# NS-Copilot: An LLM-Driven Agent System for Autonomous Neuroscience Analysis

**Wuche Liu[1], Yiran Qiao[1], Linlin Hou[1], Rui Yang[1], Shusen Pu[2], Song Wang[3], Jing Ma[1]**
[1]Department of Computer and Data Sciences, Case Western Reserve University, Cleveland, OH 44106, USA
[2]Department of Mathematics and Statistics, University of West Florida, Pensacola, FL 32514, USA
[3]Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA
**Correspondence:** wxl713@case.edu, jing.ma5@case.edu

## Abstract

AI is rapidly advancing neuroscience, yet many laboratories fail to fully unleash its potential due to significant interdisciplinary barriers. While pre-trained neural models for physiological data are progressing quickly, their heterogeneous architectures and modality-specific constraints hinder systematic integration, selection, and evaluation. Despite recent advances in large language model (LLM)-based agent systems for intelligent scientific applications, existing approaches often still lack the domain expertise required to effectively select and coordinate diverse neuroscience pre-trained models and handle unique data types in this domain. We present NS-Copilot, an LLM-driven multi-agent system for neuroscience analysis that autonomously supports end-to-end workflows for diverse professional tasks. It unifies domain-specific pre-trained models and supports key neuroscience modalities, including EEG and extracellular spike data, through a natural-language interface. Given raw data and a task description, NS-Copilot orchestrates agents with specialized roles for planning, adaptive control, code generation, and result synthesis, enabling analysis without dataset-specific heuristics. We evaluate NS-Copilot on neuroscience benchmarks spanning Alzheimer's disease, Parkinson's disease, and working memory spike decoding. Across 8 trials per task, the system consistently outperforms strong baselines on the primary metric, demonstrating the ability of NS-Copilot for effective and scalable neuroscience analysis.

## 1 Introduction

Although AI is driving rapid progress in neuroscience, substantial interdisciplinary barriers still limit the field's ability to fully realize its potential. Pre-trained neural models for physiological data have advanced quickly (Jiang et al., 2024; Ouahidi et al., 2025; Ye et al., 2025; Azabou et al., 2023), but their heterogeneous architectures and modality-specific constraints make systematic integration, model selection, and rigorous evaluation challenging (Xiong et al., 2026; Kastrati et al., 2025). Meanwhile, despite recent progress in both scientific and general-purpose LLM-based agent systems (Shen et al., 2023; Huang et al., 2026), most primarily rely on code generation or generic data analysis and machine learning toolkits which are often ill-suited for domain-specific applications, and lack the domain expertise needed to effectively select and coordinate diverse pre-trained neuroscience models and handle highly specialized data types. This gap highlights the need for an **autonomous system that can support end-to-end neuroscience analysis workflows.**

Despite its importance, this problem still presents three core challenges. **First, bridging the interface gap between raw data and model-specific input requirements.** Many pre-trained neuroscience models expect data in a highly specific and domain-dependent format, which includes particular sampling rates, channel layouts, and tensor shapes. This strict requirement reflects the deep architectural heterogeneity across the ecosystem. Preparing raw recordings to satisfy these stringent constraints requires an extremely deep understanding of both the underlying dataset structure and the target model's preprocessing pipeline, posing a prohibitive barrier for non-expert users. **Second, selecting the optimal method or model for a specific neuroscience dataset and task.** The ideal choice depends on a complex interplay of multiple factors: data modality (e.g., EEG vs. extracellular spikes), channel configuration, recording paradigm, and specific classification objectives. Since no single pre-trained architecture dominates across all conditions, the choice must be made per dataset. General-purpose AutoML systems (Erickson et al., 2020) do not search over domain-specific pre-trained models, and EEG benchmarks (Xiong et al., 2026) standardize evaluation but require man-



*Findings of the Association for Computational Linguistics: EMNLP 2026*, pages 15809–15828
October 24-29, 2026 ©2026 Association for Computational Linguistics

ual per-model configuration. **Third, adaptively optimizing the analytical workflow based on execution outcomes.** Even when a suitable model is selected and the data is correctly formatted, the generated pipeline may yield suboptimal performance due to misaligned hyperparameters, ineffective feature engineering, or flawed evaluation strategies. Recovering from such failures requires conducting a diagnosis of the root cause and determining an optimization strategy to adaptively adjust the workflow, going beyond the code-level error correction addressed by existing self-repair methods (Chen et al., 2024; Shinn et al., 2023).

We propose **NS-Copilot**, an autonomous LLM-driven agent system for neuroscience analysis, which addresses the three aforementioned challenges through a unified pipeline, requiring only raw neural data and a natural-language task description. NS-Copilot integrates multiple pre-trained neuroscience models in different applications. To bridge the interface gap, a Coder agent automatically generates data preprocessing and model invocation code at runtime based on declarative tool specifications (which describe the input requirements of each model) inside our system. To resolve model selection, a Planner agent automatically analyzes user input and dataset metadata (modality, channel count, sampling rate, and class distribution), determines appropriate evaluation metrics, and constructs a dynamically prioritized queue of compatible models for selection. To achieve adaptive optimization, a Controller agent analyzes execution metrics, diagnoses the root causes of suboptimal results, and dynamically routes the system toward targeted code patching (*modify*) or full strategy resynthesis by the Planner (*replan*). In the end, an Interpreter summarizes the final evaluation results.

In summary, our core contributions are as follows:

- **Unified pre-trained model orchestration framework.** We propose NS-Copilot, a novel LLM-driven agent system that first integrates nine representative pre-trained models across different neuroscience data modalities and tasks, achieving fully automated neuroscience analysis without dataset-specific heuristics.
- **Automated model selection.** We design the Auto Model Selection protocol, in which the Planner in NS-Copilot automatically analyzes dataset characteristics and user intention to determine the primary evaluation metric and construct a prioritized queue of compatible models, with early stopping triggered once a candidate surpasses a performance threshold.
- **Closed-loop diagnosis with explicit modify-vs-replan routing.** We design an autonomous Controller that diagnoses suboptimal execution outcomes and explicitly routes between targeted code patching (*modify*) and full strategy resynthesis (*replan*), enabling iterative optimization without human intervention.
- **Empirical validation across diverse neural paradigms.** We evaluate NS-Copilot on three benchmarks encompassing Alzheimer's disease, Parkinson's disease, and working memory decoding, where it outperforms every baseline on the primary metric of each.

## 2 Related Work

**Neural foundation models for brain data.** Pre-trained foundation models for neural recording data can be broadly categorized by their target modalities. For macroscopic EEG signals, a rapidly growing family of self-supervised models aims to learn generalizable representations from large-scale clinical and research corpora. Representative works include LaBraM (Jiang et al., 2024), CBraMod (Wang et al., 2025a), REVE (Ouahidi et al., 2025), EEGPT (Wang et al., 2024), EEGMamba (Wang et al., 2025b), and BrainOmni (Xiao et al., 2025), with architectures ranging from Transformers to state-space models, and pretraining scales from 2,500 hours to over 60,000 hours. Models such as NeuroLM (Jiang et al., 2025a), BIOT (Yang et al., 2023), and Neuro-GPT (Cui et al., 2024) further expand this ecosystem. For extracellular spike recordings, models like NDT3 (Ye et al., 2025), MtM (Zhang et al., 2024), and POYO-1 (Azabou et al., 2023) target multi-electrode array data, employing autoregressive prediction, multi-task masking, and cross-attention mechanisms, respectively, to learn transferable neural dynamics across sessions and brain regions. Notably, NDT3 further demonstrates cross-species generalization spanning non-human primates and human participants. Despite this rapid progress, the ecosystem remains highly fragmented. Each model possesses distinct input constraints, including specific channel montages, sampling rates, and data formats, making cross-model comparisons laborious and

error-prone. Recent benchmarking efforts, such as EEG-FM-Bench (Xiong et al., 2026) and EEG-Bench (Kastrati et al., 2025), have begun standardizing the evaluation of EEG models; however, these benchmarks require manual configuration on a per-model basis and do not cover the spike modality. Crucially, to our knowledge, no existing framework provides a unified interface to automatically select, invoke, and evaluate these heterogeneous models based on the characteristics of unseen datasets.

**Scientific agents.** Several systems have attempted to utilize LLMs as agents for scientific data analysis. EEGAgent (Zhao et al., 2025) uses LLMs to orchestrate EEG analysis tools but does not generate or execute code within a closed loop. NeuroWeaver (Wang et al., 2026) formalizes EEG pipeline design as an evolutionary code search problem with self-debugging, but it only targets a single modality and does not invoke pre-trained foundation models. SciDataCopilot (Rao et al., 2026) provides an agentic data preparation framework for EEG and MEG workflows, featuring code generation and execution capabilities, but its tool space contains only preprocessing routines rather than foundation models. Biomni (Huang et al., 2026) is a general-purpose biomedical agent that integrates over 150 tools and supports retrieval-augmented planning and code execution, but it is oriented toward broad biomedical tasks rather than neural recording analysis. NOBEL (Tang et al., 2026) unifies the representations of EEG, MEG, and fMRI within an LLM backbone, but it exists as a single end-to-end model rather than an orchestrating agent for multiple independent foundation models. Concurrent systems SpikeAgent (Lin et al., 2025) and BCI-Agent (Marin-Llobet et al., 2025) apply LLM/VLM agents to spike sorting and cell-type classification, respectively, both with narrower scope. None of the aforementioned systems integrate multiple pre-trained neural foundation models in the neuroscience domain into a unified orchestration pipeline to achieve automated model selection, code generation, execution, and evaluation.

**General agents.** In the domain of general agents, AutoML systems such as AutoGluon (Erickson et al., 2020), FLAML (Wang et al., 2021), and Auto-sklearn (Feurer et al., 2022) automate model selection and hyperparameter tuning for traditional machine learning models, but their search spaces do not include domain-specific pre-trained foundation models. HuggingGPT (Shen et al., 2023) and TaskMatrix.AI (Liang et al., 2023) demonstrate LLM-driven selection of pre-trained expert models, yet their selection relies on task semantics and model descriptions rather than the characteristics of the input data. Regarding code self-repair, Self-Debugging (Chen et al., 2024) teaches LLMs to debug their own code through execution feedback, Reflexion (Shinn et al., 2023) accumulates experience across attempts using language reinforcement learning, and Self-Refine (Madaan et al., 2023) iteratively generates feedback and revisions without requiring additional training. These methods primarily address code-level errors (syntax and runtime exceptions). Our Controller extends this paradigm to the analysis of execution metrics, diagnoses the root causes of poor results, and explicitly chooses between targeted code patching and full replanning. Such modify-vs-replan routing has been explored in general ML-engineering agents (Jiang et al., 2025b; Sun et al., 2023); to our knowledge, however, no LLM agent for closed-loop neuroscience data analysis adopts this routing as its Controller's core mechanism.

