Software Requirements Specification for Model–to–Brain Alignment for Speech Recognition

Xiao Shao

April 21, 2026

Contents

Revision History

Date	Version	Notes
Jan 29	1.0	Initial draft
Feb 2	1.1	Update data explaniation
Mar 1	1.2	Update based on comments
Apr 21	1.3	Update based on comments

1 Reference Material

This section records information for easy reference, including units, symbols, abbreviations, and mathematical conventions used throughout this SRS.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In addition to the basic units, several derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	**quantity**	**SI name**
s	time	second
Hz	sampling rate / frequency	hertz
T	magnetic flux density	tesla

Table 1: SI units used in this document

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	**unit**	**description**
C	–	Number of MEG sensors.
D	–	Predictor feature dimension.
B	–	Training batch size.
E	–	Training epochs.
$\mathcal{D}$	–	Speech dataset collection (e.g., $\mathcal{D}_{\text{Burgundy}}, \mathcal{D}_{\text{LibriSpeech}}$).
$\mathcal{D}_{\text{train}}$	–	Training splits of dataset $\mathcal{D}$.
$\mathcal{D}_{\text{val}}$	–	Validation splits of dataset $\mathcal{D}$.
$\mathcal{D}_{\text{test}}$	–	Test splits of dataset $\mathcal{D}$.
$\boldsymbol{a}$	–	Audio input sequence provided to the speech model.
$\boldsymbol{s}^{\star}$	–	Ground-truth transcript for ASR training.

Continued on next page

symbol	unit	description
$\boldsymbol{\theta}$	–	Trainable parameters of the speech model.
$\boldsymbol{\theta}^*$	–	Optimized model parameters.
$f_{\boldsymbol{\theta}}(\cdot)$	–	Speech model parameterized by $\boldsymbol{\theta}$.
$\mathcal{L}_{\text{ASR}}$	–	ASR training loss function used to fit $f_{\boldsymbol{\theta}}$.
η	–	Learning rate for gradient-based optimization.
ℓ	–	Layer index of the deep model.
H_ℓ	–	Hidden-state dimensionality of layer ℓ.
$\boldsymbol{h}_t^{(\ell)}$	–	Hidden state vector from layer ℓ at model time index t.
$g(\cdot)$	–	Optional transformation applied to hidden states to form predictors.
$\boldsymbol{r}_t^{(\ell)}$	–	Predictor representation vector derived from $\boldsymbol{h}_t^{(\ell)}$.
$\mathbf{R}^{(\ell)}$	–	Time-by-feature predictor matrix aligned to MEG grid, $\mathbf{R}^{(\ell)} \in \mathbb{R}^{N\times D}$.
$\mathbf{R}^{(\text{base})}$	–	Baseline predictor matrix, $\mathbf{R}^{(\text{base})} \in \mathbb{R}^{N\times D_b}$.
D_b	–	Feature dimension of the baseline predictor matrix $\mathbf{R}^{(\text{base})}$.
$\mathbf{X}(\mathbf{R})$	–	Lagged design matrix constructed from predictor matrix $\mathbf{R}$, $\mathbf{X}(\mathbf{R}) \in \mathbb{R}^{N\times(KD)}$.
$\mathbf{W}^*$	–	Ridge-regularized estimate of TRF weights.
ρ_{base}	–	Encoding score obtained using the baseline predictor.
ρ_ℓ	–	Encoding score obtained using model-derived representations from layer ℓ.
$\Delta\rho_\ell$	–	Improvement over baseline: $\Delta\rho_\ell = \rho_\ell - \rho_{\text{base}}$.
f_s	Hz	Sampling frequency of MEG signals used for TRF modeling.
f_a	Hz	Sampling frequency of the raw audio waveform.
K	–	Number of time lags used in the TRF.
L	–	Number of candidate predictors being compared.
N	–	Number of time samples used for model fitting.
N_{ROI}	–	Number of regions of interests (ROI).
t	s	Continuous time index.
Δt	s	Sampling interval of MEG signals ($\Delta t = 1/f_s$).
τ	s	Time-lag variable for TRF, which whithin $[\tau_{\min}, \tau_{\max}]$.
$\tau_{\min}$, $\tau_{\max}$	s	Start and end of the TRF lag window.
$\boldsymbol{x}(t)$	–	Predictor vector at time t.
$\mathbf{R}^{(\ell)}$	–	Time-by-feature matrix.
$\mathbf{R}^{(\text{base})}$	–	Baseline predictor.

Continued on next page

symbol	unit	description
$\mathbf{X}$	–	Design matrix formed from time-lagged predictors.
$\mathbf{X}_\tau$	–	Lagged version of $\mathbf{X}$ corresponding to a specific lag τ.
$\boldsymbol{y}(t)$	T/fT	MEG response vector across sensors at time t.
$\mathbf{Y}$	T/fT	MEG response matrix across time and sensors.
$\mathbf{w}(\tau)$	–	TRF weight vector at lag τ, which used to map predictors to responses.
$\mathbf{W}$	–	Full TRF weight matrix across all lags and sensors.
λ	–	Ridge regularization coefficient for TRF fitting.
$\hat{\mathbf{Y}}$	T/fT	Predicted MEG responses from the fitted encoding model.
ρ	–	Prediction score.
ρ_{CV}	–	Cross-validation procedure.

