824** ± 0.0080
Central	CBraMod CodeBrain(Ours)	0.6725 ± 0.0106 **0.7479** ± 0.0253	0.7526 ± 0.0074 **0.8111** ± 0.0197	0.6979 ± 0.0164 **0.7790** ± 0.0155
Frontal	CBraMod CodeBrain(Ours)	0.7134 ± 0.0245 **0.7295** ± 0.0531	0.7796 ± 0.0184 **0.7936** ± 0.0408	0.7250 ± 0.0327 **0.7825** ± 0.0282

M.3 Non-stationary Robustness

EEG signals are inherently non-stationary, with gradual fluctuations caused by electrode impedance changes, autonomic modulation, motion artifacts, and slow drift in sensor baselines. To examine how well the learned representations tolerate such structured temporal drift, we introduce a simple but effective perturbation: a linear baseline shift. This perturbation exaggerates slow-varying non-stationarity beyond what naturally appears in the data, providing a controlled stress test of robustness. We evaluate both SEED-V (emotion recognition) and ISRUC_S3 (sleep staging), comparing CodeBrain with the strongest baseline, CBraMod. Results are summarized in Table **??**.

Table 25: Non-stationary Robustness under linear baseline shift on SEED-V and ISRUC_S3.

		SEED-V		
Setting	**Model**	**Cohen's Kappa**	**Weighted F1**	**Balanced Accuracy**
Reference (no shift)	**CodeBrain** CBraMod	**0.2735** ± 0.0032 0.2569 ± 0.0143	**0.4235** ± 0.0022 0.4101 ± 0.0108	**0.4137** ± 0.0023 0.4091 ± 0.0097
Linear baseline shift	**CodeBrain** CBraMod	**0.2170** ± 0.0345 0.2027 ± 0.0147	**0.3799** ± 0.0276 0.3658 ± 0.0136	**0.3706** ± 0.0259 0.3628 ± 0.0111
		ISRUC_S3		
Setting	**Model**	**Cohen's Kappa**	**Weighted F1**	**Balanced Accuracy**
Reference (no shift)	**CodeBrain** CBraMod	**0.7671** ± 0.0091 0.7407 ± 0.0251	**0.8202** ± 0.0071 0.8056 ± 0.0219	**0.7856** ± 0.0031 0.7844 ± 0.0126
Linear baseline shift	**CodeBrain** CBraMod	**0.4762** ± 0.0835 0.4242 ± 0.0068	**0.5914** ± 0.0668 0.5478 ± 0.0087	**0.5617** ± 0.0630 0.5268 ± 0.0085

This controlled non-stationarity stress test reveals several consistent patterns. Moderate linear drift introduces clear performance degradation across both datasets, as expected for models trained under largely stationary conditions. The effect is especially pronounced on ISRUC_S3, where the limited channel count amplifies the impact of baseline shifts. Despite this, CodeBrain retains a larger proportion of its original performance than CBraMod, suggesting that its decoupled time–frequency tokenizer and multi-scale encoder yield more stable representations under slow global waveform drift. We also observe task-dependent differences in sensitivity: emotion recognition is disproportionately affected in low-frequency components, whereas sleep staging exhibits a more uniform degradation across metrics, reflecting the different spectral structures these tasks rely on. Overall, the results indicate that CodeBrain maintains stronger robustness to non-stationary perturbations, making it suitable for real-world settings where gradual baseline drift and electrode instability are unavoidable.

N Detailed Results on Scaling Laws

We provide the detailed scaling law results for both data and model size across three representative EEG datasets (FACED, SEED-V, and ISRUC_S3), covering three evaluation metrics. For brevity, only Cohen's kappa scores are included in the main text, while full results are provided in this section. Prior work (Wang et al., 2025; Jiang et al., 2024) has explored the effect of scaling EEG foundation models using 1 to 1000 hours of pretraining data. We extend this analysis in two key dimensions:

1. Scaling the pretraining data volume from 1k up to 9k hours.
2. Investigating model scaling by varying the depth of the EEGSSM encoder from 3 layers (3.86M parameters) to 24 layers (146.75M parameters) and the hidden size from 128 to 384.