## 3 Method

### 3.1 Overview

NS-Copilot is a fully automated multi-agent framework that enables users to complete end-to-end neuroscience analyses by providing only raw neural data and a natural-language task description. As illustrated in Figure 1, the system is collaboratively driven by four LLM agents (*Planner*, *Coder*, *Controller*, and *Interpreter*). Upon initiation, the *Planner* analyzes the input to determine whether suitable pre-trained models are available; if not, it generates code directly. It then constructs a prioritized queue of pre-trained models based on dataset characteristics and the user's intent. The user may optionally specify a desired *performance threshold*, which the system uses as a stopping criterion to trigger early termination once the target is achieved. If not provided, the system automatically determines an appropriate threshold. Subsequently, the system sequentially runs each model within the prioritized queue, and early stopping is triggered the moment the performance threshold is exceeded; however, if all models fail to surpass the threshold, the *Controller* intervenes to diagnose and optimize the models sequentially. Ultimately, the *Interpreter* generates a comprehensive report containing metric interpretations and recommendations.

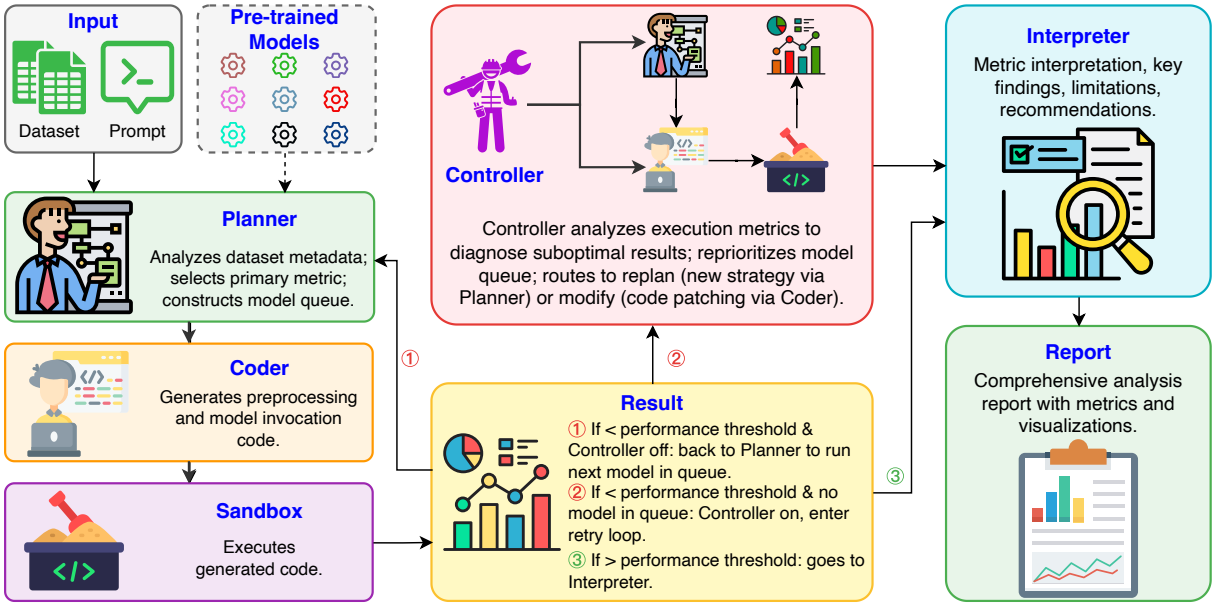


Figure 1: Overview of the NS-Copilot system. **Left column:** The user provides a dataset and a natural language prompt for task description. A registry of pre-trained models is available to the Planner via declarative tool specifications in the system. The Planner adaptively selects a primary metric, constructs a prioritized model queue, and generates an analysis plan. The Coder then synthesizes code for data preprocessing and model invocation, which is executed within a sandboxed environment. **Center column:** The *Result* module compares execution outcomes against the performance threshold. If the current model fails to exceed this threshold and alternative models remain in the queue, the system proceeds to evaluate the next model (①). If the entire queue is exhausted without surpassing the threshold, the Controller activates a closed-loop retry mechanism (②). Here, it diagnoses suboptimal results, reprioritizes the model queue, and dynamically routes the execution to either a *Replan phase* (synthesizing a new strategy via the Planner) or a *Modify phase* (executing targeted code patches via the Coder). Conversely, if any result successfully exceeds the threshold, the pipeline immediately proceeds to the Interpreter (③). **Right column:** The Interpreter produces a final comprehensive report detailing metric interpretations, key findings, limitations, and recommendations.

### 3.2 Planner

The Planner serves as the orchestrating intelligence: it analyzes the structured dataset profile (e.g., modality, channel count, sampling rate, and class distribution), dynamically determines the optimal evaluation metric (e.g., utilizing balanced accuracy for imbalanced data and macro F1 for multi-class classification tasks), and constructs a priority queue of candidate models. This ranking is derived from model metadata (such as pre-trained modality, channel compatibility, and parameter scale) and is inferred entirely at runtime.

To satisfy users' performance requirements, the system incorporates a performance threshold as a target criterion. The core premise of the Auto Model Selection protocol in our system is that pre-trained models do not always outperform carefully tuned traditional methods. This performance threshold is also important to rigorously quantify the utility of pre-trained models. Users may specify this threshold directly; if none is provided, it is derived empirically: an independent LLM programming agent (GPT-5.2-Codex; OpenAI, 2025) receives the same inputs to produce a competitive reference score.

Notably, Planner does not have access to the performance threshold score. This architecture rules out metric selection bias from the Planner: the choice of primary metric cannot be tuned to a score the Planner never observes, and is determined from the dataset profile and the task description alone. For each candidate model in the queue, the Planner formulates a targeted analysis plan (in JSON format) based on dataset characteristics and model constraints, specifying the tools to be used, the analytical steps, and the parameter configurations.

### 3.3 Coder

The Coder receives the analysis plan generated by the Planner, along with the dataset summary, and automatically generates complete Python scripts based on the tool specification. This process in-

cludes reading data from raw files, preprocessing to satisfy the input requirements of the target model (e.g., resampling and channel selection), invoking the model's inference interface to extract features, constructing a downstream classifier, and evaluating performance. The generated code is executed in a sandboxed environment, and all evaluation metrics are automatically extracted.

NS-Copilot currently integrates nine domain-specific pre-trained neural models across two primary modalities, as shown in Table 1. These models encompass a wide range of target tasks, architectures, embedding dimensions (128 to 4096), and spatial constraints (ranging from a strict 10-20 montage to completely channel-agnostic layouts).

To manage this architectural diversity, all pre-trained models are abstracted behind a unified tool API. Each model is strictly exposed as a stateless forward-inference endpoint, responsible solely for extracting latent representations; all subsequent downstream logic (such as temporal pooling and classification) is dynamically synthesized by the Coder at runtime. Each endpoint is accompanied by a structured specification (tool specification) detailing its operational constraints, such as supported modalities, acceptable channel counts, and target sampling rates.

### 3.4 Controller

If no candidate exceeds the performance threshold at the conclusion of the initial search phase, the Controller agent initiates the closed-loop optimization phase. Unlike standard LLM self-repair paradigms (which primarily focus on syntax errors or runtime exceptions), our Controller performs deep semantic debugging. By analyzing empirical execution metrics, it diagnoses the root causes of suboptimal performance and dynamically routes between two retry strategies.

At the beginning of each retry round, the Controller dynamically rearranges the search queue, elevating the priority of models that demonstrate the highest potential for improvement relative to the threshold. For each model in the updated queue, the Controller receives the complete execution trace: the synthesized code, empirical metrics, the previous analysis plan, dataset metadata, and the original user prompt. Leveraging this rich context, it identifies subtle performance anomalies (such as accuracy approaching random-chance levels or severe recall imbalances across classes) and formulates highly specific, actionable recommendations.

The Controller translates the diagnostic findings into discrete routing decisions, selecting between two optimization pathways:

**Modify mode** is triggered when the high-level logic is sound but local algorithmic adjustments are required (such as replacing the classifier, tuning regularization, or adjusting the feature extraction window). In this mode, the Planner is bypassed entirely; the previous scripts and the Controller's targeted patching instructions are passed directly to the Coder.

**Replan mode** is invoked when empirical results indicate a fundamental flaw in the methodological approach (such as a completely inappropriate feature extraction paradigm). The diagnostic feedback is routed back to the Planner to synthesize an entirely new analysis strategy from scratch.

Consistent with the initial search phase, if the optimized model exceeds the performance threshold, early stopping is triggered immediately. Otherwise, the Controller proceeds to optimize the next model in the queue until the threshold is surpassed. If all models in the queue fail to exceed the threshold after a full retry round, the system enters the next retry round. To balance computational overhead and optimization efficacy, the framework sets a default limit of two retry rounds.

### 3.5 Interpreter

When the performance of a candidate model successfully exceeds the performance threshold (whether during the initial search or the Controller retry phase), the Interpreter is invoked to generate the final evaluation report. It receives the complete execution results, including the evaluation metrics (accuracy, balanced accuracy, macro F1, AUROC, and per-class recall), the confusion matrices, and the generated visualizations, to scientifically interpret the outcomes, summarize key findings, and propose recommendations for improvement. If no model exceeds the threshold after all retry rounds, the framework reports the best-performing model across all attempts along with its complete metrics.

## 4 Experiments

In this section, we present a comprehensive empirical evaluation of NS-Copilot across three distinct neuroscience benchmarks. Specifically, our experiments are designed to answer the following core research questions.

- **RQ1**: How does NS-Copilot perform com-

Table 1: Integrated pre-trained models. *Embed.* = dimension of the feature vector produced by each model's forward pass in our system (after any pooling or token concatenation; may differ from the model's internal hidden dimension); *Ch.* = channel constraint; *Hz* = target sampling rate; *Std. pos.* = standard electrode positions; *Downstream tasks* = representative evaluation tasks reported in the original paper. Models are grouped by modality: spike (top) and EEG (bottom).

| Model | Modality | Embed. | Ch. | Hz | Downstream Tasks |
|---|---|---|---|---|---|
| NDT3 (Ye et al., 2025) | Spike | 1024 | — | — | Motor decoding (cursor control, reaching, bimanual) |
| MtM (Zhang et al., 2024) | Spike | 512 | — | — | Co-smoothing, forward prediction, choice decoding |
| POYO-1 (Azabou et al., 2023) | Spike | 128 | — | — | Motor decoding (hand velocity, cross-session transfer) |
| LaBraM (Jiang et al., 2024) | EEG | 200 | 10-20 | 200 | Abnormal detection, event classification, emotion recognition |
| REVE (Ouahidi et al., 2025) | EEG | 1536 | Any | 200 | Motor imagery, sleep staging, emotion recognition, abnormal detection |
| CBraMod (Wang et al., 2025a) | EEG | 600 | Any | 200 | Emotion recognition, motor imagery, sleep staging, seizure detection |
| EEGPT (Wang et al., 2024) | EEG | 2048 | 10-10 | 256 | Motor imagery, ERP detection, sleep staging |
| EEGMamba (Wang et al., 2025b) | EEG | 600 | Any | 200 | Emotion recognition, motor imagery, sleep staging, seizure detection |
| BrainOmni (Xiao et al., 2025) | EEG | 4096 | Std. pos. | 256 | AD/PD classification, emotion recognition, motor imagery, abnormal detection |

pared with other agents and code generation baselines?