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
API	Application Programming Interface
BIDS	Brain Imaging Data Structure
CI	Continuous Integration
CLI	Command Line Interface
CV	Cross-Validation
DL	Deep Learning
EEG	Electroencephalography
HPC	High-Performance Computing
MEG	Magnetoencephalography
mTRF	Multivariate Temporal Response Function
Req	Requirement
ROI	Region of Interest
SRS	Software Requirements Specification
TRF	Temporal Response Function
WER	Word Error Rate

Table 3: Abbreviations and acronyms used in this document

1.4 Mathematical Notation

The following conventions are used.

- **Typesetting:** Scalars are written in italic (e.g., t, f_s, λ), vectors in bold lowercase (e.g., $\boldsymbol{x}$), and matrices in bold uppercase (e.g., $\mathbf{X}, \mathbf{W}$).
- **Time indices:** Discrete-time samples are indexed by $n \in \{1, \ldots, N\}$. Continuous time is denoted by t.
- **Lag window:** A time-lag window is defined by $[\tau_{\min}, \tau_{\max}]$ and discretized into K lags.
- **Datasets:** We denote the speech dataset used for training and evaluation as $\mathcal{D} \in \{\mathcal{D}_{\text{Burgundy}}, \mathcal{D}_{\text{LibriSpeech}}\}$ with splits $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{D}_{\text{test}}$.
- **Deep model training objective:** A speech model with parameters $\boldsymbol{\theta}$ is trained to minimize an ASR-style loss on $\mathcal{D}_{\text{train}}$:

  $$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{a}, \boldsymbol{s}^\star) \sim \mathcal{D}_{\text{train}}} \left[\mathcal{L}_{\text{ASR}}(f_{\boldsymbol{\theta}}(\boldsymbol{a}), \boldsymbol{s}^\star) \right],$$

  where $\boldsymbol{a}$ is an audio input, $\boldsymbol{s}^\star$ is the target transcript, and $\mathcal{L}_{\text{ASR}}$ is the loss function depend on the selected model.
- **Gradient-based optimization:**

  $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{ASR}}(\boldsymbol{\theta}),$$

  with learning rate η, applied over mini-batches of size B for E epochs.
- **Model representations:** Given a trained model $f_{\boldsymbol{\theta}^*}$, internal representations are extracted. Let $\boldsymbol{h}_t^{(\ell)} \in \mathbb{R}^{H_\ell}$ denote the hidden state from layer ℓ at model time index t. A predictor representation is defined as

  $$\boldsymbol{r}_t^{(\ell)} = g\left(\boldsymbol{h}_t^{(\ell)}\right) \in \mathbb{R}^D,$$

  where $g(\cdot)$ is an optional transformation chosen to yield a consistent predictor dimension D.
- **Time alignment to MEG grid:** Representations $\{\boldsymbol{r}_m^{(\ell)}\}$ are resampled and aligned to the MEG sampling grid at rate f_s to produce a time-by-feature matrix $\mathbf{R}^{(\ell)} \in \mathbb{R}^{N \times D}$. In the same way, an acoustic baseline predictor (e.g., gammatone features) yields $\mathbf{R}^{(\text{base})} \in \mathbb{R}^{N \times D_b}$.
- **Lagged design matrix construction:** For a given predictor matrix $\mathbf{R}$, a lagged design matrix $\mathbf{X}(\mathbf{R}) \in \mathbb{R}^{N \times (KD)}$ is formed by concatenating time-shifted copies of $\mathbf{R}$ over lags in $[\tau_{\min}, \tau_{\max}]$.

- **TRF encoding model:** A linear TRF maps lagged predictors to MEG responses:

$$\hat{\mathbf{Y}} = \mathbf{X}(\mathbf{R})\,\mathbf{W},$$

where $\mathbf{Y} \in \mathbb{R}^{N \times C}$ is the MEG response matrix and $\mathbf{W} \in \mathbb{R}^{(KD) \times C}$ are TRF weights.

- **Ridge-regularized estimation:**

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \|\mathbf{Y} - \mathbf{X}(\mathbf{R})\mathbf{W}\|_F^2 + \lambda \|\mathbf{W}\|_F^2,$$

where λ is selected using a fixed cross-validation procedure shared across all predictors for fair comparison.

- **Encoding performance metric:** For each sensor/ROI and CV fold, predictor performance is summarized using a metric such as Pearson correlation $\rho(\hat{\mathbf{Y}}, \mathbf{Y})$ and then aggregated across folds.

- **Baseline comparison and improvement:** Let ρ_{base} denote the encoding score obtained with the baseline predictor and ρ_ℓ the score obtained using model-derived representations from layer ℓ. The improvement over baseline is defined as

$$\Delta\rho_\ell = \rho_\ell - \rho_{\text{base}}.$$

The software reports $\Delta\rho_\ell$ (and optionally confidence intervals) to quantify how much each model representation improves encoding performance.

2 Introduction

Human speech communication is remarkably efficient, yet the computational mechanisms by which the brain recognizes continuous speech remain only partially understood. Modern deep learning models for speech recognition (e.g., RNN/LSTM-based architectures) can achieve strong behavioral performance, but it is unclear which of their internal representations correspond to the neural representations that unfold in the brain over time. Magnetoencephalography (MEG) offers millisecond-scale measurements of neural activity during naturalistic listening, enabling direct tests of whether candidate speech representations encode information in a brain-like manner.