N.1 Scaling Laws with Respect to Training Data Volume

We examine how the volume of pretraining data influences the downstream performance of CodeBrain. Specifically, we scale the pretraining duration from 1k to 9k hours and evaluate the resulting models on three downstream datasets: FACED, SEED-V, and ISRUC_S3. Detailed quantitative results across three evaluation metrics (Cohen's kappa, weighted F1 score, and balanced accuracy) are presented in Tables 26, 27, and 28, respectively. As shown in Figure 22, increasing the amount of pretraining data generally leads to consistent improvements across all datasets and metrics. On FACED and ISRUC_S3, performance gains are steady throughout the entire range up to 9k hours, while on SEED-V, the trend is more modest and plateaus after 5k hours. These results highlight the importance of large-scale data for representation learning in EEG and suggest that further scaling may continue to yield performance benefits.

Table 26: Training Data Scaling Laws of CodeBrain on FACED Dataset.

Training Data	Cohen's Kappa	Weighted F1	Balanced Accuracy
1000 Hours	0.5014 ± 0.0107	0.5462 ± 0.0146	0.5452 ± 0.0163
2000 Hours	0.5133 ± 0.0068	0.5540 ± 0.0120	0.5521 ± 0.0133
3000 Hours	0.5189 ± 0.0086	0.5688 ± 0.0104	0.5687 ± 0.0083
4000 Hours	0.5208 ± 0.0032	0.5741 ± 0.0041	0.5713 ± 0.0024
5000 Hours	0.5171 ± 0.0040	0.5692 ± 0.0131	0.5661 ± 0.0105
6000 Hours	0.5273 ± 0.0085	0.5803 ± 0.0064	0.5764 ± 0.0069
7000 Hours	0.5328 ± 0.0116	0.5854 ± 0.0082	0.5809 ± 0.0097
8000 Hours	0.5336 ± 0.0082	0.5875 ± 0.0071	0.5835 ± 0.0072
9000 Hours	**0.5406 ± 0.0084**	**0.5953 ± 0.0113**	**0.5941 ± 0.0098**

In addition, we visualize the pretraining optimization behavior across different data scales in Figure 23. As expected, larger pretraining data consistently lead to lower training loss, indicating more effective representation learning. Notably, the convergence curves become progressively smoother and more stable as training data volume increases, suggesting improved optimization stability in large-scale regimes. While smaller training data volumes (e.g., 1k–3k hours) show relatively high starting loss and slower convergence, larger training data volumes (6k–9k hours) reach lower final losses and exhibit diminishing returns, aligning with trends observed in downstream performance. These findings provide further empirical support for the scalability of EEG foundation models and reinforce the role of large data in enhancing both optimization and generalization.

N.2 Scaling Laws with Respect to Model Size

We further investigate how model parameters affect downstream performance by scaling the number of layers in the EEGSSM encoder from 3 to 8, resulting in parameter counts ranging from 6.82M to 15.17M. Detailed results across the FACED, SEED-V, and ISRUC_S3 datasets are provided in

Table 27: Training Data Scaling Laws of CodeBrain on SEED-V Dataset.

Training Data	Cohen's Kappa	Weighted F1	Balanced Accuracy
1000 Hours	0.2584 ± 0.0044	0.3946 ± 0.0063	0.3799 ± 0.0096
2000 Hours	0.2648 ± 0.0062	0.4042 ± 0.0091	0.3961 ± 0.0084
3000 Hours	0.2689 ± 0.0055	0.4117 ± 0.0075	0.4028 ± 0.0082
4000 Hours	0.2672 ± 0.0043	0.4121 ± 0.0054	0.4022 ± 0.0049
5000 Hours	0.2678 ± 0.0069	0.4120 ± 0.0084	0.4026 ± 0.0067
6000 Hours	0.2669 ± 0.0031	0.4113 ± 0.0067	0.4030 ± 0.0102
7000 Hours	0.2686 ± 0.0043	0.4129 ± 0.0052	0.4027 ± 0.0048
8000 Hours	0.2703 ± 0.0049	0.4165 ± 0.0034	0.4094 ± 0.0054
9000 Hours	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023

Table 28: Training Data Scaling Laws of CodeBrain on the ISRUC_S3 Dataset.

