Early Prediction of Future Behavioral Strategy from Process Traces

Robert Kasumba
Division of Computational and Data Sciences
Washington University in St. Louis
Saint Louis, MO 63130
rkasumba@wustl.edu

Dennis L. Barbour
Department of Biomedical Engineering
Washington University in St. Louis
Saint Louis, MO 63130
dbarbour@wustl.edu

Chien-Ju Ho
Department of Computer Science and Engineering
Washington University in St. Louis
Saint Louis, MO 63130
chienju.ho@wustl.edu

Abstract

Adaptive systems often need to make task-specific decisions about people based on limited observations of their prior behavior: a tutor may need to anticipate how a learner will approach a new exercise, a game may need to adapt when a player enters a new level, and a human-AI system may need to infer whether a partner will persist with a plan or switch goals. These decisions depend on person-level tendencies that shape how people solve related tasks, but such tendencies are difficult to infer from standard behavioral evidence. One approach is to use aggregate outcome summaries, such as scores, completion rates, or productivity; these summaries are compact and available across tasks, but can collapse distinct behavioral processes into similar outcomes. Another approach is to use process-level traces, which record how behavior unfolds; however, process modeling within one task can entangle stable person-level tendencies with task-specific layout and affordances. In this work, we study early cross-task behavioral inference: whether partial source-task process traces can reveal transferable person-level structure that predicts strategy in a held-out target task. We introduce a Process-Level Latent Variable Model (PLVM), which encodes task-specific traces and fuses them into a shared person-level latent representation for cross-task prediction. We evaluate PLVM using human gameplay traces from PowerWash Simulator. Using partial traces from two earlier levels, we predict whether a player will tend to persist within local regions or frequently switch between regions in a later, held-out level. Our results show that models using process traces substantially outperform models based on aggregate outcomes, which remain near chance. Combining traces from both source tasks provides a modest additional improvement over single-task models. Controlled simulations with known latent behavioral types further show that cross-task fusion is particularly beneficial when source tasks reveal complementary aspects of the underlying behavioral tendencies.

1 Introduction

Adaptive systems often need to make task-specific decisions about a person from limited prior evidence. A tutor may need to anticipate how a learner will approach a new problem [24], a game may need to adapt when a player enters a new level [20], and a human-AI partner may need to infer

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

whether a collaborator will persist with a plan or switch goals in a new collaborative episode [9, 12]. Such decisions depend on person-level behavioral tendencies that shape how people solve related tasks. The useful prediction is therefore not only what outcome the person has achieved before, but how the person is likely to organize behavior in the target task.

A natural starting point for this prediction problem is to use outcome summaries of prior behavior. Behavioral science and applied assessment have often relied on measures such as accuracy, reaction time, score, completion time, or total progress to estimate individual differences [19, 7, 11, 13]. These summaries are compact and often available across tasks, but they can underdetermine the behavioral process that produced them: similar outcomes can arise from systematic spatial organization, local exploitation, switching, or long-range movement toward future subtasks. Even when outcome summaries are combined across tasks, they abstract away how behavior unfolds within each task [19, 7, 17, 3]. This abstraction limits adaptive systems since different behavioral processes may require different interventions even with similar observed outcomes.

This motivates the use of process traces: time-ordered behavioral records that preserve information outcome summaries discard [10, 1]. In our setting, these traces capture how behavior unfolds within a task, including where people move, when they switch targets, how they approach subtasks, and how progress accumulates. Modern sequence models, such as transformers [25], can learn rich temporal dependencies in such traces and model how behavior continues within a task. However, richer process data alone does not solve the cross-task inference problem. A strong sequence model may learn the layout, timing, and local affordances of a particular environment, entangling task-specific regularities with stable person-level tendencies. Conversely, a relevant person-level tendency may be only partially expressed in any single source task: a learner's persistence, a player's search style, or a teammate's switching tendency may appear differently across environments. The key challenge is therefore not merely to model process traces within a task, but to combine process evidence across tasks to infer behavioral structure that transfers to a held-out target task.

In this work, we study early cross-task behavioral inference: predicting how a person will organize behavior in a held-out target task from partial process traces in prior source tasks. We ask whether process traces from multiple source tasks can be fused to recover transferable person-level behavioral structure beyond what is captured by aggregate outcome summaries or single-task process models. We instantiate this problem in the PowerWash Simulator gameplay dataset [27], which contains naturalistic telemetry from human players completing long-horizon spatial cleaning tasks. We use partial traces from two source levels to predict navigation style in a later held-out target level: whether a player tends to persist locally within spatial regions or repeatedly switch across regions. This distinction is actionable because players with similar completion outcomes may require different forms of assistance, feedback, or coordination depending on how they organize their behavior.

We propose a Process-Level Latent Variable Model (PLVM), which uses task-specific encoders to summarize process traces and combines their outputs into a shared person-level latent representation for predicting behavior in a held-out task. Rather than predicting how behavior continues within a task, PLVM aims to capture behavioral tendencies that may manifest differently across tasks. We evaluate PLVM using human gameplay traces from PowerWash Simulator, using partial traces from two earlier cleaning levels to predict whether a player will persist within local regions or frequently switch between regions in a later level. PLVM outperforms outcome-based models, which remain close to chance, and improves over single-task process models. We also use controlled simulations with known latent behavioral types to examine when cross-task fusion helps, showing larger gains when source tasks reveal complementary aspects of the underlying behavioral tendencies.

Overall, our work makes three contributions:

- We formulate the problem of early cross-task behavioral strategy prediction. The goal is to predict an individual's strategy in a held-out task from partial process traces in prior tasks.
- We introduce PLVM, a modular representation-fusion framework that combines independently trained, frozen task-specific process encoders into a shared person-level representation for cross-task prediction.
- We demonstrate using real human gameplay data that process-based models substantially outperform outcome-based models, while cross-task fusion provides modest additional improvements over single-task models. Controlled simulations further show that fusion is particularly beneficial when source tasks reveal complementary aspects of a shared latent process.

2 Related Work

Our work connects to several lines of research: process tracing and behavioral assessment, search and foraging, behavioral telemetry and individual style, and sequence modeling and cross-task representation learning. These areas inform how we characterize behavioral strategies, extract information from process traces, and predict behavior across tasks.

Process tracing and behavioral assessment: Cognitive science has long sought to infer latent cognitive processes from observable behavior. Traditional assessment often relies on outcome measures such as accuracy, reaction time, completion time, or score to estimate individual differences in constructs such as working memory, attention, or cognitive flexibility [19, 7]. These measures are useful but can collapse distinct behavioral processes: two people may achieve similar performance while differing in how they search, plan, persist, or switch goals [17, 3]. Process-tracing methods address this limitation by treating fine-grained behavioral traces as evidence about latent planning and decision strategies [4, 10]. Relatedly, Tang [23] models response-process sequences with a latent-trait hidden Markov model to characterize heterogeneity across respondents. Whereas latent HMMs specify an interpretable stochastic process, PLVM learns task-specific representations from traces and uses them for downstream cross-task prediction. This prior work shows that process data can reveal structure that outcome summaries miss [4, 10, 23]. However, much of this literature focuses on explaining behavior within a task or estimating strategy from traces in the same setting. Our problem is different: we ask whether partial process traces from prior tasks can predict how an individual will organize behavior in a future task.

Search, foraging, and navigation style: Our target behavioral label is motivated by theories of search and foraging. Foraging models study how agents search for distributed resources, exploit local patches, decide when to leave a region, and trade off reward against movement or travel costs [5]. Related ideas appear in information foraging and visual foraging, where human search behavior is shaped by local persistence, switching, and the expected value of moving to another region [21, 2, 15]. These theories provide a useful vocabulary for describing spatial behavior in long-horizon tasks: some individuals persist locally, whereas others switch more frequently across regions. In our work, this distinction motivates the Zone Planner versus Zone Hopper target in PowerWash. The contribution is not a new theory of foraging, but a predictive modeling framework that learns whether such behavioral organization transfers across tasks from partial process traces.

Behavioral telemetry, virtual environments, and individual style: Games and virtual interactive environments provide dense, naturalistic behavioral traces at scale, making them useful for studying individual differences in complex behavior [17]. Prior work on behavioral telemetry has used such traces to model performance, engagement, skill, well-being, or decision style [17, 3]. Closely related is behavioral stylometry, which uses patterns of decisions to detect individual decision-making style, for example in chess [18]. This literature demonstrates that individual style can be measured from behavioral sequences. Action sequences reveal individual style and distinguish strategies with similar aggregate outcomes. However, stylometry and player modeling often operate within a single domain or aim to identify individuals, predict outcomes, or characterize behavior after observation. We instead study an early cross-task prediction problem: given partial behavior in source tasks, predict behavioral strategy in a held-out target task whose traces are not observed as input.