- **RQ2**: To what extent does the Controller's closed-loop optimization contribute to the overall system performance?
- **RQ3**: How do pre-trained models benefit the system?
- **RQ4**: What are the primary failure modes of the whole system?

### 4.1 Datasets

We evaluate NS-Copilot on three publicly available neural datasets: (1) **AD EEG** (Alzheimer's Disease; OpenNeuro ds004504 (Miltiadous et al., 2024)): resting-state EEG from 88 subjects, 19-channel 10-20 montage, 3-class classification (AD/FTD/Healthy); (2) **PD EEG** (Parkinson's Disease; OpenNeuro ds004584 (Singh et al., 2023)): resting-state EEG from 149 subjects, 64-channel 10-10 montage, binary classification (PD/Control); (3) **WM Spikes** (Working Memory; DANDI 000006 (Economo and Svoboda, 2022)): extracellular spike recordings during a delayed-response task, binary decoding of left vs. right lick decisions. We use one session of 79 units and the 256 trials that remain after applying the dataset's own quality flag and removing early-lick and no-response trials. These benchmarks span three distinct neuroscience scenarios (Alzheimer's disease classification, Parkinson's disease classification, and cognitive working memory decoding), two recording modalities, diverse channel configurations (19–64 channels), and both binary and multi-class tasks.

**Data leakage audit.** Every pre-trained model is used as a frozen feature extractor, so if a benchmark overlaps with a model's pre-training corpus, strong results may reflect prior exposure rather than genuine transferability. Considering this, we audited all nine models against the three accession identifiers, and also at the level of the underlying cohort, since one cohort can be released under several identifiers. None of the three is part of any model's pre-training corpus; Appendix B gives the per-model details. We note only one case of potential partial exposure. BrainOmni uses ds004504 (same as ours in AD) as a downstream benchmark of its own, so its released checkpoint was developed with visibility into results on that dataset. Its task there was a two-class subset of 65 of the 88

subjects; however, ours is three classes over all 88, and the backbone is frozen throughout our experiments: we do not fine-tune it on ds004504, so the exposure is limited to what the released checkpoint already carries.

### 4.2 Experimental Setup

For each benchmark, we conduct 8 independent trials, reporting the mean $\pm$ standard deviation across all metrics. Each trial represents a fully autonomous, end-to-end execution, spanning from raw data ingestion to final report generation without any manual intervention.

**Evaluation metrics.** The Planner dynamically determines the primary optimization metric based on dataset metadata: macro F1 for the 3-class AD task, and balanced accuracy for the binary PD and WM tasks. To provide a comprehensive evaluation, we additionally track accuracy, AUROC, and per-class recall across all experiments.

**Baselines.** We compare NS-Copilot against three categories of baselines: *general agents* (AutoML-Agent (Trirat et al., 2025), LAMBDA (Sun et al., 2026), AutoGen (Wu et al., 2023)), *scientific agents* (SciAgents (Ghafarollahi and Buehler, 2024)), and *code generation baselines* (Claude-Sonnet-4.5 (Anthropic, 2025), GPT-5.2-Codex (OpenAI, 2025)). The Codex score establishes the performance threshold required to trigger early stopping.

**Execution and computational budget.** All synthesized scripts are executed within a sandboxed environment inside a Docker container, with restricted imports, controlled builtins, and execution timeouts. A single script execution is allowed 300 s; a script that exceeds it is recorded as a failed attempt and contributes a score of zero to that seed. This budget is uniform across models, so it penalizes the models that need the most compute per trial, and Appendix E shows that one model, BrainOmni, is affected badly enough to distort its reported result. If the initial search phase fails to yield a model that exceeds the baseline threshold, the Controller's closed-loop optimization is strictly constrained to a default budget of two retry iterations before termination.

### 4.3 RQ1: Overall Performance

Table 2 summarizes the main results across all three benchmarks; the complete per-model, per-stage results for AD and PD are given in Appendix D. NS-Copilot outperforms the state-of-the-art baselines on the primary metric of all three datasets and on eight of the nine reported predictive metrics (the exception is AUROC on WM), demonstrating that our agent system yields meaningful gains through automatic planning and domain-specific pre-trained model integration.[1] While NS-Copilot exhibits a higher execution time due to multiple model trials and the overhead of running pre-trained models, this temporal cost is well justified by the resulting performance improvements and the complete elimination of manual, human-in-the-loop pipeline engineering. A complementary evaluation on a held-out test fold, run with queue search, early stopping and the Controller all switched off, is reported in Appendix C.

### 4.4 RQ2: Effect of the Controller

To evaluate the contribution of the Controller module, we measure the performance of our system across three distinct stages: (1) **Initial Run**, where the Planner generates an analysis plan and the Coder produces executable code with up to three self-correction attempts for runtime errors, yet without Controller intervention; (2) **+ Retry 1**, where the Controller diagnoses the Initial result by identifying anomalies and issuing targeted recommendations, subsequently triggering a full replan-or-modify cycle; and (3) **+ Retry 2**, where the Controller diagnoses the execution once more to trigger a second optimization cycle. We run this ablation on AD, over Vanilla and five pre-trained models (BrainOmni is excluded; see the caption), with the same 8 independent seeds for each configuration (48 runs). Table 3 reports the change in Macro F1 relative to the Initial Run for the forced Retry 2 round and for the value the system returns, together with the seed-to-seed standard deviation of Retry 2 and of the returned value.

Table 3 separates two things the Controller does. The forced exploration is close to neutral on AD: two retry rounds move the mean by $-0.11$ pp and improve as many runs as they degrade (20 against 20 of 48). What the system returns, the best of the three rounds for each seed, is $+3.80$ pp above the initial round and is worse than it on none of the 48 runs. A retry round is a fresh point in the configuration space rather than a refinement of the previous one: Modify mode substitutes the classi-

[1] Below-chance baselines stem from method-specific failures (Appendix A).

Table 2: Performance comparison with baselines (mean ± std over 8 trials). **Primary Metric** auto-selected by the Planner: *Macro F1* (AD), *Balanced Accuracy* (PD/WM). *Eff.* = avg execution time (s). Best in **bold**. NS-Copilot uses *online* early-stopping; per-seed winners: AD = REVE/BrainOmni/Vanilla, PD = Vanilla/LaBraM, WM = Vanilla/POYO-1.

| Dataset | Method | Primary Metric (%) | Acc. (%) | AUROC (%) | Eff. (s) |
|---|---|---|---|---|---|
| AD EEG (Macro F1↑) | AutoML-Agent (Trirat et al., 2025) | $5.55 \pm 15.71$ | $6.95 \pm 19.64$ | $9.38 \pm 26.52$ | 255 |
| | LAMBDA (Sun et al., 2026) | $43.57 \pm 38.28$ | $45.37 \pm 39.33$ | $54.99 \pm 45.83$ | 100 |
| | AutoGen (Wu et al., 2023) | $26.75 \pm 11.70$ | $32.75 \pm 10.59$ | $44.25 \pm 19.95$ | 103 |
| | Claude-Sonnet-4.5 | $48.70 \pm 6.58$ | $51.00 \pm 7.18$ | $68.47 \pm 3.27$ | 304 |
| | Codex | $51.01 \pm 2.52$ | $53.98 \pm 2.98$ | $69.81 \pm 1.15$ | 68 |
| | NS-Copilot | $\mathbf{57.16 \pm 5.71}$ | $\mathbf{60.57 \pm 4.32}$ | $\mathbf{75.09 \pm 4.83}$ | 425 |
| PD EEG (Bal. Acc.↑) | AutoML-Agent (Trirat et al., 2025) | $17.39 \pm 32.29$ | $18.33 \pm 33.95$ | $19.90 \pm 37.01$ | 293 |
| | LAMBDA (Sun et al., 2026) | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | 284 |
| | AutoGen (Wu et al., 2023) | $6.25 \pm 17.68$ | $8.38 \pm 23.69$ | $6.25 \pm 17.68$ | 308 |
| | Claude-Sonnet-4.5 | $65.12 \pm 3.27$ | $70.99 \pm 1.54$ | $74.07 \pm 3.52$ | 516 |
| | Codex | $65.52 \pm 7.98$ | $64.95 \pm 12.87$ | $71.07 \pm 8.23$ | 125 |
| | NS-Copilot | $\mathbf{71.32 \pm 2.97}$ | $\mathbf{71.91 \pm 1.87}$ | $\mathbf{78.15 \pm 1.21}$ | 975 |
| WM Spikes (Bal. Acc.↑) | AutoML-Agent (Trirat et al., 2025) | $18.88 \pm 26.05$ | $20.62 \pm 28.48$ | $19.00 \pm 26.23$ | 196 |
| | LAMBDA (Sun et al., 2026) | $70.62 \pm 3.11$ | $72.00 \pm 3.66$ | $78.50 \pm 4.38$ | 66 |
| | AutoGen (Wu et al., 2023) | $55.00 \pm 0.00$ | $56.00 \pm 0.00$ | $59.00 \pm 0.00$ | 20 |
| | SciAgents (Ghafarollahi and Buehler, 2024) | $50.00 \pm 0.00$ | $51.00 \pm 0.00$ | $59.00 \pm 0.00$ | 40 |
| | Claude-Sonnet-4.5 | $73.97 \pm 6.99$ | $74.76 \pm 7.05$ | $\mathbf{83.89 \pm 6.70}$ | 164 |
| | Codex | $72.58 \pm 1.68$ | $72.77 \pm 1.94$ | $78.87 \pm 2.23$ | 97 |
| | NS-Copilot | $\mathbf{74.43 \pm 3.18}$ | $\mathbf{74.81 \pm 2.93}$ | $80.93 \pm 4.22$ | 470 |

Table 3: Ablation of the Controller on AD, over all six pre-trained and vanilla configurations and eight seeds each (48 runs). Δ *(exploration)* is Retry 2 minus the initial round: what two forced rounds change on their own. Δ *(returned)* is what the system reports, the best of the three rounds for each seed, minus the initial round; it cannot be negative by construction, so the quantity of interest is its size, not its sign. The last two columns give the seed-to-seed standard deviation of Retry 2 and of the returned value. **Bold** marks the three configurations where selecting the best round left the output less dispersed than Retry 2. BrainOmni is excluded: its AD figures are distorted by execution timeouts (Appendix E).

| Configuration | Δ (exploration) | Δ (returned) | Retry 2 (std) | Returned (std) |
|---|---|---|---|---|
| Vanilla | −2.32 | +2.54 | 9.53 | **4.97** |
| LaBraM | −2.18 | +3.50 | 3.79 | 6.71 |
| REVE | −2.91 | +0.65 | 5.24 | **3.40** |
| CBraMod | +3.48 | +7.38 | 6.91 | 9.12 |
| EEGPT | −2.36 | +1.22 | 11.00 | **3.65** |
| EEGMamba | +5.62 | +7.51 | 6.53 | 6.79 |
| Mean | −0.11 | +3.80 | | |
| Runs improved / degraded (rest unchanged) | 20 / 20 | 25 / 0 | | |

fier, the regularization or the feature window, and Replan mode discards the previous plan and synthesizes a new strategy from scratch. The Controller routes between the two on the evidence of a single score, and when the shortfall is small that score carries more noise than signal; nothing requires the new result to beat the old one before it is recorded. A round that lands on a worse point is therefore an expected cost of exploring. The Controller's contribution lies in retaining the best round rather than in the exploration itself, and the two Δ columns are designed to disentangle these effects. This also explains the standard deviations. They grow across the forced rounds because exploration is genuinely erratic: for each of the six configurations, the largest per-stage standard deviation of Macro F1 in Table 5 falls in Retry 1 or Retry 2 rather than in the initial round. One EEGPT seed collapses in Retry 2, which alone lifts that round's standard deviation to 11.00. The selection rule discards that round, and the returned standard deviation is 3.65. The dispersion of the forced rounds describes the search, not the output. The forced exploration is more clearly harmful on PD, where

Table 6 puts Retry 2 an average of $-2.34$ pp in Balanced Accuracy below the initial round over the same six configurations.