A common approach to model–brain comparison is to treat the model's time-varying representations as predictors and quantify how well they explain neural responses using encoding models such as multivariate Temporal Response Functions (mTRFs). However, producing trustworthy comparisons across multiple models and predictors is difficult in practice. The difficulties are primarily engineering and reproducibility challenges: training models across datasets, extracting layer-wise hidden representations, aligning all predictors to the MEG time axis, fitting mTRFs under identical cross-validation and regularization settings, and

generating consistent summaries and visualizations. Without a standardized workflow, comparisons can be confounded by subtle differences in preprocessing, alignment, or evaluation procedures rather than reflecting true representational differences.

This project develops a reusable model–to–brain alignment pipeline for speech recognition. The software supports training multiple deep learning models on speech dataset (e.g., Burgundy and LibriSpeech), extracting hidden representations as predictors, constructing time-aligned predictor matrices, fitting ridge-regularized mTRF encoding models, and reporting how much each model-derived predictor improves neural prediction relative to an acoustic baseline.

2.1 Purpose of Document

The purpose of this Software Requirements Specification (SRS) is to provide a precise, testable, and shared description of the requirements for the Model–to–Brain Alignment for Speech Recognition software system. This document serves as a reference for developers, to ensure a shared understanding of the software requirements. It also provides background information and planning for design, implementation, and validation.

2.2 Scope of Requirements

The scope of this project covers (i) training selected deep learning-based speech models on supported speech dataset, (ii) extracting time-resolved hidden representations from trained models, (iii) transforming and aligning these representations to MEG recordings, (iv) fitting and evaluating mTRF encoding models under standardized cross-validation and regularization procedures, and (v) generating quantitative summaries and figures for scientific interpretation. The project does not aim to provide novel MEG preprocessing algorithms, it assumes MEG data are available in a usable, preprocessed form.

2.3 Characteristics of Intended Reader

The intended readers of this SRS are the stakeholders who will review, maintain, and use this document to design and implement Model–to–Brain Alignment for Speech Recognition. These readers are expected to have the following background and skills:

- **Software engineering:** Experience designing and implementing scientific computing software in Python, including modular code organization, configuration-driven workflows, and basic familiarity with testing practices and version control.
- **Numerical methods and data handling:** Comfort with linear algebra and numerical computation at the level of a fourth-year undergraduate course (SFWRENG 4AL3, COMPSCI 4ML3) in machine learning, including matrix and vector operations, regularized linear regression concepts, and handling large arrays. Readers should be familiar with common scientific Python libraries and data formats used in research pipelines.

- **Speech and audio processing engineering:** Practical understanding of digital signal processing at the level of a third-year course in signals and systems (SFWRENG 3MX3), sufficient to work with sampling rates, resampling, time–frequency features.
- **Deep learning technology:** Ability to train and test deep learning-based models for speech at a graduate machine learning level (CHEM ENG 787), including familiarity with backpropagation algorithm and extracting hidden representations from trained models.
- **Neural data analysis:** Familiarity with MEG data as time series measured at multiple sensors and basic preprocessing outcomes. Readers should understand the purpose of encoding models and cross-validation, but do not need expertise in MEG source reconstruction or advanced neurophysiology.

2.4 Organization of Document

This document follows a abstract-to-specific approach. It first presents high-level goals, models and scope in the real world, then derivte the functional and nonfunctional requirements from them. For readers that are already familiar with the M/EEG domain, they can start with Section 5 and look for clarifications in previous sections. This document follows the template for an SRS for scientific computing software proposed by Smith and Lai (2005); Smith et al. (2007); Smith and Koothoor (2016).

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

This system context is shown as inputs and outputs in Figure 1. This project relies on Eelbrain as an external library for mTRF calculation and visualization Brodbeck et al. (2023).

- User Responsibilities:
  - Provide speech experiment materials as input audio.
  - Provide neural experiment materials (MEG dataset).
  - Provide experiment configurations.
- Model–to–Brain Alignment for Speech Recognition Responsibilities:
  - Produce training pipeline for different models.