Sequence models and cross-task representation learning: Sequence models, including transformers, can learn temporal dependencies from behavioral traces [25]. They are well suited for modeling within-task dynamics, such as how recent actions predict future actions or how local progress predicts subsequent movement. However, strong within-task prediction does not guarantee recovery of transferable person-level structure. A sequence model may exploit the affordances of a particular task while entangling those regularities with individual behavioral tendencies. Multi-task and domain representation learning address related issues by sharing information across tasks or learning features that generalize across domains, but they are not usually framed around person-level behavioral inference from multiple partial traces [28, 6, 14, 16]. PLVM addresses this setting directly: task-specific encoders summarize process traces from each source task, and a fusion module learns a shared person-level representation used to predict behavior in a held-out target task. The work by Higashi [8] transfers individual representations across a single source condition to predict target actions. PLVM instead asks whether partial process traces from earlier tasks, beyond outcome summaries, reveal a person's future task-level strategy before it is observed; it further tests whether combining evidence across source tasks improves that prediction.

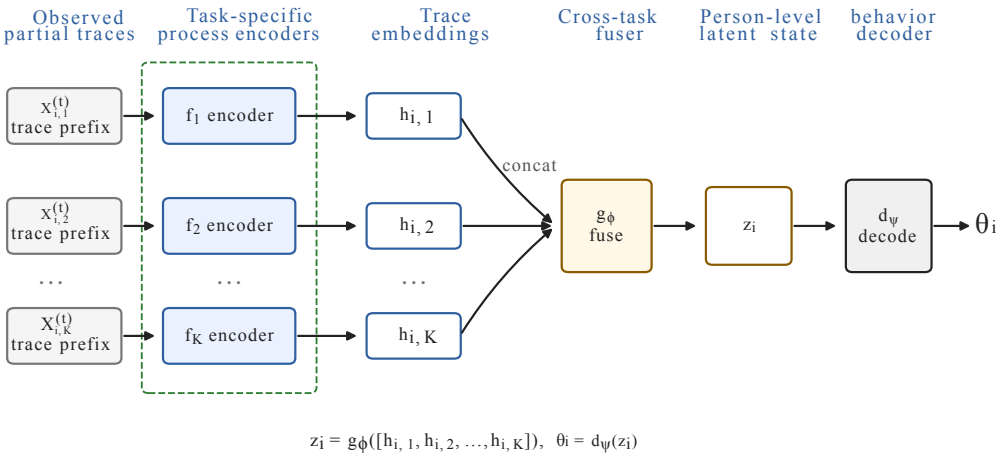


Figure 1: PLVM architecture. Task-specific process encoders map partial source-task traces to fixed-dimensional embeddings. A cross-task fuser combines these embeddings into a shared person-level latent representation, which is decoded to predict the behavioral target in a held-out target task.

3 Methods

We first formalize the problem of predicting behavioral strategy in a held-out task from partial process traces in prior tasks. We then introduce PLVM and describe the evaluation framework used to assess the predictive value of process-level information and cross-task fusion.

3.1 Problem Formulation

We consider individuals $i \in \{1, \ldots, N\}$ observed across multiple tasks. Let $\mathcal{K}_{obs}$ denote the set of observed source tasks and let $k^\star$ denote a held-out target task. For each individual i and source task $k \in \mathcal{K}_{obs}$, we observe a partial process trace

$$X_{i,k}^{(t)} = (x_{i,k,1}, \ldots, x_{i,k,t}),$$

where each token records process-level information available at event t, such as position, progress, elapsed time, local task state, or subtask-completion information. The goal is to predict a behavioral target $y_{i,k^\star}$ in the held-out target task using only partial traces from the source tasks.

3.2 Process-Level Latent Variable Model (PLVM)

PLVM consists of three components: task-specific process encoders, a cross-task fuser, and a target-behavior decoder. Each task encoder summarizes a partial process trace from one source task. The cross-task fuser combines these task embeddings into a shared person-level latent representation, and the decoder uses this representation to predict the behavioral target. In our implementation, the task-specific encoders are trained independently, then frozen; the fuser and decoder are trained jointly for the downstream cross-task prediction task. Figure 1 shows the architecture.

Task-specific process encoders: For each source task $k \in \mathcal{K}_{obs}$, we train an encoder f_k on process traces from that task. The encoder maps a partial trace to a fixed-dimensional embedding:

$$h_{i,k}^{(t)} = f_k(X_{i,k}^{(t)}).$$

In our experiments, f_k is a causal transformer trained with a self-supervised next-event prediction objective. This objective is not the final scientific target; it is used to learn representations of within-task behavioral dynamics. In simulation, the next-event target is the agent's next action or target choice. In PowerWash, the next-event target is the player's position change by the next subtask-completion event. The task-specific encoders are trained independently and then frozen; the fusion module and decoder are trained on the downstream cross-task prediction objective. Freezing

is deliberate: the encoders never see the downstream label, so any gain reflects information already present in the process representations rather than end-to-end adaptation.

Cross-task process fusion: Given embeddings from the source tasks, PLVM[1] constructs a shared person-level latent representation

$$z_i^{(t)} = g_\phi \left(\text{concat}_{k \in \mathcal{K}_{obs}} \, h_{i,k}^{(t)} \right),$$

where g_ϕ is a feed-forward fusion network. The representation $z_i^{(t)}$ is intended to summarize behavioral structure that is shared across tasks. Because each task-specific encoder is trained separately, the fusion module learns how evidence from different source tasks should be combined for the downstream behavioral target.

Target-behavior decoder: A decoder d_ψ maps the person-level representation to prediction scores:

$$\theta_i = d_\psi(z_i^{(t)}).$$

For classification targets, such as behavioral phenotype in simulation or Zone Planner/Zone Hopper in PowerWash, these scores define

$$p_\psi(y \mid z_i^{(t)}) = \text{softmax}(\theta_i)_y.$$

We train the fuser and decoder using cross-entropy loss:

$$\mathcal{L}(\phi, \psi) = -\sum_i \log p_\psi(y_{i,k^\star} \mid z_i^{(t)}).$$

$y_{i,k^\star}$ is defined from held-out target-task behavior, or from a known individual-level phenotype in simulation, while $z_i^{(t)}$ is computed only from partial traces in the source tasks.

3.3 Model Evaluation

Comparison models: We compare PLVM to two classes of baselines. Outcome baselines use aggregate summaries of source-task behavior up to the same observation cutoff, such as completion, progress, reward, productivity, or recent-window statistics. These models test whether target-task behavioral strategy can be predicted from what the person accomplished, without modeling how behavior unfolded. Single-task process models use the same task-specific process encoders as PLVM but receive only one source task at a time. These models test whether detailed traces from a single task are sufficient without cross-task fusion. PLVM differs from both by combining process-level evidence from multiple source tasks into a shared person-level representation. Each comparison isolates a specific source of predictive information. We call a feature an outcome feature if it is invariant to the order of observed events, and a process feature if it depends on that order. Because both model families observe the same prefix, an advantage for process models shows that process-level information adds predictive value; Section 5.1 tests directly whether that value lies in ordering. Because single-task and multi-task process models use the same frozen encoders, an advantage for the latter isolates the contribution of combining evidence across source tasks.

Partial observation protocol: We evaluate all models under matched observation cutoffs. At cutoff t, each model receives only the prefix of each source-task trace up to that cutoff. Outcome baselines compute summaries only from the same prefix. Single-task process models and PLVM use only prefix embeddings from the task-specific encoders. The held-out target-task trace is never provided as input. This protocol tests whether a target-task behavioral strategy can be inferred from partial source-task behavior before complete source-task traces are available.

4 Experiments

We evaluate PLVM in two complementary settings. The primary experiment uses real human gameplay from PowerWash to investigate an applied prediction problem: predicting a player's future navigation style in a held-out task from partial traces in earlier tasks. The second experiment uses a controlled simulation to examine when cross-task fusion is beneficial: whether cross-task fusion helps when different tasks reveal complementary aspects of a known latent behavioral process. Models are compared under the matched observation protocol of Section 3.3.

[1]We use the term "latent" in PLVM to refer to a learned latent representation of unobserved person-level behavioral tendencies, rather than a probabilistic latent variable with an explicit prior or posterior.

