Population Transformer: Learning Population-level Representations of Intracranial Activity

Geeling Chau[1*] **Christopher Wang**[2*] **Sabera Talukder**[1] **Vighnesh Subramaniam**[2]

Saraswati Soedarmadji[1] **Yisong Yue**[1] **Boris Katz**[2] **Andrei Barbu**[2]

[1]California Institute of Technology
{gchau, sabera, ssoedarm, yyue}@caltech.edu
[2]MIT CSAIL, CBMM
{czw, vsub851, boris, abarbu}@mit.edu

Abstract

We present a self-supervised framework that learns population-level codes for arbitrary ensembles of neural recordings. We address key challenges in scaling models with neural time-series data, namely, sparse and variable electrode distribution across subjects and datasets. The Population Transformer (PopT) stacks on top of pretrained representations and enhances downstream decoding by enabling learned aggregation of multiple spatially-sparse data channels. The pretrained PopT lowers the amount of data required for downstream decoding experiments, while increasing accuracy, even on held-out subjects and tasks. Beyond decoding, we interpret the pretrained PopT and fine-tuned models to show how they can be used to extract neuroscience insights from massive amounts of data. We release our code as well as a pretrained PopT to enable off-the-shelf improvements in multi-channel intracranial data decoding and interpretability.

1 Introduction

Building effective representations of neural data is an important tool in enabling neuroscience research. Recordings from the brain such as intracranial (iEEG) and scalp (EEG) electroencephology, consist of time series recorded simultaneously from multiple channels. The relationships between these time series are complex, and governed by the underlying functional connectivity that exists between brain regions. Our goal is to build an effective model of multi-channel activity. Recently, improvements have been made in modeling for the single channel setting [1, 2]. This suggests an approach for learning multi-channel representations via aggregating single channel embeddings. However, this is not a trivial task. For brain recordings, particularly iEEG, one must contend with sparse and variable electrode layouts, which change the semantics of input channels from subject to subject. This forces many Brain Machine Interface (BMI) approaches to rely on expensive schemes, in which models are retrained for each new participant, requiring large amounts of data for calibration [3–7]. To this end, we propose a self-supervised learning framework, Population Transformer (PopT), which is specifically designed to aggregate single-channel encodings across variable electrode layouts.

PopT is a self-supervised pretraining approach on a transformer backbone that learns subject-generic representations of arbitrary electrode ensembles. Transformers offer the flexibility to learn aggregate information across channel configurations, but large amounts of data is needed to train the attention weights [8]. During pretraining, we train on large amounts of unannotated data and simultaneously

[*] Equal contribution

1st Workshop on NeuroAI @ 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

optimize both a channel-level and ensemble-level objective. This requires the model to (1) build subject-generic representations of channel ensembles and (2) meaningfully distinguish temporal relationships between different ensembles of channels.

Our PopT approach is modular, and builds on top of powerful single-channel temporal embeddings, which provides two key advantages. First, by separating the single-channel embedding and multi-channel-aggregation into different modules, we make our approach agnostic to the specific type of temporal embedding used, leaving room for future independent improvements along either the temporal or spatial dimension (an approach that has been validated in video modeling [9]). Second, by taking advantage of learned channel embeddings, PopT training is computationally lightweight compared to end-to-end counterparts (Appendix E) and baseline aggregation approaches, allowing for adoption in lower compute resource environments.

Empirically, we find that our pretrained PopT outperforms existing aggregation approaches, highlighting the usefulness of learning spatial relationships during pretraining. Moreover, we find that these benefits hold even for subjects not seen during pretraining, lending to its usefulenss for new subject decoding. We also find that the pretrained PopT weights themselves reveal interpretable patterns for neuroscientific study.

Our main contributions are:

1. a lightweight, generic SSL framework, Population Transformer (PopT) that learns arbitrary joint representations of channel embeddings across unannotated datasets of neural activity.
2. a demonstration that a pretrained PopT benefits downstream performance, interpretability, and training efficiency in comparison to baseline aggregation approaches.
3. a trained and usable off-the-shelf model that computes population-level representations of high temporal resolution intracranial neural recordings.

2 Related Work

Self-supervised learning on neural data Channel independent pretrained models are a popular approach for neural spiking data [10], intracranial brain data [1, 11], and general time-series [2]. Additionally, in fixed-channel neural datasets, approaches exist for EEG [12–14], fMRI [15–17], and calcium imaging [18] datasets. However, these approaches do not learn population-level interactions across datasets with different recording layouts, either due to a single-channel focus or the assumption that the channel layout is fixed. Several works pretrain spatial and temporal dimensions across datasets with variable inputs [19–23], but most simultaneously learn the temporal embeddings with the spatial modeling, which make them challenging to interpret and computationally expensive to train, especially for high temporal resolution signals. To our knowledge, we are the first to study the problem of building pretrained channel aggregation models on top of pre-existing temporal embeddings trained across neural datasets with variable channel layouts, allowing for modeling of high quality neural data.