### 4.5 RQ3: Effect of Pre-trained Models

To quantify the benefit of pre-trained neural models, we compare the vanilla pipeline (handcrafted spike-count features) against POYO-1 on the WM task, with the model fixed in advance and both arms scored on the same held-out partition for each of the 8 seeds. POYO-1 yields a consistent gain of $+6.82$ pp in Balanced Accuracy on the held-out evaluation ($75.30 \pm 8.39$ vs. $68.48 \pm 7.49$), favoring the pre-trained model on 7 of 8 seeds (Table 4). This improvement is consistent with POYO-1's PerceiverIO architecture, which is designed for variable neuron counts; our comparison is end-to-end and does not isolate that property. These results indicate that pre-trained models can provide meaningful representational advantages over classical features, and that the size of the advantage varies by task, from $+13.34$ pp on AD to a near-tie on PD ($+0.68$; Table 4). It is that variation which motivates the automated model selection protocol at the core of NS-Copilot. It is worth being explicit about where the advantage comes from. Codex and Claude-Sonnet-4.5 in Table 2 are pure LLM agents: they receive the same task description and write their own analysis code, with no pre-trained neural model available. The *Vanilla* configuration is our architecture with those models removed, and on the two benchmarks with full per-stage results its initial round scores *below* Codex (49.80 vs. 51.01 on AD and 64.52 vs. 65.52 on PD; Tables 5 and 6). Orchestration on its own therefore does not beat a single-shot LLM agent. What the comparison isolates is the integration itself: making nine pre-trained models addressable through one interface, and choosing among them per task, is what produces the gain. The held-out evaluation of Appendix C is also what separates that gain from adaptive selection: there the queue search, the early stopping and the Controller retry loop are all switched off and the model is fixed in advance, so no part of the reported advantage can come from choosing the winner after the fact.

### 4.6 RQ4: Error Analysis

To systematically characterize the failure modes of our system, we analyze 325 execution attempts across the three benchmarks; a script that exceeds the 300 s budget is recorded as a failed attempt in the reported scores and is examined in Appendix E rather than counted here. The system achieves a 92.3% code-level success rate, leaving 25 code-level failures distributed across two primary categories (Figure 2). Code generation errors (68.0%) arise when the LLM produces scripts containing incorrect variable references or label format mismatches. Sandbox module restrictions (32.0%) occur when the generated code attempts to import libraries not permitted within the isolated execution environment. Both failure modes are non-catastrophic: they surface as explicit execution errors rather than as silently incorrect results, and a failure costs a candidate slot rather than the run, since the system continues through the prioritized queue and reports the best model across all attempts.

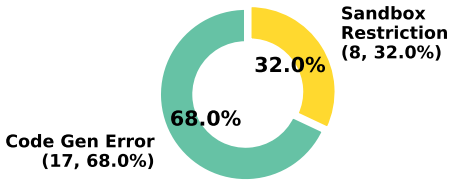


Figure 2: Error distribution: 25 code-level failures out of 325 attempts (92.3% code-level success rate).

## 5 Conclusion

We propose NS-Copilot, an autonomous LLM-driven agent system that unifies nine pre-trained models across EEG and spike modalities behind a single natural-language interface, integrating automated model selection, code generation, and closed-loop optimization. Across three benchmarks, NS-Copilot outperforms every baseline on the primary metric; ablations show that the gain comes from the pre-trained models rather than from orchestration alone, that the Controller's value lies in keeping the best round rather than in the forced exploration, and that the optimal model varies across tasks. By lowering the barrier to advanced neuroscience analysis, NS-Copilot enables researchers without extensive AI expertise to leverage state-of-the-art models and accelerate data-driven discovery.

## Limitations

Current limitations include increased execution time and computational cost due to closed-loop evaluation, which is necessary to reliably explore candidate models and select the optimal method for each task. Instrumented per-agent token measurements indicate that NS-Copilot's per-run cost is roughly 1.5–2× that of the single-step Codex baseline; this overhead is justified by the performance gains over Codex. Each of the three scenarios is represented by a single dataset. Adding further datasets within each scenario, so that the same protocol meets greater within-modality variation, is the direction in which we intend to grow the benchmark. Future work will extend the framework to incorporate additional neuroscience tools, add a dedicated validator agent that checks label alignment and output shapes, support new modalities such as fMRI and MEG, explore efficient orchestration strategies, and enable more complex, multi-step analysis workflows.

## Ethics Statement

**Use of Large Language Models.** This work uses LLMs as a core component of the proposed system: an LLM serves as the orchestration agent that selects pre-trained models, generates analysis code, and drives iterative optimization. Additionally, Claude (Anthropic) was used to assist with code development and paper grammar-checking during this research. LLMs were not used to generate any original research ideas or any original content. All LLM-generated content was reviewed and verified by the authors.

**Datasets.** All datasets used in this work are publicly available under open licenses. The Alzheimer's/FTD EEG dataset (OpenNeuro ds004504) is released under CC0 (Public Domain). The Parkinson's disease EEG dataset (OpenNeuro ds004584) is released under CC0 (Public Domain). The working memory spike dataset (DANDI 000006) is released under CC-BY 4.0. No private or restricted-access data was used. All datasets were collected by their original authors with appropriate ethical approvals as described in their respective publications.

**Pre-trained Models.** The nine integrated pre-trained models are publicly available for research use. All model code is released under permissive open-source licenses (MIT or Apache 2.0). Model weights are distributed by their respective authors via GitHub, Hugging Face, or dedicated servers under varying terms: some under permissive licenses (MIT or Apache 2.0), some under non-commercial licenses (CC-BY-NC 4.0), and some with custom access agreements. REVE weights require accepting a gated access agreement on Hugging Face. Our framework does not redistribute any model weights; users download them directly from the original sources. We obtained access to all models and used them solely for non-commercial academic research in compliance with their respective terms.

Overall, this work does not raise any significant ethical concerns.

**Broader Impact.** Our framework is designed to lower the barrier for neuroscientists to leverage AI agents and domain-specific pre-trained models. While this democratization of access is broadly positive, we note that automated analysis pipelines should complement, not replace, domain expertise. Users should critically evaluate all outputs, particularly in clinical contexts where incorrect classifications could have downstream consequences. Overall, NS-Copilot represents a step toward more accessible, scalable, and trustworthy AI-driven neuroscience research.

## Reproducibility Statement

All source code, including the complete multi-agent pipeline, tool specifications, and experiment scripts, is provided as supplementary material. To ensure reproducibility, the system is fully containerized via Docker (including the Dockerfile and Docker Compose configuration), enabling a seamless, single-command environment setup; Appendix F shows the web interface it exposes. Experimental reproduction is facilitated through `run_test.sh`, which fixes the random seeds each run uses. Because the analysis code is written by the LLM agents at run time, a repeated run is not guaranteed to reproduce a given score exactly; every result we report is therefore a mean and standard deviation over eight seeds rather than a single figure. All datasets are publicly available: the AD EEG dataset (OpenNeuro ds004504, CC0), the PD EEG dataset (OpenNeuro ds004584, CC0), and the WM spike dataset (DANDI 000006, CC-BY 4.0). Furthermore, the weights for all nine pre-trained models are publicly accessible, and `setup.sh` downloads them automatically once the user has accepted the access terms set by the in-

dividual providers. The specific LLMs utilized in this study are GPT-5.2 (OpenAI, powering all four NS-Copilot agents), GPT-5.2-Codex (OpenAI, employed to establish the performance threshold), and Claude Sonnet 4.5 (Anthropic, serving as one of the baselines). Each agent baseline (AutoML-Agent, LAMBDA, AutoGen, SciAgents) was evaluated using its originally published default LLM configuration.

## References


Anthropic. 2025. Introducing Claude Sonnet 4.5.

Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael J. Mendelson, Blake Richards, Matthew G. Perich, Guillaume Lajoie, and Eva L. Dyer. 2023. A unified, scalable framework for neural population decoding. In *NeurIPS*.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. 2024. Teaching large language models to self-debug. In *ICLR*.

Wenhui Cui, Woojae Jeong, Philipp Thölke, Takfarinas Medani, Karim Jerbi, Anand A. Joshi, and Richard M. Leahy. 2024. Neuro-GPT: Towards a foundation model for EEG. In *IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 1–5.

Michael N. Economo and Karel Svoboda. 2022. Mouse anterior lateral motor cortex (ALM) in delay response task. DANDI Archive. Dandiset 000006, version 0.220126.1855.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. 2020. AutoGluon-Tabular: Robust and accurate AutoML for structured data. In *ICML AutoML Workshop*.

Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. 2022. Auto-sklearn 2.0: Hands-free AutoML via meta-learning. *JMLR*, 23(261).

Alireza Ghafarollahi and Markus J. Buehler. 2024. SciAgents: Automating scientific discovery through multi-agent intelligent graph reasoning. *Preprint*, arXiv:2409.05556.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Ryan Li, Yusuf Roohani, Lin Qiu, Shiyi Cao, Gavin Li, Junze Zhang, Di Yin, Rick Wierenga, Deniz Kavi, Sherry Liu, Tianwei She, Shruti Marwaha, Jennefer N. Carter, Xin Zhou, and 9 others. 2026. Autonomous biomedical research with an artificial intelligence agent. *Science*, 393(6813):eadz4351.

Wei-Bang Jiang, Yansen Wang, Bao-Liang Lu, and Dongsheng Li. 2025a. NeuroLM: A universal multi-task foundation model for bridging the gap between language and EEG signals. In *ICLR*.

Wei-Bang Jiang, Li-Ming Zhao, and Bao-Liang Lu. 2024. Large brain model for learning generic representations with tremendous EEG data in BCI. In *ICLR*.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. 2025b. AIDE: AI-driven exploration in the space of code. *Preprint*, arXiv:2502.13138.

Ard Kastrati, Josua Bürki, Jonas Lauer, Cheng Xuan, Raffaele Iaquinto, and Roger Wattenhofer. 2025. EEG-Bench: A benchmark for EEG foundation models in clinical applications. *Preprint*, arXiv:2512.08959.