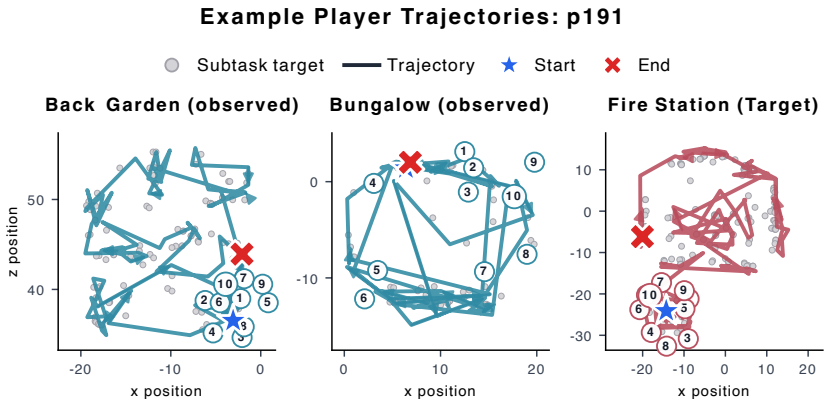


Figure 2: Example PowerWash process traces across the three tasks we used. Each panel shows subtask locations and one player's ordered completion trajectory. Back Garden and Bungalow are used as observed tasks, while Fire Station is the future held-out target task.

4.1 Prediction of Future Navigation Style using Human Gameplay

Dataset and Task Setup: We use the *PowerWash Simulator* gameplay dataset [27]: real human telemetry from 11,080 unique players completing long-horizon spatial cleaning tasks. These tasks require players to search for distributed targets, persist within local regions, switch areas, and move toward future subtask completions. We focus on three career-mode cleaning tasks: Back Garden, Bungalow, and Fire Station, which share the broad structure of spatial cleaning while differing in layout, scale, and local affordances. They also form a natural early-to-late career progression with substantial player overlap. Figure 2 shows one player's behavioral trajectory across the three tasks. We use Back Garden and Bungalow as early observation tasks and Fire Station as the held-out future task: given only a player's process-level behavior in Back Garden and Bungalow, models predict the behavioral strategy the same player will use in Fire Station.

Future Navigation-Style Target: We operationalize future behavioral strategy as the tendency to switch between spatial regions. This is actionable since players with similar completion progress may require different forms of assistance: local object discovery support for persistent players versus route stabilization or reminders for frequent switchers. As shown in Figure 3, we cluster Fire Station subtask locations into spatial zones and convert each player's subtask completions into a zone sequence $q_{i,1:T_i}$. We compute four summary statistics: zone switch rate a_i, return rate r_i, mean zone run length ℓ_i, and mean zone jump distance d_i. The navigation-style score is

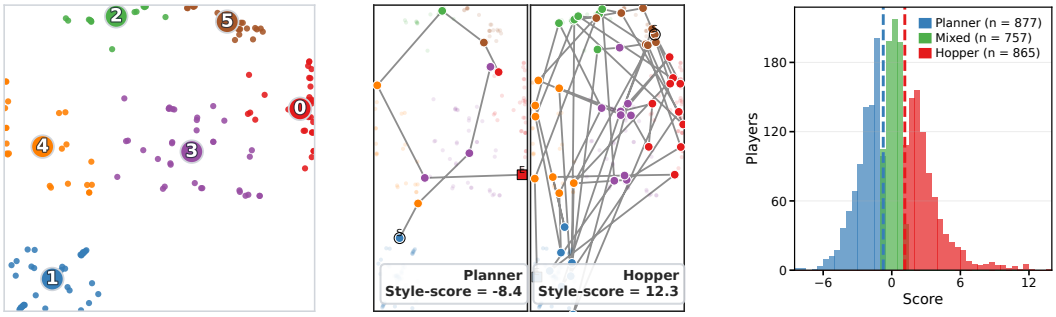
$$S_i = z(a_i) + z(r_i) - z(\ell_i) + z(d_i), \tag{1}$$

where $z(\cdot)$ denotes standardization across players. Higher scores indicate more frequent zone switching, more returns to previous zones, larger spatial transitions, and shorter same-zone runs; lower scores indicate locally persistent *Zone Planners*, while higher scores indicate *Zone Hoppers*. We label the bottom 35% of S_i as Zone Planners, the top 35% as Zone Hoppers, and treat the middle 30% as *Mixed*/ambiguous; the primary binary task excludes this middle group.

4.2 Controlled Grid-World Simulation: Mechanism Check

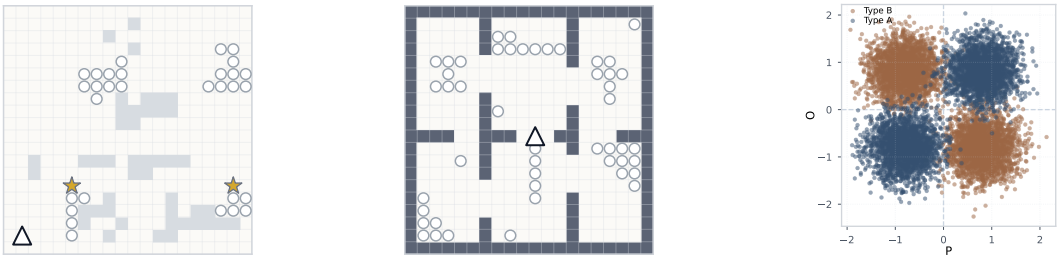
Simulation Design: We use a controlled grid-world simulation in which each synthetic agent completes two related tasks: an outdoor *foraging* task and an indoor *cleaning* task, shown in Figure 4. In both tasks, agents complete spatially distributed targets, and movement to a selected target follows shortest-path navigation. Agents therefore differ not in pathfinding ability, but in which target they choose next and how those choices are organized over time.

Known Latent Phenotypes: Each agent follows a simulated policy parameterized by two shared latent variables: patch persistence P_i, the tendency to remain within the same local region, and opportunistic reward seeking O_i, the tendency to prioritize salient or high-value targets. These variables influence target selection across both tasks, but with different task-specific strengths.



(a) (b) (c)

Figure 3: Construction of navigation-style labels. (a) Fire Station subtasks are grouped into six K-Means clusters (zones). (b) For each player, we convert the ordered subtask completions into a zone-completion trajectory; Zone Planners show more coherent local progression, whereas Zone Hoppers switch repeatedly across zones. (c) Players are labeled using the resulting navigation-style score: the bottom 35% are labeled Planners, the top 35% Hoppers, and the middle 30% Mixed/ambiguous.



(a) Foraging Task (b) Cleaning Task (c) Simulated Agents

Figure 4: Paired-task simulation setup and latent label structure. (a) Foraging environment: agents navigate a shared grid to collect food items (circles), including a small number of high-value targets (stars). (b) Cleaning environment: agents traverse a distinct grid layout to clear distributed mess targets organized into spatial patches. (c) The latent trait distribution used to generate agent types.

Ground-truth labels are assigned by a diagonal rule as shown in Figure 4c. Agents with same-sign drives, high P_i/high O_i or low P_i/low O_i, are labeled *Type A*. Agents with opposite-sign drives, high P_i/low O_i or low P_i/high O_i, are labeled *Type B*. Thus, the phenotype depends on the joint configuration of the latent variables. The tasks are designed to provide complementary evidence about this joint phenotype. In foraging, agents collect 32 food targets, including rare high-value items, making target choices more informative about O_i. In cleaning, agents clean 48 equal-valued targets arranged in clusters and strips, making choices more informative about P_i. At each decision point, agent i selects the next unfinished target j using a softmax over scores combining same-patch persistence, task-specific target value, shortest-path distance, and noise:

$$\text{score}_i^{(m)}(j) = \beta_P^{(m)} P_i \mathbf{I}\{j \in \text{ current patch}\} + \beta_O^{(m)} O_i V_j^{(m)} - \beta_D d(x_i, x_j) + \epsilon_j \qquad (2)$$

where m indexes the task, $V_j^{(m)}$ is target value (salience), $d(x_i, x_j)$ is shortest-path distance from the agent to target, and $\epsilon_j \sim \mathcal{N}(0, \sigma^2)$ is stochastic choice noise. The coefficients $\beta_P^{(m)}$ and $\beta_O^{(m)}$ control how strongly persistence and opportunism are expressed in each task. β_D is the shared distance-penalty coefficient, fixed at 0.05 in all simulation conditions. After selecting a target, the agent takes the shortest path and completes it. This design makes every single task only partially informative: foraging provides stronger evidence about O_i, while cleaning provides stronger evidence about P_i.

Table 1: Outcome-proxy sanity check after 20 minutes of game play. Std. diff. is computed as the Hopper–Planner mean difference divided by the pooled standard deviation.

Metric	Planner	Hopper	Std. diff.	AUC
Back Garden completion (%)	41.8 ± 19.6	43.2 ± 22.8	0.06	0.50
Back Garden subtasks completed	47.7 ± 17.5	49.4 ± 22.0	0.08	0.51
Back Garden subtasks/min	2.39 ± 0.88	2.47 ± 1.10	0.08	0.51
Bungalow completion (%)	39.0 ± 20.5	44.2 ± 24.2	0.23	0.55
Bungalow subtasks completed	22.1 ± 13.3	24.5 ± 14.4	0.17	0.54
Bungalow subtasks/min	1.10 ± 0.66	1.22 ± 0.72	0.17	0.54

5 Results

5.1 PowerWash: Predicting Future Navigation Style

We evaluate PLVM on the main application problem: predicting whether a player will later behave as a Planner or a Hopper in Fire Station from partial Back Garden and Bungalow traces. This is a pre-target-task prediction setting. Fire Station behavior is never observed as input at test time. We focus here on the binary Planner/Hopper task and report the harder three-class version, which includes the intermediate "Mixed" group, in Appendix D.3. We also replicate the findings with a new target task, Storybook House cleaning, with results reported in Appendix D.8.