Modeling across variable input channels Modeling spatial representations on top of temporal embeddings has been found to be beneficial for decoding [3, 24, 25], but prior works use supervised labels, so do not leverage large amounts of unannotated data. The brain-computer-interface field has studied how to align latent spaces [26–30] which either still requires creating an alignment matrix to learn across datasets or only provides post-training alignment mechanisms rather than learning across datasets. Other approaches impute missing channels or learn latent spaces robust to missing channels [31–33], but these are more suited for the occasional missing channel rather than largely varying sensor layouts. We directly learn spatial-level representations using self-supervised learning across datasets to leverage massive amounts of unannotated intracranial data.

3 Population Transformer Approach

Figure 1 overviews our Population Transformer (PopT) approach. The key ideas are: (1) to learn a generic representation of neural recordings that can handle arbitrary electrode configurations; and (2) to employ a modular system design that uses a transformer architecture to aggregate information from existing per-channel temporal embeddings. To do so, we employ a self-supervised pretraining approach to learn ensemble and channel level representations. Afterwards, one can fine-tune PopT on

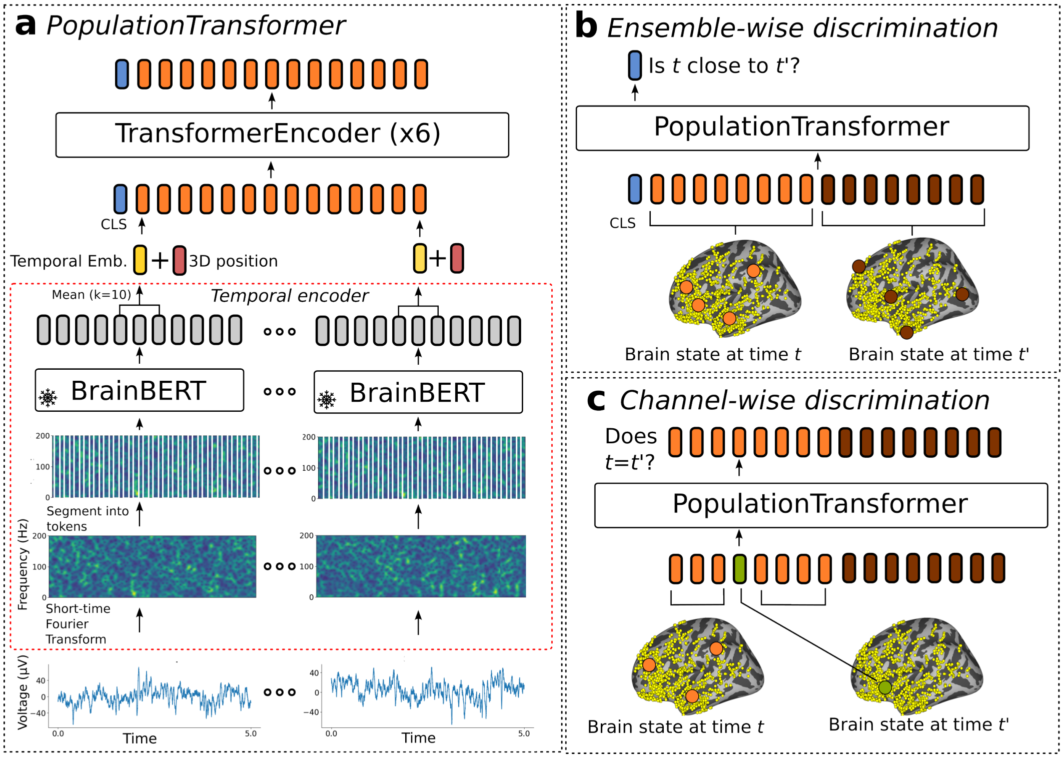

Figure 1: **Schematic of our approach**. The inputs to our model (a) are the neural activities from a collection of electrodes in a given time interval (bottom). These are passed to a frozen temporal embedding model (dotted red outline: BrainBERT [1] shown), which produces a set of time embedding vectors (yellow). The 3D positions of each electrode (red) are summed with these vectors to produce the model inputs (orange, lower). PopT produces space-contextual embeddings (orange, top) for each electrode and a `[CLS]` token (blue, top), which can be fine-tuned for downstream tasks. In pretraining, PopT learns two objectives simultaneously. In the first, (b) PopT determines whether two different sets of electrodes (orange vs brown) represent consecutive or non-consecutive times. In the second objective, (c) PopT must determine whether an input channel has been replaced with activity at a random other time that is inconsistent with the majority of inputs.

downstream decoding tasks. In addition to offering strong decoding results, including generalization to new subjects with different electrode configurations than training subjects (see Section 5), the modular system design is computationally lightweight (see Appendix E), can benefit from improved temporal representations, and is more readily interpretable (see Figure 2b).