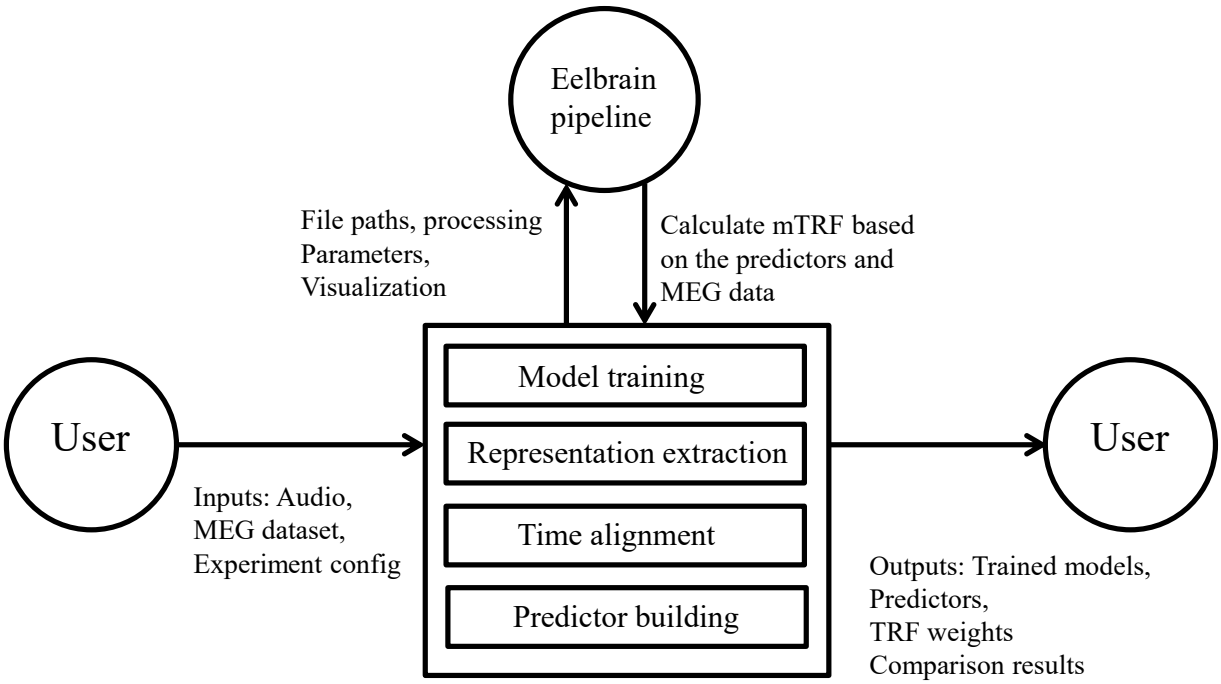

Figure 1: System Context

- Extract representation features from trained models.
- Produce time-aligned process for different models.
- Produce predictors that are compatible with the neural time base.

The software will typically be used for neuroscience research. It will not be used for mission-critical or safety-critical work.

3.2 User Characteristics

The end user of Model–to–Brain Alignment for Speech Recognition should have experience in:

- Python programming. Experience in Python programming equivalent to COMPSCI 1MD3 (Introduction to Programming) or SFWRENG 2OP3 (Object-Oriented Programming);

- Deep learning programming; Experience in deep learning programming equivalent to SFWRENG 4AL3, COMPSCI 4ML3.
- Neuroimaging data, particularly MEG/EEG data and its workflow;

3.3 System Constraints

- **Python program:** Model–to–Brain Alignment for Speech Recognition shall be implemented in Python.
  - Rationale: Python provides the most comprehensive ecosystem for scientific computing and is the primary language supported by the required neuroimaging libraries (Eelbrain).
- **Deep learning method:** Model training and representation extraction shall use PyTorch.
  - Rationale: PyTorch offers the flexibility needed for custom neural network architectures and has extensive community support for speech recognition research.
- **TRF backend:** Encoding analysis and visualization shall use the Eelbrain TRF pipeline.
  - Rationale: Eelbrain provides established tools for temporal response function analysis, which are required for relating model-derived features to time-resolved MEG responses.
- **MEG dependency:** The Eelbrain pipeline shall delegate MEG data structures and preprocessing operations to MNE-Python.
  - Rationale: MNE-Python is the standard library for MEG data handling and preprocessing, and it is required for compatibility with the selected Eelbrain-based analysis workflow.

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally the instance models.

4.1 Problem Description

The Model–to–Brain Alignment pipeline is intended to solve the problem of reproducibly comparing candidate speech representations by how well they explain neural responses during continuous speech perception. In practice, researchers often have multiple competing

candidate representations (e.g., acoustic features or hidden features from trained speech models). The core scientific question is to explore which representations are most predictive of neural measurements, under a fair and consistent evaluation protocol.

The difficulty is that the comparison requires multiple steps that can introduce confounds if handled inconsistently. For example, representations must be defined at an appropriate temporal resolution, aligned to the neural recording time base, and evaluated using the same modeling assumptions. Without a standardized and traceable workflow, observed differences in neural predictivity may reflect differences in preprocessing or evaluation rather than genuine representational differences.

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- **Speech stimulus:** The auditory signal (wav file) presented to participants, together with associated labeled transcript.
- **Neural response:** Time-resolved measurements of brain activity recorded during listening, assumed to be preprocessed into a usable form.
- **Representation:** A time-indexed representation features derived from the stimulus, such as acoustic features and internal states of a deep learning model.
- **Predictor:** A representation expressed on the same time grid as the neural response, used as an explanatory variable in an encoding model.
- **Time alignment:** The process of mapping a representation's time indices into the neural recording time base, which aims to make predictor samples correspond to neural samples.
- **Speech model:** The deep learning model that predicts neural responses from input audio.
- **Lag window:** The range of temporal delays between predictor and neural response considered by the speech model, capturing response latency.
- **Cross validation:** A procedure for estimating predictive performance on unseen data by splitting data into training and testing partitions.
- **Region of Interest:** A predefined subset of sensors or sources used to summarize neural responses.
- **Comparison result:** A quantitative summary indicating how much a candidate predictor improves neural prediction relative to the baseline under the same evaluation protocol.

4.1.2 Physical System Description

The physical system of Model–to–Brain Alignment for Speech Recognition, as shown in Figure 2, includes the following elements:

PS1: Neural current sources. Electrical currents generated by neuronal populations within the cerebral cortex. These currents are the primary sources of the electromagnetic fields measured in MEG.

PS2: Head volume conductor. The human head acts as a volume conductor for electromagnetic fields. The geometry and conductivity properties of these tissues influence how fields generated by neural sources propagate toward the sensors.

PS3: Sensor system. MEG sensors positioned on the head that measure magnetic fields resulting from neural activity.