Yaobo Liang, Chenfei Wu, Ting Song, Wenshan Wu, Yan Xia, Yu Liu, Yang Ou, Shuai Lu, Lei Ji, Shaoguang Mao, Yun Wang, Linjun Shou, Ming Gong, and Nan Duan. 2023. TaskMatrix.AI: Completing tasks by connecting foundation models with millions of APIs. *Preprint*, arXiv:2303.16434.

Zuwan Lin, Arnau Marin-Llobet, Jongmin Baek, Yichun He, Jaeyong Lee, Wenbo Wang, Xinhe Zhang, Ariel J. Lee, Ningyue Liang, Jin Du, Jie Ding, Na Li, and Jia Liu. 2025. Spike sorting AI agent. *bioRxiv*. Preprint.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *NeurIPS*.

Arnau Marin-Llobet, Zuwan Lin, Jongmin Baek, Almir Aljovic, Xinhe Zhang, Ariel J. Lee, Wenbo Wang, Jaeyong Lee, Hao Shen, Yichun He, Na Li, and Jia Liu. 2025. An AI agent for cell-type specific brain computer interfaces. *bioRxiv*. Preprint.

Andreas Miltiadous, Katerina D. Tzimourta, Theodora Afrantou, Panagiotis Ioannidis, Nikolaos Grigoriadis, Dimitrios G. Tsalikakis, Pantelis Angelidis, Markos G. Tsipouras, Evripidis Glavas, Nikolaos Giannakeas, and Alexandros T. Tzallas. 2024. A dataset of EEG recordings from: Alzheimer's disease, Frontotemporal dementia and Healthy subjects. OpenNeuro. Dataset ds004504, version 1.0.8.

OpenAI. 2025. Introducing GPT-5.2-Codex.

Yassine El Ouahidi, Jonathan Lys, Philipp Thölke, Nicolas Farrugia, Bastien Pasdeloup, Vincent Gripon, Karim Jerbi, and Giulia Lioi. 2025. REVE: A foundation model for EEG – adapting to any setup with large-scale pretraining on 25,000 subjects. In *NeurIPS*.

Jiyong Rao, Yicheng Qiu, Jiahui Zhang, Juntao Deng, Shangquan Sun, Fenghua Ling, Hao Chen, Nanqing Dong, Zhangyang Gao, Siqi Sun, Yuqiang Li, Dongzhan Zhou, Guangyu Wang, Lijun Wu, Conghui He, Xuhong Wang, Jing Shao, Xiang Liu, Yu Zhu, and 13 others. 2026. SciDataCopilot: An agentic data preparation framework for AGI-driven scientific discovery. *Preprint*, arXiv:2602.09132.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *NeurIPS*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*.

Arun Singh, Rachel Cole, Arturo Espinoza, Jim Cavanagh, and Nandakumar Narayanan. 2023. Rest eyes open. OpenNeuro. Dataset ds004584, version 1.0.0.

Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. 2023. AdaPlanner: Adaptive planning from feedback with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Maojun Sun, Ruijian Han, Binyan Jiang, Houduo Qi, Defeng Sun, Yancheng Yuan, and Jian Huang. 2026. LAMBDA: A large model based data agent. *Journal of the American Statistical Association*, 121(553):1–13.

Changli Tang, Shurui Li, Junliang Wang, Qinfan Xiao, Zhonghao Zhai, Lei Bai, Yu Qiao, Bowen Zhou, Wen Wu, Yuanning Li, and Chao Zhang. 2026. One brain, omni modalities: Towards unified non-invasive brain decoding with large language models. *Preprint*, arXiv:2602.21522.

Patara Trirat, Wonyong Jeong, and Sung Ju Hwang. 2025. AutoML-Agent: A multi-agent LLM framework for full-pipeline AutoML. In *ICML*.

Chi Wang, Qingyun Wu, Markus Weimer, and Erkang Zhu. 2021. FLAML: A fast and lightweight AutoML library. In *MLSys*, volume 3, pages 434–447.

Guangyu Wang, Wenchao Liu, Yuhong He, Cong Xu, Lin Ma, and Haifeng Li. 2024. EEGPT: Pretrained transformer for universal and reliable representation of EEG signals. In *NeurIPS*, volume 37, pages 39249–39280.

Guoan Wang, Shihao Yang, Jun-En Ding, and Feng Liu. 2026. NeuroWeaver: An autonomous evolutionary agent for exploring the programmatic space of EEG analysis pipelines. *Preprint*, arXiv:2602.13473.

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

Jiquan Wang, Sha Zhao, Zhiling Luo, Yangxuan Zhou, Shijian Li, and Gang Pan. 2025b. EEGMamba: An EEG foundation model with mamba. *Neural Networks*, 192:107816.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *Preprint*, arXiv:2308.08155.

Qinfan Xiao, Ziyun Cui, Chi Zhang, Siqi Chen, Wen Wu, Andrew Thwaites, Alexandra Woolgar, Bowen Zhou, and Chao Zhang. 2025. BrainOmni: A brain foundation model for unified EEG and MEG signals. In *NeurIPS*.

Wei Xiong, Jiangtong Li, Jie Li, Kun Zhu, and Changjun Jiang. 2026. EEG-FM-Bench: A comprehensive benchmark for the systematic evaluation and diagnostic analyses of EEG foundation models. In *ICML*.

Chaoqi Yang, M. Brandon Westover, and Jimeng Sun. 2023. BIOT: Biosignal transformer for cross-data learning in the wild. In *NeurIPS*, volume 36, pages 78240–78260.

Joel Ye, Fabio Rizzoglio, Adam Smoulder, Hongwei Mao, Xuan Ma, Patrick Marino, Raeed Chowdhury, Dalton Moore, Gary Blumenthal, William Hockeimer, Nicolas G. Kunigk, J. Patrick Mayo, Aaron P. Batista, Steven Chase, Adam Rouse, Michael L. Boninger, Charles Greenspon, Andrew B. Schwartz, Nicholas G. Hatsopoulos, and 5 others. 2025. A generalist intracortical motor decoder. In *NeurIPS*.

Yizi Zhang, Yanchen Wang, Donato Jimenez-Beneto, Zixuan Wang, Mehdi Azabou, Blake Richards, Renee Tung, Olivier Winter, The International Brain Laboratory, Eva Dyer, Liam Paninski, and Cole Hurwitz. 2024. Towards a "universal translator" for neural dynamics at single-cell, single-spike resolution. In *NeurIPS*.

Sha Zhao, Mingyi Peng, Haiteng Jiang, Tao Li, Shijian Li, and Gang Pan. 2025. EEGAgent: A unified framework for automated EEG analysis using large language models. *Preprint*, arXiv:2511.09947.

## A Baseline Failure Modes

Several baselines exhibit below-chance performance, or produce no usable result at all, on AD, PD and WM; we attribute these to method-specific failures rather than unfair comparison. AutoML-Agent's rigid Optuna-based search proved incompatible with our raw EEG inputs, causing pipeline crashes on 7/8 AD trials (returning 0). On the WM spike benchmark AutoML-Agent likewise failed to produce runnable code on 5 of 8 trials, which are scored as 0 and pull the mean below chance.

AutoGen consistently generated severely biased pipelines (e.g., 52% AD recall but 5% FTD recall on the 3-class task, indicating majority-class collapse). SciAgents, designed for hypothesis generation rather than supervised classification, produced a degenerate chance-level result on WM, the only benchmark on which we report it: its balanced accuracy is 50.00%, the chance level for a binary task, and its standard deviation is zero on all three reported metrics across the eight trials (Table 2). LAMBDA did not produce a parseable result on any of the PD trials we report: its sessions ended inside its own self-repair loop without printing final metrics, and such a trial contributes a score of zero under the same convention we apply to our own runs (Section 4.2). Importantly, these failures are themselves informative: they demonstrate that domain-aware orchestration, rather than a stronger generic LLM agent, is what enables consistent autonomous performance in this domain.

## B Data Leakage Audit

None of our three benchmarks (ds004504, ds004584, DANDI 000006) is part of the pre-training corpus of any of the nine integrated models. We matched on accession identifiers rather than dataset names, which are not unique.

Identifiers alone are not a sufficient audit, because a single cohort can be released under several of them, and the corpora themselves demonstrate this: BrainOmni pre-trains on the Healthy Brain Network cohort under both ds004186 and ds005505 to ds005512. We therefore also checked the alternative releases of our own cohorts, namely ds006036 for the AD cohort and ds004574, ds004579 and ds004580 from the laboratory that released ds004584. None appears in any corpus.

Two of the nine models draw on OpenNeuro, and both enumerate what they use: REVE lists 55 accession identifiers and BrainOmni 24, and neither list contains ours. The nearest is ds004582, which differs from ds004584 only in its final digit. Clinical cohorts of the same two diseases do occur in these corpora and should not be mistaken for ours: REVE and BrainOmni both pre-train on ds002778, a separate Parkinson's resting-state set from UC San Diego, and BrainOmni on ds004796, an Alzheimer's-risk cohort.

The remaining EEG models pre-train on TUEG, PhysioNet, SEED and comparable corpora, which do not redistribute OpenNeuro recordings. Among the spike models, POYO-1 and NDT3 pre-train on non-human primate motor cortex; the NDT3 checkpoint we load is the 200-hour configuration, whose released dataset list contains eleven monkey reaching corpora and no rodent or human source. MtM pre-trains on mouse recordings from other brain regions collected by a different laboratory.

The one partial exposure noted in Section 4.1 is BrainOmni's use of ds004504 as its own downstream benchmark, where it appears as a 65-subject two-class subset of the 88 subjects we use. Its authors state that all datasets used for downstream testing were excluded from pre-training, so the pre-training claim above holds, but the checkpoint's design choices were made with those results visible, and we therefore do not describe it as fully naive to that benchmark.

## C Held-out Evaluation

A cross-validation score that is also used to steer model selection and retries becomes optimistically biased. To separate the contribution of the pre-trained models from that selection process, we reran each task with the selection process removed entirely and with the final score taken on data that no part of the run had touched. Table 4 reports the outcome.

Two points bound what this evaluation shows. It compares one pre-trained model against *Vanilla*, the identical pipeline with the pre-trained models withheld, so the only variable is whether a pre-trained model was used; it is not a comparison against the single-step LLM baselines. And it does not test whether the queue would have selected the strongest model, since the model is fixed in advance rather than chosen.

The PD row is worth reading carefully. An earlier run of this experiment on three seeds put the PD gap at $+5.14$; all three happened to favor BrainOmni. At eight seeds the gap is $+0.68$ and five of eight seeds favor it, which is the near-tie reported here. This does not contradict the PD result in Table 2, a clear gain over Codex: that table reports the full system with selection enabled, which is a different comparison from the one made here, where a single model is held fixed against *Vanilla* on a fold no part of the run has touched. The drop from three seeds to eight is a concrete instance of the small-sample effect that motivated this evaluation in the first place.