Architecture A schematic of our Population Transformer (PopT) approach is shown in Figure 1. We adopt a transformer backbone due to its ability to accommodate variable channel configurations. Consider a given subject with N channels indexed by $C = \{1, ..., N_c\}$, and an arbitrary subset of channels $S \subseteq C$. Let $x_i^t \in \mathbb{R}^T$ denote a time window of activity from channel i that begins at time t, where T is the number of time samples in the interval. The PopT takes as input a collection of such channels activities, $X^t = \{x_i^t | i \in S\}$, as well as a special `[CLS]` token. Per channel, each interval of brain activity is passed through a temporal embedding model B, in the figure's case BrainBERT [1], to obtain a representation of each channel's temporal context, $B(x_i^t) \in \mathbb{R}^d$, where d is the embedding dimension. For BrainBERT, the first step of pre-processing involves obtaining the STFT for the signal, but preprocessing will differ depending on the embedding model used.

To allow the model to learn a common brain state representation across layouts, each channel's embedding is summed with its 3D position, so that the final processed input to the PopT is $X_B^t = \{B(x_i^t) + pos(i) + \mathcal{N}(0, \sigma) | x_i^t \in X^t\}$. The PopT receives this as an $S \times d$ matrix. Spatial location is given by the electrode's Left, Posterior, and Inferior coordinates for iEEG electrodes [34], and XYZ positions for EEG electrodes. We add Gaussian fuzzing to each coordinate location to prevent overfitting to a particular set of coordinates. Membership in a particular ensemble (see below:

ensemble-wise loss) is also encoded. The four encodings are concatenated together to form the position embedding $pos(i) = [e_{\text{left}}; e_{\text{post.}}; e_{\text{inf}}; e_{\text{ensemble}}]$, where e is given using a sinusoidal position encoding that represents a scalar coordinate as a unique combination of sines [35].

The core of PopT consists of a transformer encoder stack (see Appendix A: Architectures). The output of the PopT are spatial-contextual embeddings of the channels $Y = \{y_i\}$ and an embedding of the CLS token y_{cls}. During pretraining, the PopT additionally is equipped with a linear head for the [CLS] token output and separate linear heads for all other individual token outputs. These produce the scalars $\tilde{y}_{cls}$ and $\tilde{y}_i$ respectively, which are used in the pretraining objective (Figure 1b and c).

Self-supervised loss Our loss function has two discriminative components: (1) *ensemble-wise* — the model determines if activities from two channel ensembles occurred consecutively, requiring an effective brain state representation at the ensemble-level, (2) *channel-wise* — the model identifies outlier channels that have been swapped with a different timepoint's activity, requiring sensitivity to surrounding channel context.

A key aspect of our method is the fact that our objective is discriminative, rather than reconstructive, as is often the case in self-supervision [36, 1]. In practice, the temporal embeddings often have low effective dimension (see Wang et al. [1]), and reconstruction rewards the model for overfitting to "filler" dimensions in the feature vector (Section 5).

Pretraining In *ensemble-wise discrimination* (fig. 1b), two different subsets of channels $S_A, S_B \subset C$ are chosen with the condition that they be disjoint $S_A \cap S_B = \emptyset$. During pretraining, the model receives the activities from these channels at separate times $X_A^t = \{x_i^t \mid i \in S_A\}$ and $X_B^{t'} = \{x_i^{t'} \mid i \in S_B\}$. The objective of the task is then to determine whether these states X_A^t and $X_B^{t'}$ have occurred consecutively in time ($|t - t'| = 500ms$) or are separated by some further, randomly selected interval. Given the output of the classification head, the loss function $\mathcal{L}_N$ is the binary cross-entropy. We also vary the number of input channels during sampling to ensure the model handles ensembles of different sizes. Additionally, we select disjoint subsets for ensemble-wise discrimination to prevent the model from solving tasks through trivial copying.

In *channel-wise discrimination* (fig. 1c), the model must determine whether a channel's activity has been swapped with activity from a random time. Precisely, activity from each channel i is drawn from a time t_i. All channels are drawn from the same time $t_i = T$, and then 10% of the channels are randomly selected to have their activity replaced with activity from the same channel, but taken from a random point in time $t_i \neq T$. Then, given the token outputs of PopT, the channel-wise loss function $\mathcal{L}_C$ is the binary cross-entropy. Then, our complete objective function is $\mathcal{L} = \mathcal{L}_N + \mathcal{L}_C$.

Fine-tuning During fine-tuning, the [CLS] intermediate representation, $\tilde{y}_{cls}$ of the pretrained PopT is passed through a single layer linear neural network to produce a scalar $\hat{y}_{cls}$. This scalar is the input to binary cross entropy loss for our decoding tasks (see Section 4). After fine-tuning, we perform interpretability analysis on [CLS] attention weights with techniques outlined in Appendix D.

4 Experiment Setup

Data We use the publicly available subject data from Wang et al. [1]. Data was collected from 10 subjects (total 1,688 electrodes, with a mean of 167 electrodes per subject) who watched 26 movies (19 for pretraining, 7 for downstream decoding) while intracranial probes recorded their brain activity. To test decoding with arbitrary ensemble sizes, we select subsets of electrodes based on their individual linear task decodability, with the smallest subsets containing the electrodes with highest decodability. We follow the trialization and data preprocessing practices used in Wang et al. [1].