Outcome-Proxy Sanity Check: Before evaluating process models, we tested whether the Fire Station Planner/Hopper label was recoverable from early-task outcomes. At a 20-minute observation cutoff, we computed completion percentage, number of completed subtasks, and subtasks completed per minute for Back Garden and Bungalow. Table 1 reports each metric by future Fire Station label, along with the standardized group difference and single-metric AUC. Across these simple outcome measures, future Planners and Hoppers show only weak separation. Back Garden outcomes are nearly indistinguishable across labels, with AUCs of 0.50–0.51. Bungalow outcomes show slightly larger differences, but remain weak predictors, with AUCs of 0.54–0.55. This suggests that the future navigation-style label is not merely a proxy for early completion or productivity.

Cross-Task Prediction Performance: We next evaluated whether partial Back Garden and Bungalow traces predict a player's later Fire Station navigation style. Without observing Fire Station behavior at test time, the model must predict labels derived from its full trace. Figure 5a shows accuracy across observation cutoffs, and Table 2 summarizes representative time points. Outcome baselines remain close to chance after 60–120 minutes, while single-task process models perform better, indicating that ordered traces contain signal beyond aggregate summaries. Among the three model families, PLVM performs best across cutoffs, reaching 65.6% at 60 minutes and 65.7% at 120 minutes, compared with 62.3% and 62.9% for the stronger single-task model; PLVM exceeds both single-task models at every cutoff (Table 2). Logistic regressions on the label's own navigation statistics, computed on the observed prefixes, reach a similar level (67.4 ± 3.4% at 120 min; p = 0.28 vs. PLVM). Shuffling event order before computing them causes a 12–19 pp drop in accuracy, to near the outcome baseline (51.0% at 120 min), locating the signal in how behavior is ordered (Appendix D.9). The attainable ceiling is modest: navigation style is a real but weak tendency across levels (ICC(3,1) = 0.314; Appendix D.6).

Latent Representation Analysis: To examine the learned person-level representation, we visualized the two-dimensional PLVM latent space for a representative PowerWash model in Figure 5c. Each point represents a player embedding obtained by fusing partial Back Garden and Bungalow traces, colored by the true Fire Station navigation-style label. The labels overlap substantially, consistent with future navigation style being a noisy behavioral phenotype rather than a perfectly separable trait. However, Zone Hoppers are shifted toward the upper-right region of the latent space, while Zone Planners are more concentrated toward the lower-left and middle regions. This suggests that PLVM learns predictive structure relevant to future navigation style.

Label Sensitivity: We further tested whether the PLVM advantage over the single-task process model depends on how the navigation-style score is composed, recomputing labels with each component removed in turn (Appendix D.5). The advantage remains positive under every alternative formulation but is significant only for the full composite, and is smallest without mean zone jump distance, so its magnitude depends on how the construct is defined.

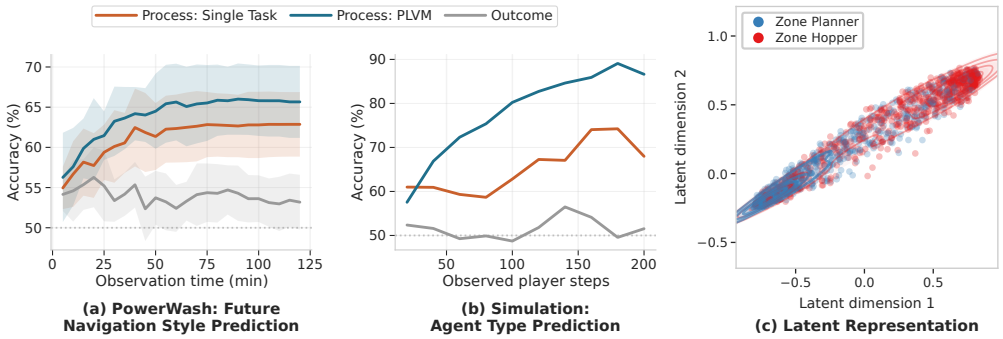


Figure 5: Behavioral style prediction. (a) PowerWash: predict a player's Fire Station strategy. (b) Simulations: predict ground-truth type. In (a) and (b), Single Task shows the stronger single-task model at full observation (Bungalow and cleaning, respectively). (c) A 2-D PLVM latent representation for a representative PowerWash model. Each point is a player. The groups overlap, as expected for a noisy future behavioral strategy, but show a modest shift in latent space (30-min cutoff; held-out linear probe AUROC = 0.731, silhouette = 0.157).

Table 2: Future navigation-style prediction accuracy at representative observation cutoffs (mean ± SD across 10 folds. **Bolded** values represent the best method). Majority class: 52.0 / 52.1 / 52.1%. The final column is the trapezoidal average of the accuracy–time curve over observation cutoffs from 5 to 120 minutes. Δ rows give paired differences in percentage points across shared folds, computed from unrounded accuracies, with Wilcoxon p-values in parentheses; bootstrap CIs and per-player McNemar tests are in Appendix D.2.

Method	20 min	60 min	120 min	Mean acc. (%)
Outcome	56.3 ± 3.7	52.4 ± 2.9	53.2 ± 3.4	54.0 ± 2.0
Single-task (Bungalow)	57.7 ± 4.0	62.3 ± 4.2	62.9 ± 4.0	61.4 ± 3.5
Single-task (Back Garden)	59.0 ± 3.0	61.4 ± 5.0	62.2 ± 4.6	60.6 ± 3.7
PLVM	$\mathbf{61.0 \pm 5.1}$	$\mathbf{65.6 \pm 4.8}$	$\mathbf{65.7 \pm 4.5}$	$\mathbf{64.1 \pm 3.7}$
Δ PLVM − Single (Bungalow)	$+3.3\ (0.050)$	$+3.3\ (0.010)$	$+2.8\ (0.017)$	$+2.7\ (0.020)$
Δ PLVM − Single (Back Garden)	$+2.0\ (0.193)$	$+4.2\ (0.020)$	$+3.4\ (0.049)$	$+3.5\ (0.006)$
Δ PLVM − Outcome	$+4.7\ (0.084)$	$+13.2\ (0.002)$	$+12.5\ (0.002)$	$+10.1\ (0.002)$

5.2 Simulation: Mechanism Check with Known Latent Phenotypes

Type recovery: Because the true latent process is unobserved in PowerWash, we use simulations as a mechanism check: agents have known behavioral types, and paired tasks reveal complementary evidence about those types. Figure 5b shows validation accuracy as a function of observed agent steps. PLVM receives matched partial windows from both foraging and cleaning, single-task process models receive one task, and outcome baselines receive only cumulative summaries such as completed targets, reward, or cleaned salience. PLVM performs best from 40 steps onward, while outcome baselines remain close to chance. Single-task process models improve with more observation but remain below PLVM because each task primarily exposes one component of the phenotype. How the evidence is combined matters as well: under a linear head, adding the second task gains only +2.2 pp (58.8%, below a single task under a nonlinear head at 68.0%), whereas under a nonlinear head it gains +18.6 pp. In PowerWash the same interaction is +1.3 pp (n.s.), so fusion rather than interaction capacity is the operative factor there (Appendix Table 12).

When does fusion help: We further varied the relationship between simulated tasks; full results are in Appendix Table 9. To compare conditions on a common scale, we apply a matched MLP head to either one task process representation or the concatenation of both, and measure the gain from combining tasks over the stronger single task. Combining tasks gains +18.9 pp when the simulated tasks are complementary (68.0% to 86.9%), +19.5 pp when each carries weak but shared signal, and 0.0 pp when they are redundant, where one task alone already reaches 89.4%. On PowerWash Fire Station at 120 minutes, the same matched-head measure gives +2.4 pp. PowerWash therefore sits

much nearer the redundant end of this axis, though unlike the redundant simulation, where one task alone was near ceiling, both PowerWash sources remain individually weak predictors. Cross-task fusion is thus most useful when source tasks provide complementary or noisy views of the same latent process, and this measure makes explicit where a given task pair falls on that axis.