Table 4: Held-out evaluation, $n = 8$ seeds (5678, 6543, 7890, 42, 123, 456, 2048, 4096). Stratified 70/15/15 train/validation/test split; every hyperparameter is selected by cross-validation on the training portion alone, and the test fold is read once, at the end. Queue search, early stopping and the Controller retry loop are switched off for these runs, which consist of a single initial round with the model fixed in advance, so no advantage reported here can arise from adaptive selection. The comparison is paired: for a given seed both arms use the same partition. *Seeds* counts how many of the eight favor the pre-trained model.

| Dataset | Metric | Pre-trained model | Vanilla | $\Delta$ | Seeds |
|---|---|---|---|---|---|
| AD | Macro F1 | $57.71 \pm 16.22$ (REVE) | $44.37 \pm 11.01$ | $+13.34$ | 6/8 |
| WM | Bal. Acc. | $75.30 \pm 8.39$ (POYO-1) | $68.48 \pm 7.49$ | $+6.82$ | 7/8 |
| PD | Bal. Acc. | $68.12 \pm 7.11$ (BrainOmni) | $67.45 \pm 6.75$ | $+0.68$ | 5/8 |

## D Full Experimental Results

Tables 5 and 6 present the complete per-model, per-stage results on the AD and PD benchmarks, including all baseline comparisons and the iterative Controller optimization trajectory for each pre-trained model. WM results are reported in Table 2 for the baseline comparison and in the WM row of Table 4 for the held-out evaluation.

## E Timeouts and the BrainOmni Re-run

Table 5 reports BrainOmni's initial round on AD as $20.76 \pm 28.79$ Macro F1. A standard deviation larger than the mean is the signature of a budget failure rather than a modeling one: several seeds produced no score at all because the script exceeded the 300 s execution limit, and those seeds enter the average as zeros. BrainOmni has the largest embedding dimension of the nine models we integrate (4096; Table 1), so it is the model this uniform budget binds hardest.

We re-ran BrainOmni on AD with the model fixed in advance and a more generous allocation, over the same eight seeds. Table 7 gives the result. Every metric roughly triples, but the informative change is the dispersion: each standard deviation falls by a factor of eight or more, which is what one expects when zeros stop entering the average. The re-runs completed in 318–680 s per seed and every seed produced a score, which is what the collapse in dispersion reflects. We omit the three per-class recall columns of Table 5: the generated scripts record them under inconsistent keys across seeds, and we could not establish a single class ordering we trust for all eight.

Table 7: BrainOmni on AD under the standard budget, as reported in Table 5, and re-run with a more generous allocation over the same eight seeds. Every standard deviation falls by a factor of eight or more.

| Metric | Standard budget | Re-run |
|---|---|---|
| Macro F1 (%) | $20.76 \pm 28.79$ | $\mathbf{62.32 \pm 3.27}$ |
| Bal. Acc. (%) | $21.09 \pm 29.22$ | $\mathbf{62.85 \pm 3.49}$ |
| Acc. (%) | $21.60 \pm 29.92$ | $\mathbf{64.17 \pm 3.31}$ |
| AUROC (%) | $28.34 \pm 39.11$ | $\mathbf{81.43 \pm 1.90}$ |

This also resolves the cross-dataset discrepancy for this model. Ranked by the initial round under each dataset's own primary metric, BrainOmni places last of seven on AD but first of seven on PD, which is not what one expects from the same pre-trained model on two resting-state EEG corpora. With the timeout lifted it places first on AD as well, and the two datasets agree. The discrepancy was a property of the compute budget, not of the model.

We report the original figure in Table 5 rather than substituting the re-run, because the table describes what the system produces under the budget every model is given. The re-run is reported here so that the number is not read as a statement about BrainOmni itself.

## F Web Interface

Figures 3–9 illustrate the full NS-Copilot web interface workflow on the PD EEG classification task.

Table 5: Full results on the AD EEG benchmark (mean $\pm$ std over 8 seeds). Top section: baselines. Bottom section: NS-Copilot with each model across three Controller stages (Initial Run, Retry 1, Retry 2). "Vanilla" denotes NS-Copilot without pre-trained models. The primary metric is Macro F1. *Eff.* is end-to-end wall-clock time and already includes the Codex baseline run that sets the early-stopping threshold.

| **Method** | **Macro F1 (%)** | **Bal. Acc. (%)** | **Acc. (%)** | **AUROC (%)** | **AD Rec. (%)** | **Healthy Rec. (%)** | **FTD Rec. (%)** | **Eff. (s)** |
|---|---|---|---|---|---|---|---|---|
| *Baselines* | | | | | | | | |
| AutoML-Agent | $5.55 \pm 15.71$ | $5.90 \pm 16.69$ | $6.95 \pm 19.64$ | $9.38 \pm 26.52$ | $9.38 \pm 26.52$ | $0.00 \pm 0.00$ | $8.33 \pm 23.57$ | 255 |
| LAMBDA | $43.57 \pm 38.28$ | $44.06 \pm 38.57$ | $45.37 \pm 39.33$ | $54.99 \pm 45.83$ | $33.61 \pm 36.35$ | $50.00 \pm 53.45$ | $11.07 \pm 12.16$ | 100 |
| AutoGen | $26.75 \pm 11.70$ | $27.25 \pm 15.03$ | $32.75 \pm 10.59$ | $44.25 \pm 19.95$ | $52.00 \pm 25.14$ | $20.38 \pm 19.70$ | $5.00 \pm 14.14$ | 103 |
| Claude-Sonnet-4.5 | $48.70 \pm 6.58$ | $48.63 \pm 6.97$ | $51.00 \pm 7.18$ | $68.47 \pm 3.27$ | $63.15 \pm 10.48$ | $61.91 \pm 12.36$ | $30.86 \pm 13.47$ | 304 |
| Codex | $51.01 \pm 2.52$ | $51.24 \pm 2.75$ | $53.98 \pm 2.98$ | $69.81 \pm 1.15$ | $64.58 \pm 4.64$ | $62.50 \pm 3.88$ | $26.63 \pm 3.63$ | 68 |
| *NS-Copilot (Vanilla)* | | | | | | | | |
| Initial Run | $49.80 \pm 3.29$ | $52.21 \pm 2.54$ | $55.46 \pm 2.32$ | $69.15 \pm 3.69$ | $67.66 \pm 3.58$ | $71.51 \pm 4.40$ | $17.71 \pm 6.72$ | 440 |
| Retry 1 | $51.15 \pm 5.74$ | $53.20 \pm 4.24$ | $55.21 \pm 4.70$ | $68.09 \pm 6.59$ | $66.12 \pm 6.59$ | $60.72 \pm 19.09$ | $32.29 \pm 20.14$ | 385 |
| Retry 2 | $47.47 \pm 9.53$ | $49.64 \pm 9.11$ | $51.77 \pm 8.59$ | $66.65 \pm 6.87$ | $65.34 \pm 8.79$ | $59.95 \pm 18.25$ | $30.50 \pm 22.15$ | 252 |
| *NS-Copilot (LaBraM)* | | | | | | | | |
| Initial Run | $28.39 \pm 3.89$ | $29.54 \pm 4.00$ | $31.11 \pm 4.18$ | $44.95 \pm 4.65$ | $32.99 \pm 14.72$ | $25.86 \pm 12.23$ | $17.92 \pm 9.13$ | 486 |
| Retry 1 | $31.47 \pm 6.98$ | $32.50 \pm 6.86$ | $33.88 \pm 7.52$ | $47.10 \pm 8.34$ | $38.98 \pm 8.87$ | $36.11 \pm 11.97$ | $23.02 \pm 5.28$ | 128 |
| Retry 2 | $26.21 \pm 3.79$ | $27.20 \pm 3.91$ | $28.16 \pm 3.52$ | $41.73 \pm 4.02$ | $37.09 \pm 9.04$ | $32.85 \pm 13.62$ | $20.52 \pm 7.12$ | 344 |
| *NS-Copilot (REVE)* | | | | | | | | |
| Initial Run | $52.95 \pm 4.49$ | $54.16 \pm 4.89$ | $56.11 \pm 5.20$ | $71.85 \pm 3.19$ | $60.79 \pm 7.01$ | $68.61 \pm 5.90$ | $32.99 \pm 4.45$ | 236 |
| Retry 1 | $51.05 \pm 6.23$ | $52.40 \pm 6.20$ | $54.08 \pm 6.73$ | $69.26 \pm 4.49$ | $58.06 \pm 8.22$ | $66.75 \pm 6.37$ | $32.38 \pm 5.59$ | 296 |
| Retry 2 | $50.04 \pm 5.24$ | $51.21 \pm 5.40$ | $52.86 \pm 5.45$ | $67.81 \pm 3.21$ | $57.00 \pm 4.92$ | $65.23 \pm 8.26$ | $31.50 \pm 7.23$ | 190 |
| *NS-Copilot (CBraMod)* | | | | | | | | |
| Initial Run | $34.59 \pm 9.74$ | $36.23 \pm 9.31$ | $36.10 \pm 9.92$ | $50.86 \pm 10.24$ | $39.62 \pm 12.80$ | $36.58 \pm 14.35$ | $30.61 \pm 7.85$ | 469 |
| Retry 1 | $35.17 \pm 10.32$ | $37.28 \pm 9.14$ | $38.08 \pm 10.00$ | $53.89 \pm 8.40$ | $42.50 \pm 13.16$ | $43.06 \pm 17.73$ | $25.69 \pm 9.78$ | 262 |
| Retry 2 | $38.06 \pm 6.91$ | $39.86 \pm 5.94$ | $40.97 \pm 6.37$ | $56.59 \pm 5.72$ | $42.96 \pm 9.71$ | $49.32 \pm 13.40$ | $26.35 \pm 9.71$ | 281 |
| *NS-Copilot (EEGPT)* | | | | | | | | |
| Initial Run | $48.83 \pm 3.13$ | $49.91 \pm 2.89$ | $50.99 \pm 2.24$ | $67.29 \pm 2.41$ | $54.10 \pm 2.75$ | $58.53 \pm 6.39$ | $36.58 \pm 12.68$ | 296 |
| Retry 1 | $48.34 \pm 4.17$ | $49.56 \pm 4.56$ | $50.69 \pm 3.92$ | $66.70 \pm 3.44$ | $52.96 \pm 2.76$ | $59.53 \pm 6.24$ | $36.03 \pm 12.11$ | 175 |
| Retry 2 | $46.46 \pm 11.00$ | $48.31 \pm 9.30$ | $48.96 \pm 9.91$ | $65.58 \pm 8.83$ | $50.38 \pm 10.70$ | $53.96 \pm 18.65$ | $40.50 \pm 7.73$ | 170 |
| *NS-Copilot (EEGMamba)* | | | | | | | | |
| Initial Run | $30.11 \pm 4.42$ | $31.39 \pm 4.67$ | $32.42 \pm 4.33$ | $50.09 \pm 3.23$ | $39.61 \pm 9.23$ | $29.69 \pm 3.21$ | $24.60 \pm 8.56$ | 475 |
| Retry 1 | $34.10 \pm 5.41$ | $35.14 \pm 5.58$ | $35.89 \pm 5.42$ | $52.12 \pm 5.33$ | $41.81 \pm 9.77$ | $34.12 \pm 7.56$ | $28.79 \pm 7.32$ | 527 |
| Retry 2 | $35.74 \pm 6.53$ | $36.27 \pm 5.97$ | $37.39 \pm 5.81$ | $54.17 \pm 5.27$ | $43.54 \pm 9.73$ | $37.12 \pm 6.11$ | $28.17 \pm 8.91$ | 569 |
| *NS-Copilot (BrainOmni)* | | | | | | | | |
| Initial Run | $20.76 \pm 28.79$ | $21.09 \pm 29.22$ | $21.60 \pm 29.92$ | $28.34 \pm 39.11$ | $22.23 \pm 30.72$ | $25.44 \pm 35.41$ | $15.76 \pm 22.29$ | 864 |
| Retry 1 | $28.82 \pm 30.84$ | $29.24 \pm 31.27$ | $29.97 \pm 32.05$ | $38.33 \pm 40.98$ | $30.90 \pm 33.11$ | $35.79 \pm 38.32$ | $21.20 \pm 23.41$ | 642 |
| Retry 2 | $27.28 \pm 29.44$ | $27.96 \pm 30.07$ | $28.55 \pm 30.74$ | $36.71 \pm 39.51$ | $29.45 \pm 31.98$ | $33.40 \pm 36.25$ | $21.06 \pm 23.23$ | 597 |
| *NS-Copilot (Auto Mode)* | | | | | | | | |
| Best model | $\mathbf{57.16 \pm 5.71}$ | $\mathbf{58.52 \pm 5.30}$ | $\mathbf{60.57 \pm 4.32}$ | $\mathbf{75.09 \pm 4.83}$ | $\mathbf{66.18 \pm 3.62}$ | $\mathbf{72.97 \pm 4.74}$ | $\mathbf{36.25 \pm 14.86}$ | 425 |