Decoding We evaluate the effectiveness of our pretrained PopT model by fine-tuning it on the four downstream decoding task used in the evaluation of Wang et al. [1]. Two of the tasks are audio focused: determining whether a word is spoken with a high or low pitch and determining whether a word is spoken loudly or softly. And two of the tasks have a more linguistic focus: determining whether the beginning of a sentence is occurring or determining whether any speech at all is occurring.

Our approach enables decoding on any arbitrary size of ensemble. We verify that our model is able to leverage additional channels for improved decoding performance that scales the number of inputs. To test this, we first order the electrodes by their individual linear decodability per task, and we increase the number of channels available to the model at fine-tuning time.

Baselines We want to determine whether the information about spatial relationships learned during pretraining was useful at fine-tuning time. For comparison, we concatenate the single-channel temporal embeddings and train a linear (Linear) or non-linear (DeepNN) aggregator on the decoding task. This sets a baseline for how much improvement is achievable from existing aggregation approaches [37]. To determine whether our performance can be attributed to using a more powerful architecture, we also fine-tune a PopT without pretraining, i.e. with randomly initialized weights.

5 Results

Decoding Performance We find that using a pretrained PopT consistently benefits downstream decoding compared to baseline channel aggregation techniques (Table 1). Additionally, while scaling performance with increasing number of channels is a challenging task for most baseline aggregation approaches, a pretrained PopT is able to scale well with increasing ensemble sizes (Figure 2a).

Model	**Pitch**	**Volume**	**Sent. Onset**	**Speech/Non-speech**
BrainBERT:				
Linear Agg.	0.59 ± 0.08	0.66 ± 0.08	0.70 ± 0.09	0.71 ± 0.11
Deep NN Agg.	0.58 ± 0.08	0.67 ± 0.08	0.71 ± 0.10	0.72 ± 0.10
Non-pretrained PopT	0.53 ± 0.06	0.61 ± 0.13	0.74 ± 0.10	0.70 ± 0.08
Pretrained PopT	$\mathbf{0.69 \pm 0.07}^*$	$\mathbf{0.84 \pm 0.06}^*$	$\mathbf{0.86 \pm 0.05}^*$	$\mathbf{0.89 \pm 0.07}^*$
TOTEM:				
Linear Agg.	0.55 ± 0.02	0.66 ± 0.03	0.79 ± 0.04	0.77 ± 0.05
Deep NN Agg.	0.57 ± 0.02	0.67 ± 0.03	0.78 ± 0.03	0.75 ± 0.05
Non-pretrained PopT	0.53 ± 0.02	0.64 ± 0.02	0.79 ± 0.03	0.77 ± 0.05
Pretrained PopT	$\mathbf{0.60 \pm 0.02}^*$	$\mathbf{0.73 \pm 0.02}^*$	$\mathbf{0.86 \pm 0.03}^*$	$\mathbf{0.84 \pm 0.06}^*$

Table 1: **Pretraining PopT is critical to downstream decoding performance** We test on a variety of audio-linguistic decoding tasks (see Section 4) with 90 channels as input. The temporal encoder used for aggregation in sections 1 and 2 are denoted in the section header. We also evaluate against an end-to-end pretrained iEEG model in section 3. Shown are the ROC-AUC mean and standard error across subjects. Best per section are bolded. Asterisks * indicate that the bolded model is significantly better than the second-place model ($p < 0.05$, Wilcoxon rank-sum).

To gain confidence on our method's generalizability to channel encoders, we applied our framework to two different channel encoders: (1) an sEEG temporal encoder (BrainBERT [1]) and (2) a general time-series encoder (TOTEM [2]). We see significant boosts in performance with the pretrained PopT in both cases when compared with baseline aggregation approaches, across all 4 auditory-linguistic tasks (Table 1). These results suggest that our framework can generalize to benefit joint aggregation of other single-channel embeddings and neural recording modalities.

Interpretability To analyze what our massively pretrained + fine-tuned model for sEEG data may be doing, we uncover the attention weights the model places on each input channel. We find agreement in our model's attention placement with brain regions typically involved in langauge processing (e.g. Wernicke's area), especially in the Speech vs. Non-speech downstream task (Figure 2b).

Efficiency To show that our technique is accessible to low data and compute regimes, we demonstrate that a pretrained PopT reaches the same decoding performance as other baseline approaches with an order of magnitude fewer samples and steps (Figure 2c and d). Pretraining PopT itself on more unnanotated data is also an order of magnitude more lightweight than pretraining existing end-to-end temporal-spatial models (see Appendix E). By focusing on population-level learning and leveraging the growing base of pretrained single-channel embedding techniques, our framework is efficient for learning new decoding tasks and continual pretraining.