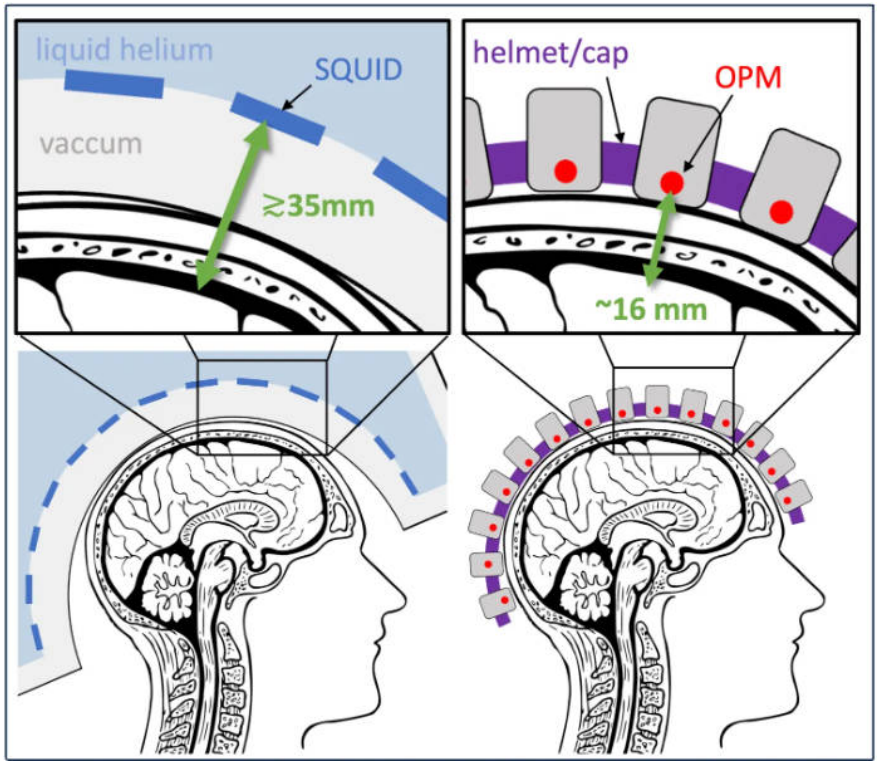

Figure 2: Physical System of MEG

4.1.3 Goal Statements

Given (i) speech stimuli, (ii) neural responses, and (iii) a set of candidate speech representations extracted from acoustic baselines and trained models, the system shall produce outputs that support model-to-brain comparison.

GS1: Produce a standardized and time-aligned predictor representation for each candidate speech representation.

GS2: Produce encoding-model evaluation results that quantify how well each predictor explains neural responses.

GS3: Produce comparison results showing how each candidate representation performs relative to an acoustic baseline.

GS4: Produce traceable analysis artifacts that allow the evaluation results to be inspected and reproduced.

4.2 Solution Characteristics Specification

The instance models that govern Model–to–Brain Alignment for Speech Recognition are presented in Subsection 4.2.5. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Assumptions

A1: Over the selected lag window, the relationship between a predictor representation and the neural response can be approximated by a linear time-lagged encoding model. [TM:TRF, IM4, IM5, IM6, IM7]

A2: Within each analysis segment, the mapping from predictors to neural responses is treated as time-invariant for model fitting. [TM:TRF, TM:Ridge, IM5, IM6, IM7]

A3: After preprocessing and resampling, the temporal alignment error between speech-derived predictors and MEG responses is negligible relative to the analysis resolution and can be absorbed within the chosen lag window. [IM3, IM4, IM7]

A4: The analyzed neural response is assumed to depend primarily on the supplied predictor set and its recent temporal history, so effects outside the selected lag window are ignored. [TM:TRF, IM4, IM6, IM7]

4.2.2 Theoretical Models

This section focuses on the general equations and laws that Model–to–Brain Alignment for Speech Recognition is based on.

RefName: **TM:Training**

Label: Empirical risk minimization for learning speech representations

Equation: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(\boldsymbol{a},\boldsymbol{y})\in\mathcal{D}_{\text{train}}} \mathcal{L}_{\text{train}}(f_{\boldsymbol{\theta}}(\boldsymbol{a}), \boldsymbol{y})$

Description: This theoretical model states that the parameters of a speech model can be learned by minimizing an empirical training loss over a training dataset. Here $\boldsymbol{a}$ denotes the speech input, $\boldsymbol{y}$ denotes the supervision signal associated with the task, $f_{\boldsymbol{\theta}}$ is the model parameterized by $\boldsymbol{\theta}$, and $\mathcal{L}_{\text{train}}$ is the task-specific training loss. In Model–to–Brain Alignment for Speech Recognition, the purpose of training is to obtain a model whose internal representations can later be extracted and compared in neural encoding analysis.

Notes: None.

Source: https://docs.pytorch.org/docs/stable/optim.html

Ref. By: IM1

Preconditions for TM:Training: A training dataset $\mathcal{D}_{\text{train}}$ and a training loss $\mathcal{L}_{\text{train}}$ are defined.

Derivation for TM:Training: Not Applicable

RefName: TM:Representation

Label: Model-extracted representations as candidate predictors

Equation: $\boldsymbol{r}^{(\ell)}(t) = g\Big(\boldsymbol{h}^{(\ell)}(t)\Big) \in \mathbb{R}^D$

Description: This theoretical model defines a candidate predictor as a time-indexed feature vector derived from an internal model representation. The hidden features of layer ℓ is denoted by $\boldsymbol{h}^{(\ell)}(t)$, and $g(\cdot)$ is an optional transformation used to obtain a consistent predictor dimension D across models and layers. After time alignment to the MEG sampling grid, $\boldsymbol{r}^{(\ell)}(t)$ is treated as an explanatory signal in the speech model.