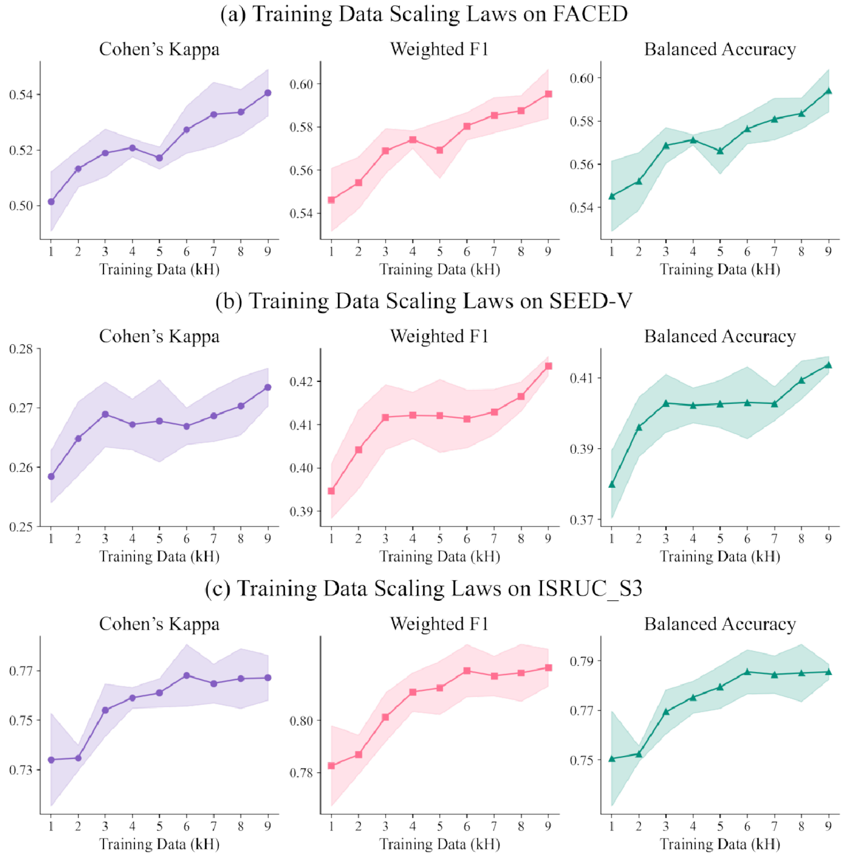
Training Data	Cohen's Kappa	Weighted F1	Balanced Accuracy
1000 Hours	0.7340 ± 0.0187	0.7826 ± 0.0154	0.7505 ± 0.0192
2000 Hours	0.7347 ± 0.0051	0.7869 ± 0.0075	0.7524 ± 0.0032
3000 Hours	0.7540 ± 0.0106	0.8012 ± 0.0093	0.7694 ± 0.0089
4000 Hours	0.7590 ± 0.0042	0.8108 ± 0.0074	0.7753 ± 0.0065
5000 Hours	0.7610 ± 0.0058	0.8124 ± 0.0102	0.7794 ± 0.0086
6000 Hours	0.7681 ± 0.0125	0.8190 ± 0.0100	0.7856 ± 0.0089
7000 Hours	0.7648 ± 0.0079	0.8170 ± 0.0076	0.7845 ± 0.0076
8000 Hours	0.7668 ± 0.0121	0.8182 ± 0.0109	0.7851 ± 0.0116
9000 Hours	**0.7671** ± 0.0091	**0.8202** ± 0.0071	**0.7856** ± 0.0031

Tables 29, 30, and 31, respectively. To visualize the trend more clearly, Figure 24 presents the performance curves as model size increases. Across all three datasets and evaluation metrics, we observe a consistent performance gain as the model size increases. The improvements are particularly pronounced on the FACED and ISRUC_S3 datasets, where all three metrics show steady growth up to the largest model. In contrast, performance on SEED-V improves more modestly and begins to plateau beyond 13.5M parameters. These results suggest that increasing model capacity can enhance generalization ability, especially for datasets with richer structure or more complex temporal dynamics, while also indicating that optimal scaling may be task-dependent.