Ablation of loss components and position information An ablation study confirms that both the network-wise and channel-wise component of the pretraining objective contribute to the downstream performance (Table 2). We also find that including the 3D position information for each channel is critical for decoding. Additionally, we find that the discriminative nature of our loss is necessary for decoding. Attempting to add an L1 reconstruction term to our pretraining objective results in poorer performance, perhaps because the model learns to overfit on low-entropy features in the embedding. Our discriminative loss requires the model to understand the embeddings in terms of how they can be distinguished from one another, which leads the model to extract more informative representations.

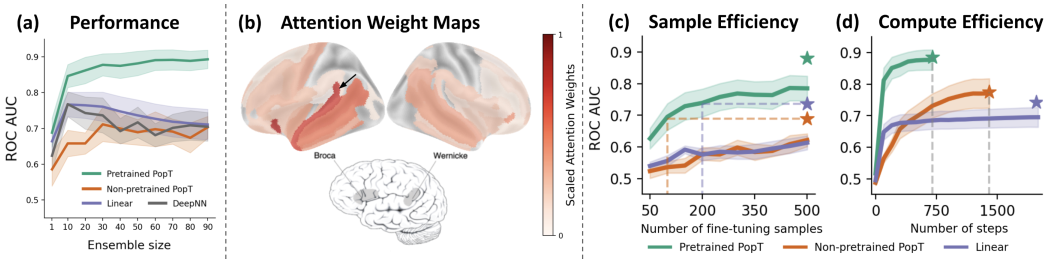

Figure 2: **(a) Pretrained PopT enables downstream performance scaling with ensemble size** Increasing channel ensemble size from 1 to 90 (x-axis), we see pretrained PopT (green) decoding performance (y-axis) not only beat non-pretrained approaches (orange, purple, grey), but also continually improve more with increasing channel count. Shaded bands show the standard error across subjects. **(b) Attention weights from a fine-tuned PopT identify candidate functional brain regions.** Candidate functional maps can be read from attention weights of a PopT fine-tuned on our decoding tasks. Note the weight placed on regions near Wernicke's area (black arrow) for this Speech vs. Non-speech tuned model. Lower brain figure highlight regions related to auditory-linguistic processing such as language production area Broca's area and language understanding Wernicke's area (adapted from [38]). **(c) Pretrained PopT is more sample efficient when fine-tuning.** Varying the number of samples available to each model at train time (x-axis), we see how the pretrained PopT is highly sample efficient, requiring only a fraction of samples to reach the full performance level of non pretrained aggregation approaches (dashed lines). Bands show standard error across test subjects. Stars indicate performance of the model trained on the full fine-tuning dataset. **(d) Pretrained PopT is consistently more compute efficient when fine-tuning.** Number of steps required for each model to reach final performance during fine-tuning (dashed lines). We find that pretrained PopT consistently requires fewer than 750 steps (each step is training on a batch size of 256) to converge, in contrast to the 2k steps required for the non pretrained PopT. Linear aggregation can be similarily compute efficient, but occasionally benefits from more training steps depending on dataset size. Bands show standard error across test subjects. Stars indicate fully trained performance.

	Pitch	Volume	Sent. Onset	Speech/Non-speech
PopT	$\mathbf{0.69 \pm 0.07}$	$\mathbf{0.84 \pm 0.06}$	$\mathbf{0.86 \pm 0.05}$	$\mathbf{0.89 \pm 0.07}$
PopT w/o group-wise loss	0.66 ± 0.07	0.83 ± 0.06	0.84 ± 0.04	0.88 ± 0.08
PopT w/o channel-wise loss	0.67 ± 0.06	0.81 ± 0.08	0.84 ± 0.06	0.87 ± 0.09
PopT w/o position encoding	0.59 ± 0.07	0.67 ± 0.10	0.75 ± 0.08	0.79 ± 0.08
PopT with reconstruction loss	0.60 ± 0.11	0.73 ± 0.11	0.81 ± 0.05	0.83 ± 0.09
PopT with L1 reconstruction only	0.56 ± 0.04	0.65 ± 0.08	0.73 ± 0.10	0.74 ± 0.10

Table 2: **PopT ablation study.** We individually ablate our losses and positional encodings during pretraining then decode on the resulting models. Shown are ROC-AUC mean and standard error across subjects. The best performing model across all decoding tasks uses all three of our proposed components, showing that they are all necessary. Removing our positional encoding during pretraining and fine-tuning drops the performance the most, indicating that position encoding is highly important for achieving good decoding. Additionally, we attempt adding a reconstruction component to the loss or purely using the L1 mask loss, but find that this leads to poorer performance (last two rows).

6 Conclusion

We presented a self-supervised learning scheme for learning effective representations of intracranial activity from temporal embeddings. By decoupling temporal and spatial feature extraction, we are able to leverage existing temporal embeddings to learn spatiotemporal representations efficiently and with a smaller number of parameters. We showed that self-supervised pretraining imbues our model with knowledge of spatial relationships between these embeddings and improved downstream decoding that scales with the number of available channels. This scheme produces interpretable weights from which attention weight maps can be read to help uncover learned relationships from the massively pretrained framework. Finally, we release the pretrained weights for our PopT with BrainBERT inputs as well as our code for pretraining with any temporal embedding.