6 Discussion & Conclusion

We introduced PLVM for early cross-task behavioral strategy prediction: using partial traces from earlier tasks to infer how an individual will organize behavior in a future task. Our results support two claims, one strong and one more modest. The primary finding is that future behavioral strategy is legible in process traces but largely invisible in aggregate outcomes. On the same observed prefixes, PLVM exceeded outcome baselines by 12–13 pp at 60 and 120 minutes in every fold, and the gap replicated on a second target level. Shuffling the observed events removes the signal from engineered process features, so the gap is attributable to ordering rather than to how much was observed or where it occurred. PLVM matches construct-informed references that apply the label's own definition to the source levels, recovering the signal without being told where it lies; the modest absolute accuracy is shared by every predictor we tested, consistent with the target's limited stability across levels (ICC = 0.31). The second and more modest finding is that combining process evidence across source tasks improves prediction further: PLVM exceeds both single-task models at every cutoff on both target levels, though the gain is small and its significance depends on the test. The simulation clarifies the boundary condition: fusion recovers a latent phenotype only when neither source task expresses it fully on its own. The matched-head measure places Back Garden and Bungalow toward the low-complementarity end of that range, which, together with the target's limited cross-level stability, is consistent with the size of the real-data gain and suggests where larger gains should be sought.

Adaptive systems typically personalize from what people have achieved, such as scores, completion rates, or progress. Our results show that these summaries can be blind to how people will organize behavior next: future Planners and Hoppers were indistinguishable by outcome, yet separable from process traces within the first hour of earlier tasks, before any behavior in the new task was observed. This allows a tutor, game, or human-AI partner to adapt at the start of a new task rather than after enough of it has unfolded.

Broader Impact: If used appropriately, process-level inference could help adaptive learning, game-based, and collaborative systems respond to how people approach a task rather than only to eventual scores or completion. Such systems could offer support or feedback earlier, although this study does not evaluate whether doing so improves learning, performance, or well-being. The same capability could be misused for unethical behavioral profiling, surveillance, or identification. In the present work, this risk is limited. The simulations use synthetic agents, and the PowerWash analysis is an offline evaluation on an existing research dataset; no predictions are deployed or used to make decisions about players. Future applications of PLVM to real-world behavioral data should require clear consent, transparent data collection, privacy protections, and limits on downstream use.

Limitations and Future Work: Three limits bound the interpretation of this work. First, the source encoders predict position change and the navigation-style label is itself a movement composite, so the learned representation may reflect a persistent low-level navigation habit rather than a broader person-level tendency. Second, the primary target is an operational label: a fixed, equally weighted composite binarized by excluding the middle 30%. The fusion advantage remains positive but loses significance when any single component is removed, and the three-class setting is more sensitive to the label definition (Appendix D.5), so the size of the fusion gain depends on how the construct is defined. Third, both target levels come from the same environment, so we demonstrate transfer across tasks within PowerWash rather than across structurally different domains. The simulation validates the mechanism under known latent structure, but its tasks are constructed so that each reveals one latent dimension and cannot establish that PowerWash shares it. Future work will test other interactive domains, train source encoders on non-positional objectives to separate transferable organization from persistent movement habit, compare frozen against end-to-end jointly trained encoders, validate inferred strategies against external measures, and evaluate whether they improve downstream interventions.

Acknowledgments

We acknowledge the support of the McDonnell International Scholars Academy and the Biswas Family Foundation. The views expressed in this paper are solely those of the authors and do not necessarily reflect those of the funders.

Competing interests. The authors declare no competing interests.

References

[1] Ella Anghel, Lale Khorramdel, and Matthias von Davier. The use of process data in large-scale assessments: A literature review. *Large-Scale Assessments in Education*, 12:13, 2024.

[2] Marcos Bella-Fernández, Manuel Suero Suñé, and Beatriz Gil-Gómez de Liaño. Foraging behavior in visual search: A review of theoretical and mathematical models in humans and animals. *Psychological Research*, 86:331–349, 2022.

[3] Sara Bunian, Alessandro Canossa, Randy Colvin, and Magy Seif El-Nasr. Modeling individual differences in game behavior using HMM. *Proceedings of the AAAI conference on Artificial Intelligence and Interactive Digital Entertainment*, 13(1):158–164, 2017.

[4] Frederick Callaway, Falk Lieder, Paul M Krueger, and Thomas L Griffiths. Mouselab-MDP: A new paradigm for tracing how people plan. In *The 3rd Multidisciplinary Conference on Reinforcement Learning and Decision Making*, 2017.

[5] Eric L Charnov. Optimal foraging, the marginal value theorem. *Theoretical Population Biology*, 9:129–136, 1976.

[6] Liam Collins, Hamed Hassani, Mahdi Soltanolkotabi, Aryan Mokhtari, and Sanjay Shakkottai. Provable multi-task representation learning by two-layer ReLU neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 9292–9345. PMLR, 2024.

[7] Naomi P. Friedman and Akira Miyake. Unity and diversity of executive functions: Individual differences as a window on cognitive structure. *Cortex*, 86:186–204, 2017.

[8] Hiroshi Higashi. Predicting human decision-making across task conditions via individuality transfer. *eLife*, 14:RP107163, 2026.

[9] Guy Hoffman, Tapomayukh Bhattacharjee, and Stefanos Nikolaidis. Inferring human intent and predicting human action in human–robot collaboration. *Annual Review of Control, Robotics, and Autonomous Systems*, 7:73–95, 2024.

[10] Yash Raj Jain, Frederick Callaway, Thomas L Griffiths, Peter Dayan, Ruiqi He, Paul M Krueger, and Falk Lieder. A computational process-tracing method for measuring people's planning strategies and how they change over time. *Behavior Research Methods*, 55:2037–2079, 2023.

[11] Robert Kasumba, Dom CP Marticorena, Anja Pahor, Geetha Ramani, Imani Goffney, Susanne M Jaeggi, Aaron R Seitz, Jacob R Gardner, and Dennis L Barbour. Distributional latent variable models with an application in active cognitive testing. *IEEE Transactions on Cognitive and Developmental Systems*, 17:1212–1222, 2025.

[12] Robert Kasumba, Guanghui Yu, Chien-Ju Ho, Sarah Keren, and William Yeoh. Goal recognition design for general behavioral agents using machine learning. *Transactions on Machine Learning Research*, 2025.

[13] Robert Kasumba, Zeyu Lu, Dom CP Marticorena, Mingyang Zhong, Paul Beggs, Anja Pahor, Geetha Ramani, Imani Goffney, Susanne M Jaeggi, Aaron R Seitz, Jacob R Gardner, and Dennis L Barbour. Bayesian distributional models of executive functioning. *Behavior Research Methods*, 58(11):310, 2026.

[14] Arsham Gholamzadeh Khoee, Yinan Yu, and Robert Feldt. Domain generalization through meta-learning: a survey. *Artificial Intelligence Review*, 57:285, 2024.

[15] Tómas Kristjánsson, Ian M Thornton, Andrey Chetverikov, and Árni Kristjánsson. Dynamics of visual attention revealed in foraging tasks. *Cognition*, 194:104032, 2020.

[16] Samuel Lippl and Jack Lindsey. Inductive biases of multi-task learning and finetuning: multiple regimes of feature reuse. *Advances in Neural Information Processing Systems*, 37:118745–118776, 2024.

[17] Dom CP Marticorena, Zeyu Lu, Chris Wissmann, Yash Agarwal, David Garrison, John M Zempel, and Dennis L Barbour. Immersive virtual games: winners for deep cognitive assessment. *arXiv preprint arXiv:2502.10290*, 2025.

[18] Reid McIlroy-Young, Yu Wang, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Detecting individual decision-making style: Exploring behavioral stylometry in chess. *Advances in Neural Information Processing Systems*, 34:24482–24497, 2021.

[19] Akira Miyake, Naomi P Friedman, Michael J Emerson, Alexander H Witzki, Amy Howerter, and Tor D Wager. The unity and diversity of executive functions and their contributions to complex "frontal lobe" tasks: A latent variable analysis. *Cognitive Psychology*, 41(1):49–100, 2000.

[20] Panagiotis D Paraschos and Dimitrios E Koulouriotis. Game difficulty adaptation and experience personalization: A literature review. *International Journal of Human–Computer Interaction*, 39 (1):1–22, 2023.

[21] Peter Pirolli and Stuart Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.

[22] Rolf Steyer, Axel Mayer, Christian Geiser, and David A Cole. A theory of states and traits—revised. *Annual Review of Clinical Psychology*, 11(1):71–98, 2015.

[23] Xueying Tang. A latent hidden Markov model for process data. *Psychometrika*, 89(1):205–240, 2024.

[24] Kurt VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017.

[26] Walter P Vispoel, Carrie A Morris, and Murat Kilinc. Applications of generalizability theory and their relations to classical test theory and structural equation modeling. *Psychological Methods*, 23(1):1–26, 2018.