Table 6: Full results on the PD EEG benchmark (mean ± std over 8 seeds). Top section: baselines. Bottom section: NS-Copilot with each model across three Controller stages (Initial Run, Retry 1, Retry 2). "Vanilla" denotes NS-Copilot without pre-trained models. The primary metric is Balanced Accuracy. Note that Claude-Sonnet-4.5 achieves the highest PD Recall but at the cost of substantially lower Control Recall (46.28%), indicating a strong prediction bias toward the majority class. *Eff.* is end-to-end wall-clock time and already includes the Codex baseline run that sets the early-stopping threshold.

| **Method** | **Bal. Acc. (%)** | **Acc. (%)** | **Macro F1 (%)** | **AUROC (%)** | **Control Rec. (%)** | **PD Rec. (%)** | **Eff. (s)** |
|---|---|---|---|---|---|---|---|
| *Baselines* | | | | | | | |
| AutoML-Agent | $17.39 \pm 32.29$ | $18.33 \pm 33.95$ | $9.15 \pm 25.88$ | $19.90 \pm 37.01$ | $10.71 \pm 30.30$ | $7.81 \pm 22.10$ | 293 |
| LAMBDA | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | 284 |
| AutoGen | $6.25 \pm 17.68$ | $8.38 \pm 23.69$ | $5.00 \pm 14.14$ | $6.25 \pm 17.68$ | $0.00 \pm 0.00$ | $12.50 \pm 35.36$ | 308 |
| Claude-Sonnet-4.5 | $65.12 \pm 3.27$ | $70.99 \pm 1.54$ | $65.50 \pm 3.43$ | $74.07 \pm 3.52$ | $46.28 \pm 12.37$ | $\mathbf{84.50 \pm 7.53}$ | 516 |
| Codex | $65.52 \pm 7.98$ | $64.95 \pm 12.87$ | $62.08 \pm 12.98$ | $71.07 \pm 8.23$ | $64.57 \pm 10.11$ | $61.00 \pm 22.34$ | 125 |
| *NS-Copilot (Vanilla)* | | | | | | | |
| Initial Run | $64.52 \pm 1.25$ | $70.49 \pm 1.18$ | $64.40 \pm 0.66$ | $74.41 \pm 0.93$ | $47.34 \pm 7.57$ | $81.82 \pm 5.33$ | 113 |
| Retry 1 | $59.09 \pm 8.50$ | $61.65 \pm 12.15$ | $53.99 \pm 13.74$ | $64.74 \pm 7.77$ | $51.57 \pm 30.95$ | $66.58 \pm 28.92$ | 156 |
| Retry 2 | $60.48 \pm 7.08$ | $61.59 \pm 12.23$ | $56.08 \pm 13.43$ | $66.14 \pm 7.78$ | $57.16 \pm 23.88$ | $63.75 \pm 27.14$ | 158 |
| *NS-Copilot (LaBraM)* | | | | | | | |
| Initial Run | $59.48 \pm 0.56$ | $61.08 \pm 0.65$ | $58.35 \pm 0.54$ | $60.20 \pm 1.32$ | $55.10 \pm 0.00$ | $64.08 \pm 0.94$ | 70 |
| Retry 1 | $56.49 \pm 3.81$ | $57.93 \pm 3.99$ | $55.10 \pm 3.86$ | $58.18 \pm 4.17$ | $52.21 \pm 3.46$ | $60.73 \pm 4.34$ | 76 |
| Retry 2 | $60.25 \pm 1.47$ | $61.58 \pm 1.57$ | $58.75 \pm 1.33$ | $60.70 \pm 1.13$ | $56.72 \pm 2.57$ | $64.02 \pm 2.28$ | 95 |
| *NS-Copilot (REVE)* | | | | | | | |
| Initial Run | $62.83 \pm 0.16$ | $68.48 \pm 0.01$ | $63.06 \pm 0.10$ | $63.11 \pm 0.61$ | $46.56 \pm 0.53$ | $79.19 \pm 0.26$ | 78 |
| Retry 1 | $56.47 \pm 6.13$ | $61.08 \pm 8.02$ | $56.27 \pm 6.53$ | $58.92 \pm 7.72$ | $43.07 \pm 1.94$ | $69.85 \pm 11.56$ | 90 |
| Retry 2 | $58.92 \pm 4.40$ | $63.91 \pm 5.75$ | $58.73 \pm 4.69$ | $60.98 \pm 5.29$ | $44.39 \pm 2.56$ | $73.44 \pm 8.45$ | 117 |
| *NS-Copilot (CBraMod)* | | | | | | | |
| Initial Run | $58.72 \pm 0.42$ | $60.70 \pm 0.51$ | $57.37 \pm 0.51$ | $60.73 \pm 0.66$ | $53.23 \pm 0.31$ | $64.42 \pm 0.83$ | 85 |
| Retry 1 | $59.58 \pm 3.95$ | $61.64 \pm 5.26$ | $58.11 \pm 4.60$ | $62.64 \pm 4.16$ | $53.36 \pm 3.95$ | $65.67 \pm 8.29$ | 87 |
| Retry 2 | $60.55 \pm 1.57$ | $62.95 \pm 1.46$ | $59.24 \pm 1.41$ | $63.86 \pm 3.26$ | $53.23 \pm 3.37$ | $67.64 \pm 2.38$ | 97 |
| *NS-Copilot (EEGPT)* | | | | | | | |
| Initial Run | $65.72 \pm 1.24$ | $68.38 \pm 1.45$ | $65.16 \pm 1.22$ | $69.22 \pm 1.41$ | $57.69 \pm 0.60$ | $73.69 \pm 1.81$ | 91 |
| Retry 1 | $58.42 \pm 10.01$ | $59.90 \pm 10.47$ | $57.21 \pm 10.18$ | $61.42 \pm 13.44$ | $53.91 \pm 9.66$ | $62.88 \pm 11.73$ | 106 |
| Retry 2 | $61.08 \pm 8.28$ | $62.57 \pm 8.29$ | $59.86 \pm 8.23$ | $64.91 \pm 10.97$ | $56.67 \pm 9.28$ | $65.48 \pm 8.76$ | 126 |
| *NS-Copilot (EEGMamba)* | | | | | | | |
| Initial Run | $56.14 \pm 0.39$ | $57.92 \pm 0.38$ | $54.81 \pm 0.41$ | $57.79 \pm 0.60$ | $51.19 \pm 0.31$ | $61.27 \pm 0.52$ | 96 |
| Retry 1 | $48.31 \pm 4.86$ | $50.35 \pm 5.12$ | $46.96 \pm 5.15$ | $49.80 \pm 6.61$ | $42.22 \pm 5.46$ | $54.31 \pm 5.91$ | 77 |
| Retry 2 | $52.11 \pm 2.65$ | $56.11 \pm 5.26$ | $50.39 \pm 3.00$ | $52.84 \pm 3.59$ | $40.31 \pm 13.97$ | $63.81 \pm 13.76$ | 104 |
| *NS-Copilot (BrainOmni)* | | | | | | | |
| Initial Run | $67.29 \pm 2.44$ | $71.39 \pm 1.88$ | $67.21 \pm 2.46$ | $73.50 \pm 1.77$ | $55.36 \pm 4.07$ | $79.21 \pm 1.30$ | 114 |
| Retry 1 | $49.99 \pm 21.81$ | $51.12 \pm 22.86$ | $48.78 \pm 21.68$ | $52.36 \pm 23.91$ | $46.64 \pm 19.40$ | $53.29 \pm 25.15$ | 74 |
| Retry 2 | $56.28 \pm 7.41$ | $57.83 \pm 8.71$ | $54.99 \pm 7.94$ | $61.15 \pm 11.79$ | $51.74 \pm 5.36$ | $60.79 \pm 11.54$ | 105 |
| *NS-Copilot (Auto Mode)* | | | | | | | |
| Best model | $\mathbf{71.32 \pm 2.97}$ | $\mathbf{71.91 \pm 1.87}$ | $\mathbf{69.22 \pm 2.47}$ | $\mathbf{78.15 \pm 1.21}$ | $\mathbf{69.80 \pm 11.88}$ | $72.95 \pm 7.27$ | 975 |

NS-Copilot

Classify subjects into PD (Parkinson's Disease) and Control groups using the uploaded EEG dataset. Do NOT use any clinical scores (e.g. MOCA, UPDRS), demographic information, or any non-EEG columns from metadata files as classifier features. Report accuracy, balanced accuracy, macro-averaged F1 score, macro-averaged AUROC, per-class recall, and confusion matrix.

**Running Codex Baseline & Planner**

Codex baseline and Planner are running in parallel...

**Planner Decision**

**Primary Metric:** balanced_accuracy

**Model Queue:** REVE, EEGPT, BrainOmni, EEGMamba, CBraMod, LaBraM, Vanilla

**Reasoning:** This is a binary PD vs Control classification task where class imbalance is common/unknown, so balanced_accuracy is the most appropriate primary metric for model comparison while still reporting accuracy, macro-F1, macro-AUROC, per-class recall, and confusion matrix. Only EEG foundation models are applicable; spike models (ndt3/mtm/poyo1) are excluded due to data-type mismatch. Ordering prioritizes large, broadly pretrained EEG models with flexible montage support (ReVE) and strong general-purpose EEG encoders (EEGPT, BrainOmni), followed by efficient channel-agnostic models (EEGMamba, CBRaMod), then LabRAM (more constrained to 10-20), and finally a vanilla handcrafted-feature baseline.