References

[1] Christopher Wang, Vighnesh Subramaniam, Adam Uri Yaari, Gabriel Kreiman, Boris Katz, Ignacio Cases, and Andrei Barbu. Brainbert: Self-supervised representation learning for intracranial recordings. In *The Eleventh International Conference on Learning Representations*, 2022.

[2] Sabera Talukder, Yisong Yue, and Georgia Gkioxari. Totem: Tokenized time series embeddings for general time series analysis. *arXiv preprint arXiv:2402.16412*, 2024.

[3] Sina Faezi, Rozhin Yasaei, and Mohammad Abdullah Al Faruque. Htnet: Transfer learning for golden chip-free hardware trojan detection. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1484–1489. IEEE, 2021.

[4] Christian Herff, Dean J Krusienski, and Pieter Kubben. The potential of stereotactic-eeg for brain-computer interfaces: current progress and future directions. *Frontiers in neuroscience*, 14: 483258, 2020.

[5] Stephanie Martin, Iñaki Iturrate, José del R Millán, Robert T Knight, and Brian N Pasley. Decoding inner speech using electrocorticography: Progress and challenges toward a speech prosthesis. *Frontiers in neuroscience*, 12:367292, 2018.

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

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

[8] 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)*, pages 4171–4186, 2019.

[9] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021.

[10] Ran Liu, Mehdi Azabou, Max Dabagia, Jingyun Xiao, and Eva Dyer. Seeing the forest and the tree: Building representations of both individual and collective dynamics with transformers. *Advances in neural information processing systems*, 35:2377–2391, 2022.

[11] Sabera J Talukder and Georgia Gkioxari. Time series modeling at scale: A universal representation across tasks and domains. 2023.

[12] Hsiang-Yun Sherry Chien, Hanlin Goh, Christopher M Sandino, and Joseph Y Cheng. Maeeg: Masked auto-encoder for eeg representation learning. *arXiv preprint arXiv:2211.02625*, 2022.

[13] 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.

[14] Ke Yi, Yansen Wang, Kan Ren, and Dongsheng Li. Learning topology-agnostic eeg representations with geometry-aware modeling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[15] Armin Thomas, Christopher Ré, and Russell Poldrack. Self-supervised learning of brain dynamics from broad neuroimaging data. *Advances in Neural Information Processing Systems*, 35:21255–21269, 2022.

[16] Xuan Kan, Wei Dai, Hejie Cui, Zilong Zhang, Ying Guo, and Carl Yang. Brain network transformer. *Advances in Neural Information Processing Systems*, 35:25586–25599, 2022.

[17] Josue Ortega Caro, Antonio Henrique Oliveira Fonseca, Christopher Averill, Syed A Rizvi, Matteo Rosati, James L Cross, Prateek Mittal, Emanuele Zappala, Daniel Levine, Rahul M Dhodapkar, et al. Brainlm: A foundation model for brain activity recordings. *bioRxiv*, pages 2023–09, 2023.

[18] Antonis Antoniades, Yiyi Yu, Joseph Canzano, William Wang, and Spencer LaVere Smith. Neuroformer: Multimodal and multitask generative pretraining for brain data. *arXiv preprint arXiv:2311.00136*, 2023.

[19] 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, 2024.

[20] Chaoqi Yang, M Westover, and Jimeng Sun. Biot: Biosignal transformer for cross-data learning in the wild. *Advances in Neural Information Processing Systems*, 36, 2024.

[21] 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. URL https://openreview.net/forum?id=QzTpTRVtrP.

[22] Joel Ye, Jennifer Collinger, Leila Wehbe, and Robert Gaunt. Neural data transformer 2: multi-context pretraining for neural spiking activity. *Advances in Neural Information Processing Systems*, 36, 2024.

[23] Donghong Cai, Junru Chen, Yang Yang, Teng Liu, and Yafeng Li. Mbrain: A multi-channel self-supervised learning framework for brain signals. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23, page 130–141, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599426. URL https://doi.org/10.1145/3580305.3599426.

[24] Trung Le and Eli Shlizerman. Stndt: Modeling neural population activity with spatiotemporal transformers. *Advances in Neural Information Processing Systems*, 35:17926–17939, 2022.

[25] Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael Mendelson, Blake Richards, Matthew Perich, Guillaume Lajoie, and Eva Dyer. A unified, scalable framework for neural population decoding. *Advances in Neural Information Processing Systems*, 36, 2024.

[26] Chethan Pandarinath, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018.

[27] Brianna M Karpowicz, Yahia H Ali, Lahiru N Wimalasena, Andrew R Sedler, Mohammad Reza Keshtkaran, Kevin Bodkin, Xuan Ma, Lee E Miller, and Chethan Pandarinath. Stabilizing brain-computer interfaces through alignment of latent dynamics. *BioRxiv*, pages 2022–04, 2022.