Notes: None.

Source: https://doi.org/10.1371/journal.pcbi.1013244

Ref. By: IM3

Preconditions for TM:Representation: A trained model and a well-defined layer index ℓ exist for the stimulus.

Derivation for TM:Representation: Not Applicable

4.2.3 General Definitions

This section collects the laws and equations that will be used in building the instance models.

4.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models.

Number	DD1
Label	**Sampling interval of MEG signals**
Symbol	Δt
SI Units	s
Equation	$\Delta t = 1/f_s$
Description	The neural sampling interval Δt is defined by the neural sampling rate f_s. This quantity determines the discrete-time grid used for alignment and TRF lag discretization.
Sources	None
Ref. By	Section 4.2.6

Number	DD2
Label	**Discretized TRF lag grid**
Symbol	$\{\tau_k\}_{k=1}^{K}$
SI Units	s
Equation	$\tau_k = \tau_{\min} + (k-1)\Delta t, \quad k = 1, \ldots, K$
Description	The continuous lag variable τ is restricted to the lag window $[\tau_{\min}, \tau_{\max}]$ and discretized onto the MEG sampling grid with step size Δt. The number of lags K is chosen such that $\tau_1 = \tau_{\min}$ and $\tau_K \leq \tau_{\max}$.
Sources	None
Ref. By	Section 4.2.6

Number	DD3
Label	**Discrete mTRF prediction**
Symbol	$\hat{\mathbf{Y}}$
SI Units	T/fT
Equation	$\hat{\mathbf{Y}} = \mathbf{X}\mathbf{W}$
Description	Let $\mathbf{Y} \in \mathbb{R}^{N \times C}$ be the observed MEG response matrix over C sensors and N time samples. The predicted MEG responses $\hat{\mathbf{Y}}$ are obtained by applying a linear mapping from the lagged design matrix $\mathbf{X}$ to MEG channels using the TRF weight matrix $\mathbf{W} \in \mathbb{R}^{(KD) \times C}$.
Sources	None
Ref. By	Section 4.2.6

Number	DD4
Label	**Ridge objective for TRF fitting**
Symbol	$\mathcal{L}$
SI Units	–
Equation	$\mathcal{L}(\mathbf{W}) = \|\mathbf{Y} - \mathbf{X}\mathbf{W}\|_F^2 + \lambda \|\mathbf{W}\|_F^2$
Description	The objective $\mathcal{L}$ defines TRF parameter estimation as ridge-regularized least squares. The scalar $\lambda \geq 0$ controls the strength of ℓ_2 regularization. Minimizing $\mathcal{L}$ yields fitted weights $\mathbf{W}$ used in the prediction equation (DD3).
Sources	None
Ref. By	DD5, DD6

Number	DD5
Label	**Prediction score**
Symbol	ρ
SI Units	–
Equation	$\rho = \mathrm{corr}\left(\hat{\mathbf{Y}}, \mathbf{Y}\right)$
Description	The prediction score ρ summarizes model performance by computing the Pearson correlation between predicted MEG responses $\hat{\mathbf{Y}}$ and observed MEG responses $\mathbf{Y}$. Correlation may be computed per sensor and optionally aggregated over sensors or ROIs using a fixed rule defined by the experiment specification.
Sources	None
Ref. By	DD6

Number	DD6
Label	**Cross-validation score aggregation**
Symbol	ρ_{CV}
SI Units	–
Equation	$\rho_{\mathrm{CV}} = \frac{1}{J}\sum_{j=1}^{J} \rho^{(j)}$
Description	Cross-validation estimates generalization performance by evaluating a score on held-out data across J folds. Let $\rho^{(j)}$ be the score computed on the test split of fold j using the same lag window $[\tau_{\min}, \tau_{\max}]$ and egularization-selection procedure for all predictors. The aggregate score ρ_{CV} is the mean across folds.
Sources	None
Ref. By	Section 4.2.6

4.2.5 Instance Models

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.4 to replace the abstract symbols in the models identified in Sections 4.2.2 and 4.2.3.

The goals GS1–GS4 are solved by the instance models IM3–IM7, which define (i) how predictors are aligned and assembled into TRF-ready inputs, (ii) how TRF models are fit un-

der a fixed protocol, (iii) how performance is evaluated fairly using cross-validation, (iv) how baseline-referenced gains are computed, and (v) how artifacts are defined for reproducibility.

Number	IM1
Label	**Training a deep learning speech model**
Input	Training dataset $\mathcal{D}_{\text{train}}$, model family $f_{\boldsymbol{\theta}}$, training loss $\mathcal{L}_{\text{train}}$, and optimization procedure Ω.
Output	Trained model parameters $\boldsymbol{\theta}^*$ that minimize the empirical training objective: $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{\lvert\mathcal{D}_{\text{train}}\rvert} \sum_{(\boldsymbol{a},\boldsymbol{y})\in\mathcal{D}_{\text{train}}} \mathcal{L}_{\text{train}}(f_{\boldsymbol{\theta}}(\boldsymbol{a}), \boldsymbol{y})$.
Description	This model instantiates TM:Training by learning model parameters from a concrete training dataset. The resulting trained model serves as the source of hidden representations that will later be extracted as candidate predictors for model-to-brain alignment.
Sources	Standard deep learning optimization workflow.
Ref. By	IM2