**Codex Baseline Result**

**balanced_accuracy:** 0.6419

**Duration:** 102s

**Metrics:** {
"accuracy": 0.6242,
"balanced_accuracy": 0.6419,

Figure 3: User input with task description, parallel performance threshold computation and Planner execution, and Planner's model queue with reasoning.

NS-Copilot

**balanced_accuracy:** 0.6419

**Duration:** 102s

**Metrics:** {
"accuracy": 0.6242,
"balanced_accuracy": 0.6419,
"f1_macro": 0.6133,
"auroc_macro": 0.731,
"recall_control": 0.6939,
"recall_pd": 0.59
}

**Running REVE (1/7)**

**Initial** — REVE is being executed...

**REVE Completed**

**Initial** — balanced_accuracy = 0.6229 (Codex baseline: 0.6419). Early stopping not triggered. Now running **EEGPT**...

**Running EEGPT (2/7)**

**Initial** — EEGPT is being executed...

Figure 4: Performance threshold metrics and initial model search (REVE followed by EEGPT).

🤖 **NS-Copilot**

💻 **Generated Code (Winner)**

**Model:** EEGPT (Initial)

```
# ----------------------------
# Multi-step PD vs Control decoding on EEG-only features
# Steps:
# 1) EEGPT embeddings -> subject-level decoding (CV)
# 2) Spectral bandpower features extraction (EEG-only)
# 3) Logistic regression decoding on bandpower (CV)
# 4) Confusion matrix plot for EEGPT decoding (Control, PD order) + recall interpretation
# ----------------------------

# Helper: robust subject-id extraction from file path
def _extract_sub_id(path):
    m = re.search(r"(sub-\d+)", str(path))
    return m.group(1) if m else None

# Helper: compute required metrics from y_true/y_pred and optional y_proba
def _compute_required_metrics(y_true, y_pred, y_proba=None, class_labels=None):
    y_true = np.asarray(y_true)
    y_pred = np.asarray(y_pred)

    # Ensure integer labels for metric functions
    le = LabelEncoder()
    le.fit(y_true if class_labels is None else np.asarray(class_labels))
    y_true_enc = le.transform(y_true)
    y_pred_enc = le.transform(y_pred)

    labels_enc = np.arange(len(le.classes_))
    cm = confusion_matrix(y_true_enc, y_pred_enc, labels=labels_enc)

    acc = accuracy_score(y_true_enc, y_pred_enc)
    bacc = balanced_accuracy_score(y_true_enc, y_pred_enc)
    mf1 = f1_score(y_true_enc, y_pred_enc, average="macro", labels=labels_enc, zero_division=0)

    # Per-class recall
    per_class_recall_vals = {}
```

Figure 5: Generated Python code for the winning model (EEGPT).

🤖 **NS-Copilot**

## Analysis Summary ✓

Using EEG-only features extracted with an EEGPT-based embedding approach, PD vs Control classification achieved moderate performance (accuracy 0.738, macro-AUROC 0.742). Performance was notably asymmetric: PD was detected well (recall 0.83) while Control recall was substantially lower (0.55), indicating a tendency to over-predict PD.

**Key Findings**

1. **Overall discrimination between PD and Control is moderate using EEG-only embeddings.**
   - Evidence: Accuracy=0.7383, balanced accuracy=0.6905, macro-F1=0.6952, macro-AUROC=0.7422 (n=149 subjects). Cross-validated macro-AUROC mean=0.7360±0.0912; balanced accuracy mean=0.6730±0.0757; macro-F1 mean=0.6587±0.0719.
   - Significance: EEG contains disease-relevant information sufficient for above-chance group classification without clinical/demographic covariates, but performance is not yet strong enough for high-stakes individual-level decisions.
2. **Classifier sensitivity is higher for PD than for Controls, suggesting bias toward predicting PD and/or greater heterogeneity in Controls.**
   - Evidence: Per-class recall: Control=0.5510 (27/49), PD=0.8300 (83/100). Confusion matrix [[27, 22], [17, 83]] shows 22 Controls misclassified as PD vs 17 PD misclassified as Control.
   - Significance: This asymmetry implies elevated false-positive PD risk (Controls labeled as PD), which is clinically problematic for screening; it also indicates that balanced metrics (balanced accuracy, macro-F1) are more informative than raw accuracy given class imbalance.
3. **Generalization appears variable across folds/repeats, consistent with limited sample size and/or subject heterogeneity.**
   - Evidence: 5-fold CV repeated 3 times: accuracy mean=0.6890±0.0687; macro-AUROC mean=0.7360±0.0912; balanced accuracy mean=0.6730±0.0757; macro-F1 mean=0.6587±0.0719.
   - Significance: The non-trivial standard deviations suggest performance depends on the train/test split; more data and/or improved modeling/feature strategies may be needed for stable deployment.

**Metrics Interpretation**

- **accuracy**: Overall proportion of correctly classified subjects across both classes; can be inflated when classes are imbalanced (here PD appears more prevalent than Control).
- **balanced_accuracy**: Mean of per-class recalls (sensitivity for each class), mitigating class-imbalance effects; here it highlights reduced Control sensitivity relative to PD.

Figure 6: Interpreter analysis summary with key findings and metric interpretation.

NS-Copilot

- **per_class_recall**: Class-specific sensitivity: proportion of true class members correctly identified; here PD recall is high (0.83) while Control recall is low (0.55), indicating many Controls are labeled PD.
- **confusion_matrix**: Counts of true vs predicted labels; reveals error types (false positives/false negatives) and class-specific failure modes. Here, false positives for PD (Controls→PD) are prominent.

**Limitations**

- Class imbalance (PD=100 vs Control=49) likely contributes to biased decision boundaries and reduced Control recall; accuracy alone is therefore insufficient.
- Only a single operating point is reported for the confusion matrix; without threshold tuning or calibration, the chosen threshold may not reflect the best trade-off for the intended use (e.g., screening vs confirmation).
- Cross-validation variability (e.g., macro-AUROC std ~0.09) suggests limited stability, potentially due to small sample size, inter-subject heterogeneity, or site/session effects not accounted for.
- The results reflect one feature extraction/model configuration (EEGPT embeddings with mean pooling); alternative EEG representations or preprocessing choices may yield different performance.
- No external held-out test set is reported; repeated CV estimates generalization but can still be optimistic if there are subtle dependencies in the dataset.

**Recommendations for Next Steps**

1. Address class imbalance by using class-weighted loss, balanced sampling, or threshold adjustment to improve Control recall; report the resulting trade-offs (e.g., sensitivity/specificity curves).
2. Perform probability calibration (Platt scaling or isotonic regression) and select an operating threshold aligned with the application (minimize false positives vs false negatives).
3. Report additional uncertainty estimates (e.g., bootstrap CIs for AUROC/balanced accuracy) and inspect fold-wise performance to identify unstable splits or outlier subjects.
4. Explore feature/model variants: alternative pooling (attention), band-specific embeddings, channel selection/robust referencing, artifact handling, and simpler baselines (bandpower + linear models) for interpretability and benchmarking.
5. If metadata includes recording site/device/session identifiers, use them only for stratification or grouping (not as features) to reduce leakage and assess robustness across acquisition conditions.
6. Validate on an external dataset or a strict held-out cohort split (e.g., by acquisition batch) to better estimate real-world generalization.

Figure 7: Interpreter limitations and recommendations for next steps.

NS-Copilot

## Auto Model Selection Results (with Codex Baseline)

**Primary metric**: balanced_accuracy
**Codex baseline**: 0.6419387755102041
**Best model**: 🏆 EEGPT (balanced_accuracy=0.6905)
**Early stopped**: True

**Model order** (LLM-decided): ['REVE', 'EEGPT', 'BrainOmni', 'EEGMamba', 'CBraMod', 'LaBraM', 'Vanilla']
**Reasoning**: This is a binary PD vs Control classification task where class imbalance is common/unknown, so balanced_accuracy is the most appropriate primary metric for model comparison while still reporting accuracy, macro-F1, macro-AUROC, per-class recall, and confusion matrix. Only EEG foundation models are applicable; spike models (ndt3/mtm/poyo1) are excluded due to data-type mismatch. Ordering prioritizes large, broadly pretrained EEG models with flexible montage support (ReVE) and strong general-purpose EEG encoders (EEGPT, BrainOmni), followed by efficient channel-agnostic models (EEGMamba, CBRaMod), then LabRAM (more constrained to 10-20), and finally a vanilla handcrafted-feature baseline.

**Per-model results:**

**REVE**: balanced_accuracy=0.6229 ≤ Codex baseline (0.6419)
🏆 **EEGPT**: balanced_accuracy=0.6905 (Initial)
**BrainOmni**: skipped (early stopping triggered)
**EEGMamba**: skipped (early stopping triggered)
**CBraMod**: skipped (early stopping triggered)
**LaBraM**: skipped (early stopping triggered)
**Vanilla**: skipped (early stopping triggered)

**Comparison: Codex Baseline vs Best Model**

| Metric | Codex Baseline | 🏆 EEGPT |
|---|---|---|
| accuracy | 0.6242 | **0.7383** |

Figure 8: Auto Model Selection results with per-model execution outcomes, and a comparison table between the winner model and the performance threshold.

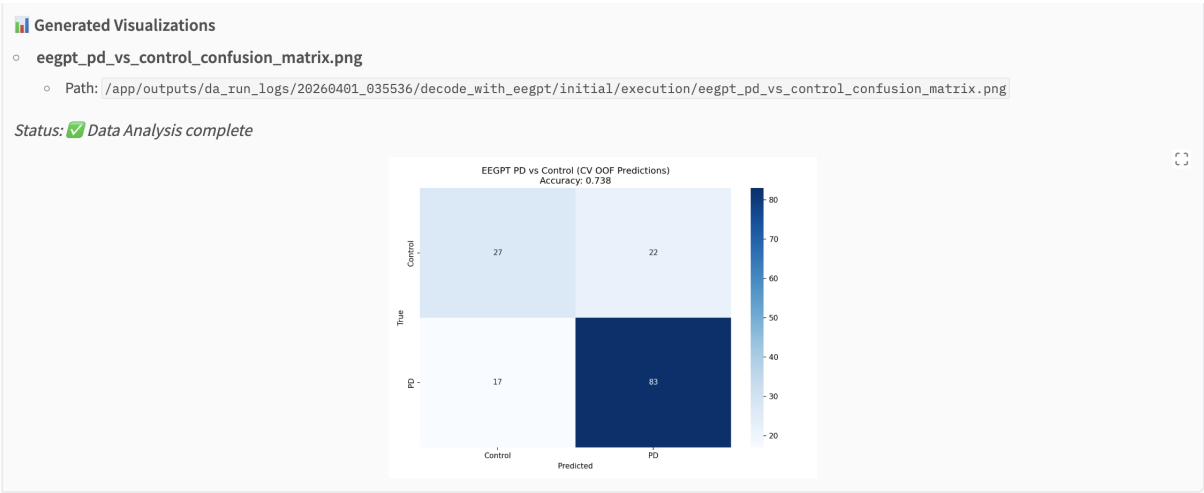


Figure 9: NS-Copilot web interface: generated confusion matrix visualization and task completion status.