[28] Alan D Degenhart, William E Bishop, Emily R Oby, Elizabeth C Tyler-Kabara, Steven M Chase, Aaron P Batista, and Byron M Yu. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. *Nature biomedical engineering*, 4(7):672–685, 2020.

[29] Justin Jude, Matthew G Perich, Lee E Miller, and Matthias H Hennig. Robust alignment of cross-session recordings of neural population activity by behaviour via unsupervised domain adaptation. feb 2022. doi: 10.48550. *arXiv preprint arXiv.2202.06159*.

[30] Xuan Ma, Fabio Rizzoglio, Kevin L Bodkin, Eric Perreault, Lee E Miller, and Ann Kennedy. Using adversarial networks to extend brain computer interface decoding accuracy over time. *elife*, 12:e84296, 2023.

[31] Sabera Talukder, Jennifer J Sun, Matthew Leonard, Bingni W Brunton, and Yisong Yue. Deep neural imputation: A framework for recovering incomplete brain recordings. *arXiv preprint arXiv:2206.08094*, 2022.

[32] Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. *arXiv preprint arXiv:2110.05357*, 2021.

[33] Geeling Chau, Yujin An, Ahamed Raffey Iqbal, Soon-Jo Chung, Yisong Yue, and Sabera Talukder. Generalizability under sensor failure: Tokenization+ transformers enable more robust latent spaces. *arXiv preprint arXiv:2402.18546*, 2024.

[34] Graham Wideman. Orientation and voxel-order terminology: Ras, las, lpi, rpi, xyz and all that, 2024. URL http://www.grahamwideman.com/gw/brain/orientation/orientterms.htm.

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

[36] Andy T. Liu, Shang-Wen Li, and Hung-yi Lee. TERA: self-supervised learning of transformer encoder representation for speech. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:2351–2366, 2021. doi: 10.1109/TASLP.2021.3095662. URL https://doi.org/10.1109/TASLP.2021.3095662.

[37] Gaurav R Ghosal and Reza Abbasi-Asl. Multi-modal prototype learning for interpretable multivariable time series classification. *arXiv preprint arXiv:2106.09636*, 2021.

[38] What is aphasia? — types, causes and treatment, Mar 2017. URL https://www.nidcd.nih.gov/health/aphasia.

[39] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[40] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[41] ildoonet. ildoonet/pytorch-gradual-warmup-lr: Gradually-warmup learning rate scheduler for pytorch, 2024. URL https://github.com/ildoonet/pytorch-gradual-warmup-lr.

[42] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

[43] Christophe Destrieux, Bruce Fischl, Anders Dale, and Eric Halgren. Automatic parcellation of human cortical gyri and sulci using standard anatomical nomenclature. *Neuroimage*, 53(1): 1–15, 2010.

[44] Nilearn, 2015. URL https://nilearn.github.io/stable/index.html.

A Architectures and training

Pretrained PopT The core Population Transformer consists of a transformer encoder stack with 6 layers, 8 heads. All layers ($N = 6$) in the encoder stack are set with the following parameters: $d_h = 512$, $H = 8$, and $p_{\text{dropout}} = 0.1$. We pretrain the PopT model with the LAMB optimizer [39] ($lr = 1e - 4$), with a batch size of $n_{\text{batch}} = 256$, and train/val/test split of 0.98, 0.01, 0.01 of the data. We pretrain for 500,000 steps, and record the validation performance every 1,000 steps. Downstream evaluation takes place on the weights with the best validation performance. We use the intermediate representation at the `[CLS]` token $d_h = 512$ and put a linear layer that outputs to $d_{out} = 1$ for fine-tuning on downstream tasks. These parameters for pretraining were the same for any PopT that needed to be pretrained (hold-one-out subject, subject subsets, ablation studies).

Non-pretrained PopT The architecture for the non-pretrained PopT is the same as the pretrained PopT (above). However, no pretraining is done, and the weights are randomly initialized with the default initializations.

Linear The linear baseline consists of a single linear layer that outputs to $d_{out} = 1$. The inputs are flattened and concatenated BrainBERT embeddings $d_{emb} = 756$ or TOTEM embeddings $d_{emb} = 64$ from a subset of channels $S \subset C$. Thus, the full input dimension is $d_{input} = d_{emb} * |S|$.

Deep NN The inputs are the same as above, but the decoding network now consists of 5 stacked linear layers, each with $d_h = 512$ and a GeLU activation.

Downstream Training For both PopT models, we train with these parameters: AdamW optimizer [40], $lr = 5e^{-4}$ where transformer weights are scaled down by a factor of 10 ($lr_t = 5e^{-5}$), $n_{batch} = 256$, a Ramp Up scheduler [41] with warmup 0.025 and Step LR gamma 0.99, reducing 100 times within the 2000 total steps that we train for. For Linear and DeepNN models, we train with these parameters: AdamW optimizer [40], $lr = 5e^{-4}$, $n_{batch} = 256$, a Ramp Up scheduler [41] with warmup 0.025 and Step LR gamma 0.95, reducing 25 times within the 17,000 total steps we train for. For all downstream decoding, we use a fixed train/val/test split of 0.8, 0.1, 0.1 of the data.