Number	IM2
Label	**Extraction of hidden-state predictors from a trained model**
Input	Trained model parameters $\boldsymbol{\theta}^*$ (IM1), speech stimulus $\boldsymbol{a}$, layer index ℓ, and optional transformation $g(\cdot)$.
Output	Predictor stream $\boldsymbol{r}^{(\ell)}(t)$ defined by $\boldsymbol{r}^{(\ell)}(t) = g\Big(\boldsymbol{h}^{(\ell)}(t)\Big) \in \mathbb{R}^D$.
Description	This model instantiates TM:Representation by applying a trained deep learning model to the speech stimulus, extracting the hidden representation at layer ℓ, and optionally transforming it into a standardized predictor stream. The resulting predictor is then passed to the alignment stage.
Sources	Internal representation extraction from trained neural networks.
Ref. By	IM3

Number	IM3
Label	**Time alignment and standardization of predictors**
Input	Predictor stream $\boldsymbol{r}^{(\ell)}(t)$ from TM:Representation, and MEG sampling frequency f_s.
Output	Time-aligned predictor matrix $\tilde{\mathbf{R}} \in \mathbb{R}^{N \times D}$ on the MEG sampling grid
Description	This model refines TM:Representation by mapping the abstract predictor stream $\boldsymbol{r}^{(\ell)}(t)$ onto the MEG time base, producing a standardized predictor matrix $\tilde{\mathbf{R}}$ for downstream TRF analysis.
Sources	https://eelbrain.readthedocs.io/en/stable/generated/eelbrain.align.html
Ref. By	IM4

Number	IM4
Label	**Lagged design matrix for TRF fitting**
Input	Time-aligned predictor matrix $\tilde{\mathbf{R}}$ (IM3), lag window $[\tau_{\min}, \tau_{\max}]$, and discretized lag grid $\{\tau_k\}_{k=1}^{K}$ (DD2).
Output	Lagged design matrix $\mathbf{X} \in \mathbb{R}^{N \times (KD)}$.
Description	This model operationalizes the lagged design matrix definition using the common lag window and discretization shared across all predictors.
Sources	https://doi.org/10.7554/eLife.85012
Ref. By	IM5

Number	IM5
Label	**Ridge-regularized mTRF fitting**
Input	Design matrix $\mathbf{X}$ (IM4), observed MEG responses $\mathbf{Y} \in \mathbb{R}^{N \times C}$, and regularization coefficient λ (DD4).
Output	Fitted TRF weights $\mathbf{W}^* \in \mathbb{R}^{(KD) \times C}$ that minimize the ridge objective: $\mathbf{W}^* = \arg\min_{\mathbf{W}} \|\mathbf{Y} - \mathbf{X}\mathbf{W}\|_F^2 + \lambda \|\mathbf{W}\|_F^2$.
Description	This model specifies the fitting rule for TRF weights using ridge-regularized least squares. The same estimation formulation is used for all predictors to ensure comparability.
Sources	https://doi.org/10.7554/eLife.85012
Ref. By	IM6

Number	IM6
Label	**Predicted MEG responses from fitted TRF**
Input	Design matrix $\mathbf{X}$ and fitted weights $\mathbf{W}^*$ (IM5).
Output	Predicted MEG responses $\hat{\mathbf{Y}}$ given by $\hat{\mathbf{Y}} = \mathbf{X}\mathbf{W}^*$ (DD3).
Description	This model defines the deterministic prediction step used before computing evaluation metrics.
Sources	https://doi.org/10.1371/journal.pcbi.1013244
Ref. By	IM7

Number	IM7
Label	**Cross-validated encoding score for a predictor**
Input	Predictor definition, MEG responses $\mathbf{Y}$, cross-validation procedure ρ_{CV} (DD6), lag window parameters, and score definition ρ (DD5).
Output	Cross-validated score ρ_{CV} for the predictor, computed as the mean of fold-wise scores $\rho^{(j)}$ over J folds.
Description	For each fold j, compute aligned predictors (IM3), construct $\mathbf{X}$ (IM4), fit weights on training data (IM5), predict held-out responses (IM6), then compute $\rho^{(j)}$ as correlation between $\hat{\mathbf{Y}}$ and $\mathbf{Y}$ on held-out data. Using an identical CV protocol and lag window across predictors enforces fair evaluation.
Sources	https://doi.org/10.7554/eLife.85012
Ref. By	None

4.2.6 Input Data Constraints

Table 4 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

Table 4: Input Variables

Var	**Physical Const.**	**Software Const.**	**Typical Value**	**Uncert.**
f_s	$f_s > 0$	$10 \leq f_s \leq 2000$ Hz	100 Hz	low
f_a	$f_a > 0$	$8000 \leq f_a \leq 96000$ Hz	16000 Hz	low
N	$N \in \mathbb{N},\ N > 0$	$N \geq N_{\min}$	dataset-dependent	medium
C	$C \in \mathbb{N},\ C > 0$	$1 \leq C \leq C_{\max}$	dataset-dependent	low
D	$D \in \mathbb{N},\ D > 0$	$1 \leq D \leq D_{\max}$	dataset-dependent	low
L	$L \in \mathbb{N},\ L > 0$	$1 \leq L \leq L_{\max}$	3–10	low
$\tau_{\min}$	$\tau_{\min} < \tau_{\max}$	$-0.5 \leq \tau_{\min} \leq 0$ s	-0.1 s	low