Table 29: Model Size Scaling Laws of CodeBrain on the FACED Dataset.

Layer	**Hidden size**	**Params**	**Cohen's Kappa**	**Weighted F1**	**Balanced Accuracy**
3	128	3.86M	0.4786±0.0131	0.5231±0.0164	0.5287±0.0207
3	200	6.82M	0.4818±0.0165	0.5362±0.0113	0.5317±0.0182
4	200	8.49M	0.4988±0.0082	0.5506±0.0091	0.5497±0.0084
5	200	10.16M	0.5096±0.0049	0.5686±0.0104	0.5642±0.0067
6	200	11.83M	0.5244±0.0113	0.5705±0.0085	0.5778±0.0095
7	200	13.50M	0.5314±0.0069	0.5872±0.0080	0.5846±0.0067
8	200	15.17M	0.5406±0.0084	0.5953±0.0113	0.5941±0.0098
12	256	33.15M	0.5478±0.0013	0.5912±0.0031	0.5901±0.0128
24	384	146.75M	**0.5503**±0.0120	**0.5964**±0.0178	**0.5985**±0.0233

To better understand the optimization behavior during pretraining, we plot the training loss curves for different model sizes in Figure 25. As expected, larger models consistently achieve lower final



(a) Training Data Scaling Laws on FACED

(b) Training Data Scaling Laws on SEED-V

(c) Training Data Scaling Laws on ISRUC_S3

Figure 22: Training Data Scaling Laws on FACED, SEED-V, and ISRUC_S3.

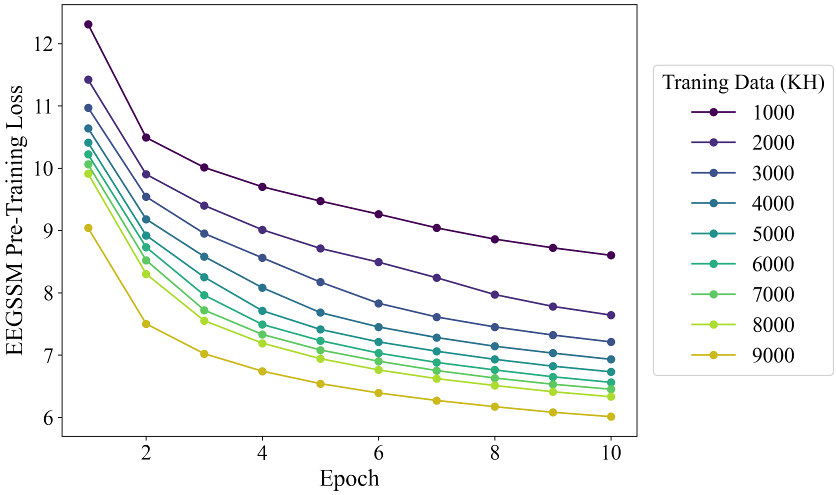


Figure 23: EEGSSM Pre-Training Loss Curve for Different Training Data Volume.

training loss, indicating stronger capacity to fit the pretraining objective. The loss reduction is particularly evident when increasing from 3 to 6 layers, while the gain starts to saturate beyond 7 layers. Even when increased to 24 layers, the reduction in pre-training loss brought by more than 100M parameters is not significant. This trend mirrors the downstream performance in Figure 24 and Tables 29–31, suggesting that both optimization efficiency and generalization benefit from increased

Table 30: Model Size Scaling Laws of CodeBrain on the SEED-V Dataset.