[27] Matti Vuorre, Kristoffer Magnusson, Niklas Johannes, James Butlin, and Andrew K Przybylski. An intensive longitudinal dataset of in-game player behaviour and well-being in PowerWash Simulator. *Scientific Data*, 10:622, 2023.

[28] Jun Yu, Xiaokang Liu, Chongliang Luo, Jin Huang, Rong Zhou, Yixin Liu, Jie Hu, Jianmin Chen, Ke Zhang, Dazheng Zhang, et al. Multitask learning 1997–2024: Part I fundamentals. *Harvard Data Science Review*, 7(3), 2025.

Appendix

This appendix provides additional details needed to reproduce and interpret the experiments.

A Code and Data

The code required to reproduce the results is available at `https://github.com/NMIL230/nmil-p-plvm-neurips2026`. Instructions on how to access the data are provided in the README of the GitHub repository.

B PowerWash Dataset and Label Construction

We use three career-mode tasks from the PowerWash Simulator dataset: Back Garden, Bungalow, and Fire Station. Back Garden and Bungalow are used as observed source tasks, and Fire Station is used as the held-out future target task. All PowerWash models are evaluated on players with usable source-task traces and a valid Fire Station navigation-style label.

B.1 Cohort Construction

The PowerWash Simulator dataset contains 11,080 unique players. We focus on the career-mode sequence Back Garden $\rightarrow$ Bungalow $\rightarrow$ Fire Station because these tasks form a consistent progression, share the structure of spatial cleaning, and provide substantial within-player overlap. Table 3 summarizes the progression funnel.

Table 3: PowerWash career-mode progression funnel. Percentages are relative to the full dataset of 11,080 unique players.

Milestone	Players	% of total
Total players in dataset	11,080	100.0
Started any career job	10,347	93.4
Completed Back Garden	8,044	72.6
Completed Bungalow	6,618	59.7
Completed Fire Station	3,846	34.7
Completed all three tasks	3,794	34.2

The three-task cohort represents players who progressed substantially through career mode, which is necessary for within-player cross-task prediction. Importantly, this cohort is not fragmented: over 98% of players who completed Fire Station had also completed both Back Garden and Bungalow. Only 52 Fire Station completers lacked one of the two earlier completed tasks, likely reflecting edge cases such as FreePlay leakage or logging anomalies.

Table 4: Cross-task overlap among the PowerWash tasks used in this study.

Task pair	Players in common
Back Garden + Bungalow	6,583
Back Garden + Fire Station	3,809
Bungalow + Fire Station	3,815

For task-specific encoder training, we retained task attempts completed within two hours to remove unusually long or interrupted sessions. This yielded 6,012 eligible Back Garden players and 6,061 eligible Bungalow players. These are task-specific cohorts and are not required to contain exactly the same players.

For the downstream Fire Station prediction task, we required a scorable and sufficiently complete Fire Station attempt so that the Planner/Hopper label was reliable. Of 3,569 players with Fire Station attempt files, 2,541 had at least one scorable attempt with subtask-completion events, and 2,499 met the label criterion of at least 90% completion, corresponding to at least 165 of 183 unique

subtasks completed. Requiring these players to also have Back Garden and Bungalow source-task data produced the final PLVM cohort of 2,197 players.

Table 5: Cohort construction for encoder training and downstream Fire Station prediction.

Cohort/filter	Players
Eligible Back Garden players for encoder training	6,012
Eligible Bungalow players for encoder training	6,061
Players with Fire Station attempt files	3,569
Players with scorable Fire Station subtask-completion events	2,541
Players meeting Fire Station label criteria	2,499
Final PLVM cohort with source-task data and Fire Station label	2,197

These filters ensure that source-task representations are learned from usable traces and that downstream labels are computed from sufficiently complete Fire Station behavior. They are intended to support reliable within-player cross-task prediction rather than to select for a desired model outcome.

B.2 Fire Station Label Construction

Fire Station navigation-style labels are constructed from the ordered sequence of completed subtasks. We cluster Fire Station subtask locations into $K = 6$ spatial zones and convert each player's ordered completions into a zone sequence. For player i, we compute four statistics:

- zone switch rate a_i, the fraction of consecutive completions that move to a different zone;
- return rate r_i, the tendency to revisit previously visited zones;
- mean zone run length ℓ_i, the average number of consecutive completions within the same zone;
- mean zone jump distance d_i, the average spatial distance between successive completed zones.

The navigation-style score is

$$S_i = z(a_i) + z(r_i) - z(\ell_i) + z(d_i),$$

where $z(\cdot)$ denotes standardization across players. Players in the bottom 35% of S_i are labeled *Zone Planners*, players in the top 35% are labeled *Zone Hoppers*, and the middle 30% are treated as *Mixed*/ambiguous and excluded from the primary binary analysis.

C Model Inputs, Hyperparameters, and Encoder Performance

All downstream process models use the same frozen task-specific encoders. Single-task process models pass one task's frozen embedding to an MLP head without compression; PLVM fuses both embeddings through g_ϕ into a low-dimensional latent. PLVM-1 to PLVM-4 (Appendix D.4) vary only this latent dimension.

For the main binary PowerWash task, we report validation accuracy averaged across 10 cross-validation folds. For sanity checks, we also report single-metric AUCs to quantify whether simple outcome summaries separate future Planners and Hoppers.

C.1 Model Hyperparameters

Task-specific transformers are trained independently using within-task next-event prediction objectives and are frozen before training the PLVM fusion and decoder modules. For simulation, the next-event target is the next action or target choice. For PowerWash, the next-event target is the player's position change by the next subtask-completion event.

C.2 Task-Specific Encoder Performance

Task-specific encoders are not the final prediction models; they are used to learn within-task process representations before downstream fusion. In simulation, the task-specific transformers achieved

Table 6: Inputs available to each model family. Outcome baselines use aggregate summaries, whereas process models use ordered behavioral traces.

Experiment	Model family	Inputs
Simulation	Outcome baseline	Cumulative summaries up to cutoff, including completion count/percentage, total reward, high-value target counts/reward, and salience totals.
Simulation	Single-task process	Ordered behavioral trace from one simulated task only, encoded by the corresponding task-specific transformer.
Simulation	PLVM	Ordered behavioral traces from both simulated tasks at matched cutoffs, encoded by frozen task-specific transformers and fused downstream.
PowerWash	Outcome baseline	Aggregate summaries from the observed prefix only, including completion percentage, completed subtasks, subtasks per minute, recent-window counts, trend/slope, zero-rate, and participant running mean.
PowerWash	Single-task process	Ordered subtask-completion trace from one observed PowerWash task only, encoded by the corresponding task-specific transformer.
PowerWash	PLVM	Ordered subtask-completion traces from Back Garden and Bungalow at matched cutoffs, encoded by frozen task-specific transformers and fused downstream.

Table 7: Model hyperparameters used in the reported experiments.

Model	Hidden size	Layers	Heads	Context length	Learning rate
Simulation transformer	256	2	4	200 steps	0.001
PowerWash transformer	32	2	4	120 events	0.001
Simulation PLVM	256 encoder / 32 decoder	2 decoder	–	–	0.0001
PowerWash PLVM	16 encoder / 16 decoder	2 decoder	–	–	0.0001
Outcome baselines	–	–	–	–	tuned by validation

held-out next-action prediction accuracy of approximately 87–89%, indicating that they learned the within-task target-selection dynamics. In PowerWash, the encoders predicted the player's position change by the next subtask-completion event. Table 8 reports held-out performance for both domains.

Table 8: Held-out performance of the task-specific process encoders. For PowerWash, Metrics 1–3 are the RMSE of the predicted position change to the next subtask-completion event along x, y, and z respectively, in game-world coordinate units. For simulation, a single metric is reported: next-action prediction accuracy, given as the range across the foraging and cleaning encoders.

Domain/task	Prediction target	Metric 1	Metric 2	Metric 3
Simulation foraging/cleaning	Next action or target choice	87–89% accuracy		
PowerWash Back Garden	$\Delta x, \Delta y, \Delta z$ to next subtask	2.70	0.39	2.26
PowerWash Bungalow	$\Delta x, \Delta y, \Delta z$ to next subtask	3.46	0.72	2.78

D Additional Results

D.1 Simulation Complementarity Analysis

We ran an additional simulation analysis to test when cross-task fusion should help. The main simulation uses complementary tasks: foraging more strongly expresses opportunistic reward seeking, while cleaning more strongly expresses patch persistence. We compare this setting with two alternatives. In the redundant condition, both tasks reveal similar information, so a strong single-task model should already recover the label. In the weak-but-shared condition, each task contains noisy evidence about the same latent process, so fusion can help by aggregating weak signals.

Table 9: Simulation complementarity analysis. PLVM gain is PLVM accuracy minus the best single-task accuracy. Matched-head gain applies an identical MLP head to one task representation or to the concatenation of both, so the comparison isolates the value of the second task from architectural differences; this is the measure used in Section 5.2 and applied there to PowerWash. The two gains coincide in the complementary and weak conditions but come from different models.