$\tau_{\max}$	$\tau_{\max} > \tau_{\min}$	$0.05 \leq \tau_{\max} \leq 2.0$ s	1.0 s	low
K	$K \in \mathbb{N},\ K > 0$	$K = \left\lfloor \frac{\tau_{\max} - \tau_{\min}}{\Delta t} \right\rfloor + 1$	implied by $f_s, \tau_{\min}, \tau_{\max}$	low
λ	$\lambda \geq 0$	$\lambda \in [\lambda_{\min}, \lambda_{\max}]$	selected by CV	medium
ρ_{CV}	defined partitioning	$2 \leq J \leq 20$ folds/partitions	$J = 4$	low

4.2.7 Properties of a Correct Solution

A correct solution must exhibit:

- **Time-base consistency:** Any predictor and neural response used together must be defined on the same sampling grid after alignment.
- **Protocol invariance across predictors:** When comparing multiple predictors, the lag window, lag discretization, cross-validation procedure, and regularization selection over λ must be identical across predictors so that differences in scores reflect predictor information rather than evaluation artifacts.
- **Numerical validity:** All reported fitted weights, predictions, and evaluation scores must be well-defined for every evaluated partition.
- **Traceability of results:** Each reported score must be uniquely attributable to a specific predictor definition, lag window, CV partitioning, and regularization setting, so that the evaluation can be repeated under the same specification.

Table 5: Output Variables

Var	Physical Constraints
ρ	$-1 \leq \rho \leq 1$ (by DD5)
ρ_{CV}	$-1 \leq \rho_{\text{CV}} \leq 1$ (by DD6)
$\Delta\rho$	$-2 \leq \Delta\rho \leq 2$

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: The system should accept speech stimuli and MEG signals as inputs.

R2: The system shall validate that the provided inputs are internally consistent before model fitting.

R3: Given a predictor definition, the system shall construct a time-aligned predictor matrix on the MEG sampling grid.

R4: The system shall evaluate all candidate predictors under a consistent protocol.

R5: For each predictor, the system shall compute an encoding score.

R6: The system shall compute baseline-referenced improvement for each candidate predictor relative to a specified acoustic baseline predictor.

5.2 Nonfunctional Requirements

NFR1: **Usability** Model–to–Brain Alignment for Speech Recognition shall support intended users (researchers and students with experience in scientific Python and neural data workflows.

NFR2: **Maintainability** The effort required to implement any of the likely changes shall be less than 0.5 of the original development effort.

NFR3: **Portability** Model–to–Brain Alignment for Speech Recognition shall run on modern Windows, macOS, and Linux systems. The procedures in the Verification and Validation Plan shall be executable on all supported operating environments.

NFR4: **Accuracy** The accuracy of results produced by Model–to–Brain Alignment for Speech Recognition shall meet the level required for scientific comparison of predictors in neural speech encoding.

5.3 Rationale

The requirements emphasize protocol consistency and traceability because the primary scientific objective is to compare predictors under controlled conditions (GS**??**, GS**??**). In particular, the choice to formalize the workflow around time alignment, shared lag windows, and cross-validation ensures that differences in ρ_{CV} and $\Delta\rho$ reflect differences in predictor information content rather than differences in evaluation settings. Typical parameter ranges in Section 4.2.6 are conservative to avoid unstable regression fits and to support practical runtimes while still permitting exploratory experiments.

6 Likely Changes

LC1: Support more deep learning-based speech models.

LC2: Support more biological inspired models.

LC3: Support BIDS data.

LC4: Support convect audio data to gammatone filterbank.

7 Unlikely Changes

LC5: The format of input experiment configuration will not change.

LC6: The input dataset format for raw data will not change.

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 6 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 7 shows the dependencies of instance models, requirements, and data constraints on each other. Table 8 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

References

Christian Brodbeck, Proloy Das, Marlies Gillis, Joshua P Kulasingham, Shohini Bhattasali, Phoebe Gaston, Philip Resnik, and Jonathan Z Simon. Eelbrain, a python toolkit for time-continuous analysis with temporal response functions. *Elife*, 12:e85012, 2023.

W. Spencer Smith and Nirmitha Koothoor. A document-driven method for certifying scientific computing software for use in nuclear safety analysis. *Nuclear Engineering and Technology*, 48(2):404–418, April 2016. ISSN 1738-5733. doi: http://dx.doi.org/10.1016/j.net.2015.11.008. URL http://www.sciencedirect.com/science/article/pii/S1738573315002582.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ågerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages

	TM??	TM4.2.2	DD1	DD2	DD3	DD4	DD5	DD6
TM??								
TM4.2.2								
DD1								
DD2			X					
DD3			X	X				
DD4			X	X	X			
DD5								
DD6						X	X	
IM3		X	X					
IM4		X	X	X				
IM5					X	X		
IM6					X			
IM7					X	X	X	X

Table 6: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM3	IM4	IM5	IM6	IM7	4.2.6
R1	X					X
R2	X	X				X
R3	X	X				X
R5				X	X	
R6					X	

Table 7: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??
TM??				X
TM4.2.2	X			
DD1	X			
DD2	X	X		
DD3	X	X	X	
DD4	X	X	X	
DD5	X			
DD6	X			
IM3	X			
IM4	X	X		
IM5	X	X	X	
IM6	X	X	X	
IM7	X	X	X	

Table 8: Traceability Matrix Showing the Connections Between Assumptions and Other Items

107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

W. Spencer Smith, Lei Lai, and Ridha Khedri. Requirements analysis for engineering computation: A systematic approach for improving software reliability. *Reliable Computing, Special Issue on Reliable Engineering Computation*, 13(1):83–107, February 2007.