Layer	Hidden size	Params	Cohen's Kappa	Weighted F1	Balanced Accuracy
3	128	3.86M	0.2576±0.0047	0.3969±0.0042	0.3896±0.0027
3	200	6.82M	0.2609±0.0078	0.4004±0.0112	0.3956±0.0098
4	200	8.49M	0.2638±0.0080	0.4108±0.0091	0.4030±0.0121
5	200	10.16M	0.2645±0.0102	0.4127±0.0158	0.4013±0.0084
6	200	11.83M	0.2663±0.0056	0.4202±0.0068	0.4079±0.0055
7	200	13.50M	0.2724±0.0057	0.4211±0.0060	0.4120±0.0086
8	200	15.17M	0.2735±0.0032	0.4235±0.0022	0.4137±0.0023
12	256	33.15M	0.2807±0.0029	0.4317±0.0036	0.4182±0.0028
24	384	146.75M	**0.2831**±0.0033	**0.4342**±0.0030	**0.4216**±0.0031

Table 31: Model Size Scaling Laws of CodeBrain on the ISRUC_S3 Dataset.

Layer	Hidden size	Params(M)	Cohen's Kappa	Weighted F1	Balanced Accuracy
3	128	3.86M	0.7434±0.0087	0.7813±0.0107	0.7493±0.0058
3	200	6.82M	0.7456±0.0083	0.7862±0.0104	0.7514±0.0084
4	200	8.49M	0.7486±0.0014	0.7942±0.0040	0.7604±0.0035
5	200	10.16M	0.7516±0.0025	0.7985±0.0056	0.7639±0.0079
6	200	11.83M	0.7570±0.0082	0.8064±0.0068	0.7734±0.0061
7	200	13.50M	0.7620±0.0045	0.8153±0.0091	0.7824±0.0100
8	200	15.17M	0.7671±0.0091	0.8202±0.0071	0.7856±0.0031
12	256	33.15M	0.7753±0.0113	0.8316±0.0074	0.7940±0.0061
24	384	146.75M	**0.7791**±0.0108	**0.8352**±0.0072	**0.8008**±0.0040

model size—though with diminishing returns as parameter count grows. These results reinforce the scalability of EEGSSM and underscore the importance of balancing capacity with task-specific requirements.

Our results demonstrate consistent improvements in downstream performance as both data volume and model capacity increase, suggesting that EEG foundation models may continue to benefit from further scaling, similar to trends observed in vision and language domains.

N.3 Computational Analysis Across Scales.

Complementing the scaling analyses on model size and data size presented in the previous subsections, we further examine how computational cost scales with architectural capacity. This provides a detailed breakdown of parameters, FLOPs, throughput, and GPU memory usage across all CodeBrain model configurations, allowing a holistic view of the efficiency capacity.

Table 32 shows a smooth increase in parameters, FLOPs, and memory consumption as model depth and hidden size grow. Notably, throughput decreases sub-linearly with scale, demonstrating that the architecture maintains high computational efficiency even at larger capacities.

Lightweight variants (3-6 layers) offer high throughput (3.5-5 samples/s) and low memory usage ($<$ 8 GB), suitable for real-time or resource-constrained EEG applications. Mid-sized models (8-12 layers) provide an excellent balance between performance and efficiency, which aligns with where the accuracy scaling curve begins to saturate. The largest configuration (24 layers, 146M parameters) substantially increases capacity and FLOPs, corresponding to the upper regime of diminishing returns observed in the model-size scaling law. We therefore select the 8-layer configuration as the main model used in our experiments after balancing performance and efficiency.