Task relationship	Best single-task	PLVM	PLVM gain	Matched-head gain
Complementary tasks	68.0	86.6	+18.6	+18.9
Redundant tasks	89.4	88.4	−1.0	0.0
Weak but shared signal	56.6	74.8	+18.2	+19.5

The results support the intended mechanism. PLVM gives a large gain when tasks reveal complementary aspects of the phenotype and when each task contains weak but shared signal. When tasks are redundant, the best single-task model already performs well and fusion provides no additional benefit. This clarifies the scope of PLVM: cross-task fusion is useful when tasks provide different or noisy views of the same latent process, but it is not expected to help when one task already reveals the relevant structure.

D.2 Expanded model comparisons

Figures 6 and 7 report every model we evaluated, including the single-task models under both linear and nonlinear heads. Figure 5 in the main paper shows only the best-performing variant in each family at full observation: the outcome baseline using summaries from both tasks and the Bungalow single-task process model. Table 2 reports both single-task models, because their ordering changes with the cutoff: Back Garden is stronger below roughly 30 minutes before Bungalow overtakes it.

The ordering across families is stable and does not depend on which variant is plotted. Every process model exceeds every outcome model from roughly 20 minutes onward in PowerWash and across all step counts in simulation, and the outcome baselines remain close to chance. Within each head type, the two-task models sit above both single-task models.

The two domains differ sharply in how much fusion-head capacity matters. In PowerWash, linear and nonlinear heads perform similarly at every observation cutoff, with only a slight nonlinear advantage for cross-task fusion. In simulation, however, moving from a linear to a nonlinear head improves the stronger single-task model by roughly 11 points and the two-task model by roughly 28 points. The linear two-task model reaches only 58.8%, below the nonlinear single-task model at 68.0%. Observing both tasks by itself is therefore insufficient. When the target depends on the joint configuration of two latent drives, the second task's evidence cannot be fully exploited without modeling their interaction.

Robustness to the test and unit of analysis: Table 10 repeats the main comparisons reported in Table 2 under alternative paired tests. The process-over-outcome result holds under every test from 60 minutes onward: the bootstrap CI excludes zero by a wide margin, the paired t-test and the per-player McNemar test both give $p < 0.001$, and PLVM exceeds the outcome baseline in 10 of 10 folds. At 20 minutes the bootstrap CI still excludes zero and McNemar is significant ($p = 0.014$), but the fold-level tests are not (t-test $p = 0.068$; Wilcoxon $p = 0.084$). The fusion result is weaker and test-dependent at the earliest cutoff. Fold-level tests agree with the Wilcoxon results of Table 2 in direction and magnitude, although the percentile bootstrap is somewhat more liberal at $n = 10$, excluding zero in one case where Wilcoxon is borderline (Bungalow at 20 min, $p = 0.050$). Per-player McNemar tests agree: against Bungalow the advantage is significant at all three cutoffs ($p = 0.019$, 0.012, 0.030), and against Back Garden at 60 and 120 minutes ($p = 0.003$, 0.017) but not at 20 minutes ($p = 0.137$). The fusion advantage is therefore consistent in direction across tests, controls, and units of analysis, and significant under every test against both controls at 60 and 120 minutes; at 20 minutes its significance depends on the control and the test.

D.3 Three-Class PowerWash Prediction

The main PowerWash analysis uses the binary Planner/Hopper task, excluding the ambiguous middle 30%. We also evaluate a three-class setting that includes the Mixed group. As expected, this task is

Table 10: Robustness of the PLVM advantage to the test and the unit of analysis. Fold-level rows use the 10 cross-validation folds (paired t-test; 95% percentile bootstrap over folds, 20,000 resamples). Per-player rows pool held-out player predictions across folds ($n = 1498$ at 20 min, 1519 at 60 and 120 min) and use the exact McNemar test. McNemar is not reported for mean accuracy, which has no per-player correct/incorrect outcome.

	20 min	60 min	120 min	Mean acc. (%)
PLVM − Bungalow				
Δ (pp)	$+3.26$	$+3.29$	$+2.79$	$+2.70$
95% CI (folds)	$[0.73, 6.26]$	$[1.52, 4.89]$	$[0.97, 4.39]$	$[1.09, 4.27]$
t-test p	0.055	0.006	0.014	0.011
McNemar p	0.019	0.012	0.030	–
PLVM − Back Garden				
Δ (pp)	$+1.96$	$+4.23$	$+3.42$	$+3.46$
95% CI (folds)	$[-0.64, 4.68]$	$[1.77, 6.41]$	$[0.74, 5.76]$	$[1.77, 4.98]$
t-test p	0.206	0.008	0.032	0.003
McNemar p	0.137	0.003	0.017	–
PLVM − Outcome				
Δ (pp)	$+4.73$	$+13.23$	$+12.47$	$+10.12$
95% CI (folds)	$[0.64, 9.15]$	$[11.11, 15.21]$	$[11.07, 13.71]$	$[8.95, 11.35]$
t-test p	0.068	< 0.001	< 0.001	< 0.001
McNemar p	0.014	< 0.001	< 0.001	–

harder because the Mixed group is behaviorally intermediate rather than clearly separated. Figure 8 shows that performance decreases for all models, but PLVM remains numerically above single-task models, though not significantly (see the first row in Table 11).

D.4 PLVM Latent Dimension Sensitivity

We evaluated PLVM variants with different latent dimensionalities to test whether performance depended on a specific choice of latent size. Figures 9 and 10 show results for the binary Planner/Hopper task and the harder three-class Planner/Mixed/Hopper task. Across both settings, low-dimensional PLVM variants perform competitively, and the two-dimensional model provides the best overall trade-off between accuracy and interpretability. We therefore use PLVM-2 in the main analyses: it is simple enough to visualize directly, while still matching or exceeding the performance of higher-dimensional variants across most observation cutoffs.

D.5 Label Sensitivity Analysis

We tested sensitivity to the navigation-style score using a leave-one-component-out analysis. We removed mean zone jump distance, mean zone run length, return rate, or zone switch rate in turn, recomputed the labels using the same process as in subsection 4.1, and compared PLVM against the best single-task process baseline (Table 11). The 2-class advantage remains positive under every formulation but is significant only with all four components, and is smallest without jump distance.

D.6 How stable is navigation style?

Our headline real-data result is an accuracy against a behaviorally derived label rather than a ground-truth annotation. Interpreting it requires knowing how reliably the construct can be measured and what a trivial predictor already achieves.

Stability across levels: We score each level independently with the statistic that defines our target, median-split into Zone Planner / Zone Hopper, on the 2,197 players of the prediction cohort (Table 5). Treating the three levels as repeated measurements of the same player gives ICC(3,1)$= 0.314$ (95% CI [0.288, 0.342]) and a Fleiss' κ of 0.221; only 41.6% of players receive the same label on all three levels, against 25% expected by chance. Navigation style shows modest cross-level consistency: labels agree more often than expected by chance, but many players are classified differently across

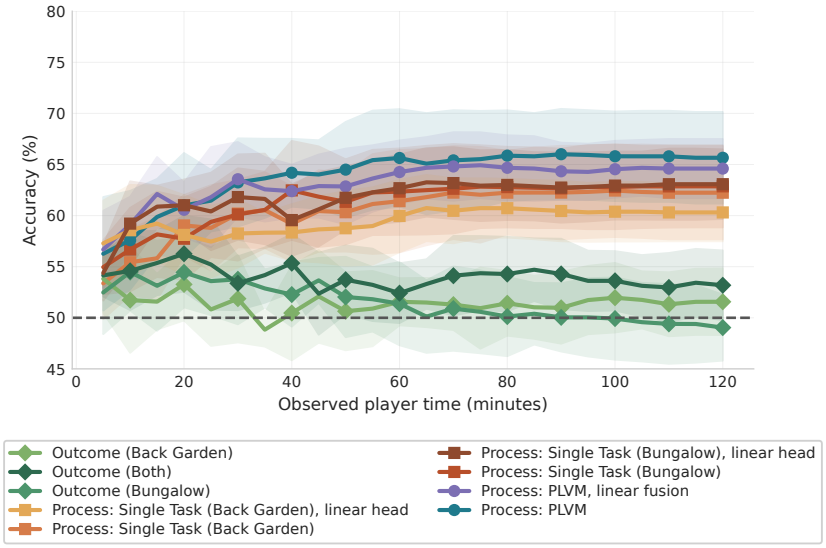


Figure 6: Expanded model comparison for the PowerWash Fire Station two-class prediction. Three model families are shown. Outcome baselines (greens) use aggregate summaries of the observed prefix from Back Garden, Bungalow, or both. Single-task models (oranges) use the ordered process trace from one source task, and two-task models (purple and blue) use the process traces from both; within each, the linear and nonlinear variants differ only in the head applied to the same frozen trace embeddings. All models observe the same prefix and differ only in how it is represented. Shaded bands give ± 1 standard deviation across the 10 cross-validation folds.