Compute Resources To run all our experiments (data processing, pretraining, evaluations, interpretability), one only needs 1 NVIDIA Titan RTXs (24GB GPU RAM). Pretraining PopT takes 2 days on 1 GPU. Our downstream evaluations take a few minutes to run each. For the purposes of gathering all the results in the paper, we parallelized the experiments on roughly 8 GPUs.

B Decoding tasks

We follow the same task specification as in Wang et al. [1], with the modification that the pitch and volume examples are determined by percentile (see below) rather than standard deviation in order to obtain balanced classes.

Pitch The PopT receives an interval of activity and must determine if it corresponds with a high or low pitch word being spoken. For the duration of a given word, pitch was extracted using Librosa's `piptrack` function over a Mel-spectrogram (sampling rate 48,000 Hz, FFT window length of 2048, hop length of 512, and 128 mel filters). For this task, for a given session, positive examples consist of words in the top-quartile of mean pitch and negative examples are the words in the bottom quartiles.

Volume The volume of a given word was computed as the average intensity of root-mean-square (RMS) (`rms` function, frame and hop lengths 2048 and 512 respectively). As before, positive examples are the words in the top-quartile of volume and negative examples are those in the bottom quartiles.

Sentence onset Negative examples are intervals of activity from 1s periods during which no speech is occurring in the movie. Positive examples are intervals of brain activity that correspond with hearing the first word of a sentence.

Speech vs. Non-speech Negative examples are as before. Positive examples are intervals of brain activity that correspond with dialogue being spoken in the stimuli movie.

C Data

Subj.	Age (yrs.)	# Electrodes	Movie	Recording time (hrs)	Held-out
1	19	91	Thor: Ragnarok	1.83	
			Fantastic Mr. Fox	1.75	
			The Martian	0.5	x
2	12	100	Venom	2.42	
			Spider-Man: Homecoming	2.42	
			Guardians of the Galaxy	2.5	
			Guardians of the Galaxy 2	3	
			Avengers: Infinity War	4.33	
			Black Panther	1.75	
			Aquaman	3.42	x
3	18	91	Cars 2	1.92	x
			Lord of the Rings 1	2.67	
			Lord of the Rings 2 (extended edition)	3.92	
4	9	135	Megamind	2.58	
			Toy Story	1.33	
			Coraline	1.83	x
5	11	205	Cars 2	1.75	x
			Megamind	1.77	
6	12	152	Incredibles	1.15	
			Shrek 3	1.68	x
			Megamind	2.43	
7	6	109	Fantastic Mr. Fox	1.5	
8	4.5	72	Sesame Street Episode	1.28	
9	16	102	Ant Man	2.28	
10	12	173	Cars 2	1.58	x
			Spider-Man: Far from Home	2.17	

Table 3: **Subject statistics** Subjects used in PopT training, and held-out downstream evaluation. Table taken from [1]. The number of uncorrupted, electrodes that can be Laplacian re-referenced are shown in the second column The average amount of recording data per subject is 4.3 (hrs).

D Interpretation Methods

Candidate functional brain regions from attention weights After fine-tuning our weights on a decoding task, we can examine the attention weights of the `[CLS]` output for candidate functional brain regions. We obtain a normalized Scaled Attention Weight metric (see next section) across all subjects to be able to analyze candidate functional brain regions across sparsely sampled subject datasets. The Scaled Attention Weight is computed from raw attention weights at the `[CLS]` token passed through the attention rollout algorithm [42]. The resulting weights from each channel are then grouped by brain region according to the Destrieux layout [43].

Scaled Attention Weight First, we obtain an attention weight matrix across all trials which includes weights between all tokens. Then, we perform attention rollout [42] across layers to obtain the contributions of each input channel by the last layer. We take the resulting last layer of rollout weights for all channels, where the target is the `[CLS]` token, normalize within subject, and scale by ROC AUC to obtain the Scaled Attention Weight per channel. Finally, we plot the 0.75 percentile weight per region, as mapped by the Destrieux atlas [43] using Nilearn [44].

E Model and Compute Requirements

	e5	e50	e90
PopT		20M	
Deep NN	3M	20M	36M
Linear	3.8k	38k	69k
Brant [19]		500M	
LaBraM [21]		350M	

Table 4: **Parameter counts**. Since PopT takes existing temporal embeddings as input, the number of parameters that must be trained is an order of magnitude less than recent end-to-end approaches.

	GPU count	GPU type	Time to train	TFLOPS
PopT	1	NVIDIA TITAN RTX (24GB)	2 days	2.1M
Brant [19]	4	NVIDIA Tesla A100 (80G)	2.8 days	18.8M
LaBraM [21]	8	NVIDIA Tesla A800 (40G)	–	–

Table 5: **Pretraining compute requirements** Based on published train times (none were given for LaBraM) it is evident that PopT has smaller hardware and shorter training time requirements.