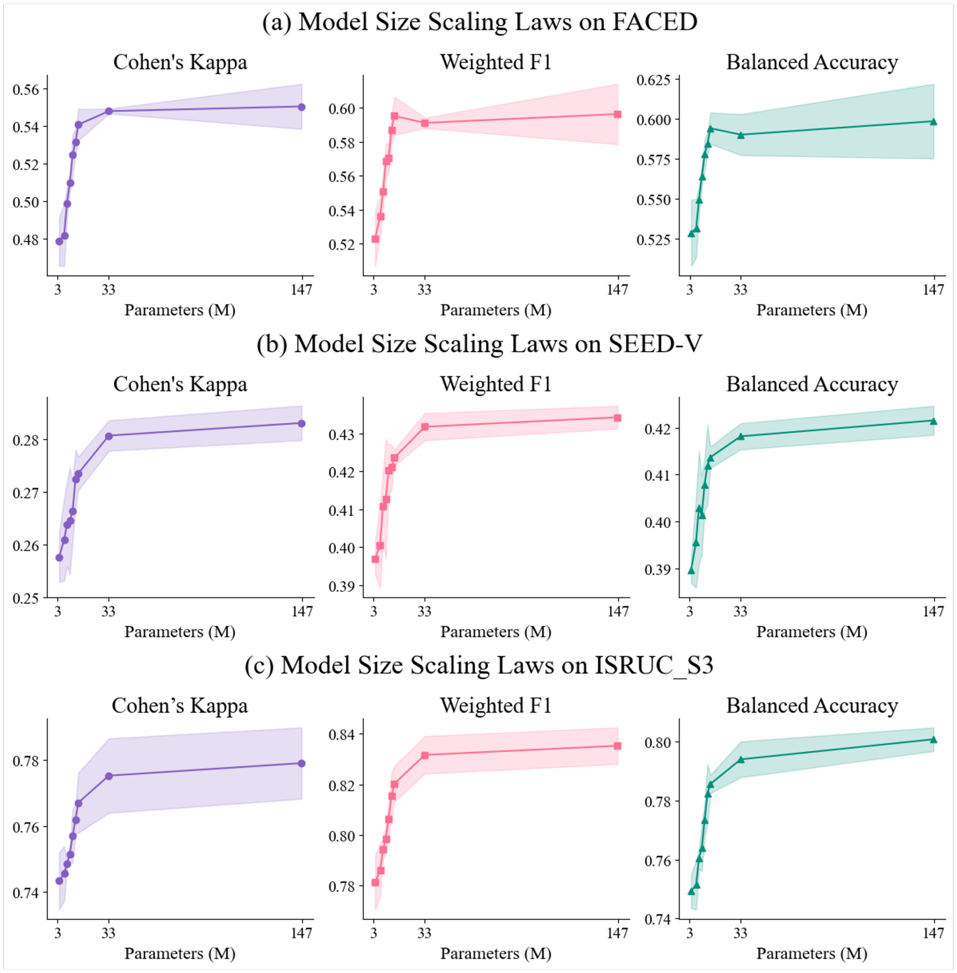


Figure 24: Model Size Scaling Laws on FACED, SEED-V, and ISRUC_S3.

Table 32: Computational analysis across different model scales..

Layer	Hidden Size	Params	FLOPs	Throughput	GPU Memory
3	128	3.96M	1.7G	4.90	4.87
3	200	6.82M	3.35G	4.62	5.63
4	200	8.49M	4.43G	3.77	6.48
5	200	10.16M	5.51G	3.67	7.33
6	200	11.83M	6.58G	3.52	8.10
7	200	13.50M	7.66G	2.94	8.95
8	200	15.17M	8.74G	2.78	9.79
12	256	34.38M	19.04G	1.84	15.41
24	384	146.75M	72.99G	1.47	38.43

Overall, these results show that CodeBrain follows predictable computational scaling behavior and offers flexible operating points for various deployment budgets.

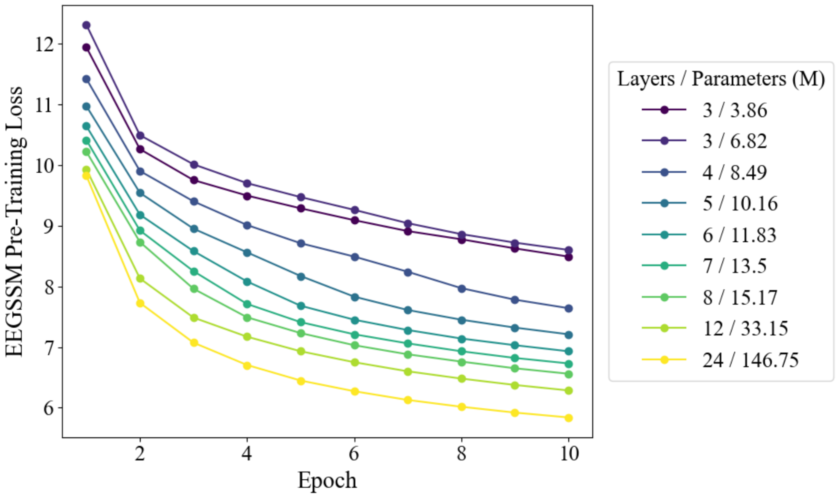


Figure 25: EEGSSM Pre-Training Loss Curve for Different Model Size.

O Low-Resource Comparison With Existing Methods

Table 33: Comparison under data-limited settings on the FACED dataset (9-class).

Methods	Cohen's Kappa	Weighted F1	Balanced Accuracy
Linear Probing			
LaBraM (lp)	0.3026 ± 0.0121	0.3789 ± 0.0154	0.3812 ± 0.0148
CBraMod (lp)	0.3378 ± 0.0139	0.4123 ± 0.0117	0.4146 ± 0.0123
CodeBrain (lp)	**0.3587** ± 0.0136	**0.4311** ± 0.0109	**0.4327** ± 0.0127
10% Few-Shot			
LaBraM (10%)	0.1358 ± 0.0163	0.2247 ± 0.0196	0.2265 ± 0.0174
CBraMod (10%)	0.1632 ± 0.0156	0.2595 ± 0.0138	0.2604 ± 0.0148
CodeBrain (10%)	**0.1716** ± 0.0101	**0.2599** ± 0.0104	**0.2654** ± 0.0093
30% Few-Shot			
BIOT (30%)	0.2573 ± 0.0346	0.3501 ± 0.0341	0.3428 ± 0.0329
LaBraM (30%)	0.2672 ± 0.0371	0.3548 ± 0.0325	0.3513 ± 0.0315
CBraMod (30%)	0.3239 ± 0.0265	0.4056 ± 0.0256	0.4035 ± 0.0233
CodeBrain (30%)	**0.3356** ± 0.0253	**0.4114** ± 0.0225	**0.4104** ± 0.0281
BIOT (full)	0.4476 ± 0.0254	0.5136 ± 0.0112	0.5118 ± 0.0118
LaBraM (full)	0.4698 ± 0.0102	0.5288 ± 0.0188	0.5273 ± 0.0107
CBraMod (full)	0.5041 ± 0.0122	0.5618 ± 0.0093	0.5509 ± 0.0089
CodeBrain (full)	**0.5406** ± 0.0084	**0.5953** ± 0.0113	**0.5941** ± 0.0098

Table 34: Comparison under data-limited settings on the SEED-V dataset (5-class).

Methods	Cohen's Kappa	Weighted F1	Balanced Accuracy
Linear Probing			
LaBraM (lp)	0.1941 ± 0.0184	0.3457 ± 0.0135	0.3413 ± 0.0144
CBraMod (lp)	0.2239 ± 0.0053	0.3823 ± 0.0041	0.3791 ± 0.0050
CodeBrain (lp)	**0.2302** ± 0.0166	**0.3889** ± 0.0154	**0.3829** ± 0.0136
10% Few-Shot			
LaBraM (10%)	0.0302 ± 0.0065	0.2194 ± 0.0079	0.2228 ± 0.0091
CBraMod (10%)	0.0174 ± 0.0029	0.2071 ± 0.0125	0.2127 ± 0.0023
CodeBrain (10%)	**0.1690** ± 0.0170	**0.3410** ± 0.0133	**0.3331** ± 0.0138
30% Few-Shot			
BIOT (30%)	0.1775 ± 0.0425	0.3492 ± 0.0416	0.3505 ± 0.0375
LaBraM (30%)	0.2044 ± 0.0384	0.3700 ± 0.0321	0.3686 ± 0.0305
CBraMod (30%)	0.2291 ± 0.0246	0.3886 ± 0.0255	0.3877 ± 0.0236
CodeBrain (30%)	**0.2376** ± 0.0284	**0.3943** ± 0.0259	**0.3902** ± 0.0271
BIOT (full)	0.2261 ± 0.0262	0.3856 ± 0.0203	0.3837 ± 0.0187
LaBraM (full)	0.2386 ± 0.0209	0.3974 ± 0.0111	0.3976 ± 0.0138
CBraMod (full)	0.2569 ± 0.0143	0.4101 ± 0.0108	0.4091 ± 0.0097
CodeBrain (full)	**0.2735** ± 0.0032	**0.4235** ± 0.0022	**0.4137** ± 0.0023

To evaluate model performance under constrained conditions, we examine three transfer settings: 30% few-shot, 10% few-shot, and linear probing. The 30% and 10% settings reflect data-limited conditions, where only a small portion of labeled target data is available for adaptation. In contrast, linear probing represents a compute-limited condition where the backbone is frozen, and only a single linear layer is trained. Results for both the FACED (9-class) and SEED-V (5-class) datasets are reported in Tables 33 and 34.

Across all three settings and both datasets, CodeBrain consistently achieves the best performance among the evaluated methods, demonstrating stronger data efficiency and reliable transfer behavior under limited supervision. We also observe that few-shot performance is generally lower than linear probing, which is expected because full fine-tuning with very limited labeled data tends to be more sensitive to overfitting and distribution shift, whereas linear probing offers a more stable evaluation by freezing the backbone and relying only on the pretrained representations. This also confirms that the pretrained representations learned by CodeBrain already contain meaningful structure. Full-data results are also included for reference.

While CodeBrain remains superior to prior EFMs in all limited-data and compute-limited conditions, all models exhibit notable performance drops compared to full fine-tuning, which is expected. EEG signals inherently exhibit extremely low signal-to-noise ratios and substantial domain shift across subjects, devices, montages, and recording setups. Under such conditions, a small number of labeled samples is often insufficient to fully adapt pretrained representations to the target distribution, making low-resource EEG transfer a challenging but important research problem. Systematic evaluations in these settings (both few-shot and linear probing) therefore remain underexplored in the literature, and our study provides an initial step toward addressing this gap.

P Limitations and Future Work

Our work presents promising results but also highlights several limitations that offer directions for future exploration.

First, the interpretability analyses in this paper focus on *representation-level* structure learned during pretraining. As this paper centers on developing a foundation model, our focus is mainly placed on understanding the representation space learned during pretraining. Future work may incorporate *decision-level* interpretability during finetuning, for example by exploring sparse codebook selection, task-guided gating mechanisms, or disentangling how temporal and frequency codes contribute to class-specific predictions.

Second, our model and experiments focus on only *scalp EEG* data. Extending the proposed framework to modalities with richer frequency content, such as intracranial EEG (iEEG), electrocorticography (ECoG), or stereo-EEG (sEEG), offers an opportunity to study how the model behaves under substantially different signal characteristics.

These directions offer promising directions toward building more general and more interpretable brain foundation models that operate across signal modalities and provide both meaningful insights.

Q Code Access

The code has been made publicly available via an anonymous GitHub repository: https://anonymous.4open.science/r/CodeBrain-D7E6

R Use of LLM

In preparing this manuscript, we made use of large language models (LLMs) such as ChatGPT, Gemini, and Deepseek for auxiliary writing support. Specifically, LLMs were employed in three ways:

(i) Polishing the English writing style, including grammar correction and improving fluency;

(ii) Assisting with LaTeX formatting and typesetting to ensure consistent presentation of mathematical equations, tables, and figures.

All core technical content, theoretical results, and experimental findings were designed, implemented, and validated by the authors. LLM usage was restricted to language refinement and formatting assistance, without influencing the originality or validity of the scientific contributions.