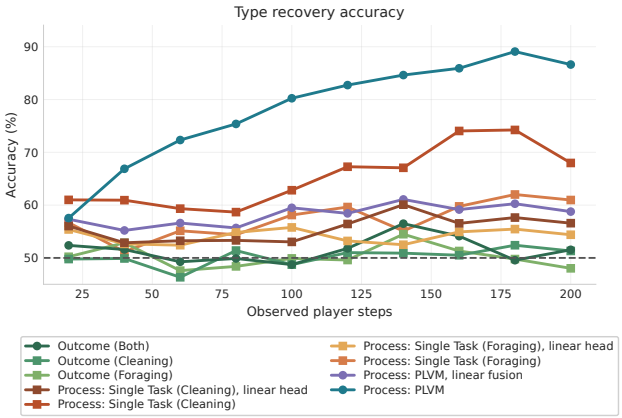


Figure 7: Expanded model comparison for recovery of the simulated behavioral type, using the same three model families as Figure 6. Outcome baselines (greens) use cumulative summaries from foraging, cleaning, or both. Single-task models (oranges) use the ordered process trace from one simulated task, and two-task models (purple and blue) use the process traces from both, each under a linear or nonlinear head over the same frozen trace embeddings.

levels. This stability may reflect a mixture of person-level, task-specific, and person × task effects rather than a context-invariant trait; latent state–trait and generalizability frameworks formalize this distinction [22, 26]. This limited consistency bounds how well one level's label can be predicted from the others.

Carrying the label forward: Assigning each player the label from Back Garden, scored on the same held-out players, folds, and target as PLVM, attains $64.5 \pm 5.5\%$, statistically indistinguishable from the PLVM ($65.7 \pm 4.5\%$; paired Wilcoxon p=0.32). This predictor is a binarized form of the navigation statistics in Appendix D.9, whose continuous version also matches the PLVM. Both

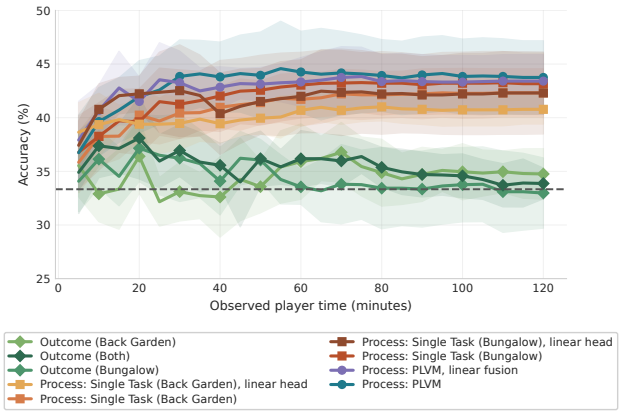


Figure 8: Three-class Fire Station navigation-style prediction, including Zone Planner, Mixed, and Zone Hopper labels. Performance decreases relative to the binary task because the Mixed group is less behaviorally separable.

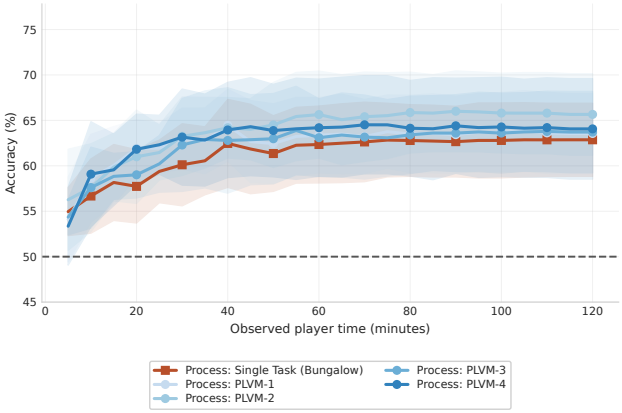


Figure 9: Latent-dimension sensitivity for the binary PowerWash Planner/Hopper prediction task. PLVM variants with low-dimensional latent states perform similarly, with PLVM-2 providing the strongest and most interpretable overall model.

confirm that the signal the PLVM recovers is the same order-dependent tendency that makes navigation style transferable across levels. Both baselines, however, compute the target's own statistic on the source levels, which presumes the target construct is known in advance and expressed the same way in earlier tasks. The PLVM assumes neither: its encoders are trained without the label, the same frozen encoders serve both target levels, and in simulation, where the target is a latent configuration with no directly computable analogue, it recovers the phenotype that single tasks cannot (Section 5.2).

D.7 Fusion Mechanism Ablation Analysis

Combining evidence from a second source task and representing how the two tasks interact are separable choices, and PLVM differs from the single-task baseline in both at once. To separate them we cross the two factors, applying a linear or a nonlinear head to either one task representation or the concatenation of both, over the same frozen embeddings. Table 12 reports the accuracies, the gain from the second task under each head, and the difference between those gains.

At full observation the two domains separate sharply. In simulation the second task is worth +2.2 pp under a linear head and +18.6 pp under a nonlinear one, an interaction of +16.4 pp; the two-task linear model reaches only 58.8%, below a single task under a nonlinear head at 68.0%. This follows from the label construction, since the simulated phenotype depends on the sign alignment of two

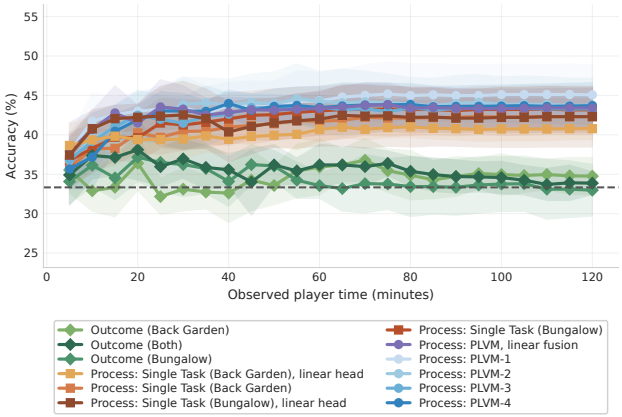


Figure 10: Latent-dimension sensitivity for the three-class PowerWash Planner/Mixed/Hopper prediction task. The three-class task is harder, but PLVM variants remain competitive with each other and outperform outcome-only baselines.

Table 11: Sensitivity of the PLVM fusion advantage to the label formulation. Each cell reports Δ accuracy in percentage points (PLVM $-$ single-task) at the given cutoff, with the corresponding p-value.

	2-class		3-class	
Label formulation	Δ @ 60 min	Δ @ 120 min	Δ @ 60 min	Δ @ 120 min
All components	$+3.3$ (p=.010)	$+2.8$ (p=.017)	$+1.2$ (p=.232)	$+0.6$ (p=.846)
Drop jump distance	$+0.7$ (p=.432)	$+0.8$ (p=.232)	$+0.7$ (p=.492)	$+0.4$ (p=.515)
Drop run length	$+1.1$ (p=.401)	$+0.9$ (p=.432)	0.0 (p=1.000)	-0.7 (p=.275)
Drop return rate	$+2.0$ (p=.232)	$+1.6$ (p=.110)	0.0 (p=.846)	0.0 (p=.767)
Drop switch rate	$+1.7$ (p=.110)	$+1.4$ (p=.131)	$+0.6$ (p=.770)	$+0.3$ (p=.953)

latent drives. In Fire Station the same interaction is +1.3 pp (n.s.) at 120 minutes, so cross-task fusion rather than interaction capacity is the operative factor there.

At the earliest cutoffs the nonlinear models are unstable relative to their linear counterparts and the interaction changes sign across domains; short prefixes give the higher-capacity models too little to fit, and we draw conclusions only from full observation.

D.8 Generalizability beyond the Fire Station task

To test whether the results are specific to Fire Station, we repeated the experiment with Storybook House as the held-out target. Storybook House also follows Back Garden and Bungalow in career mode, so the setting remains prospective: models observe only the two source levels and predict navigation style on Storybook House, labeled with the same procedure as Fire Station (Table 13).

Process versus outcome: The primary finding replicates. Outcome baselines stay near chance in both settings, while the PLVM exceeds them by $+9.5$ and $+13.4$ pp at 60 and 120 minutes in the 2-class setting and by $+8.7$ and $+10.5$ pp in the 3-class setting (p=0.002 at each).

Cross-task fusion: The PLVM remains above both single-task controls at every cutoff, but the strength of the evidence depends on which source task is withheld and on label granularity. In the 2-class setting the gain over the stronger control (Bungalow) is not significant ($+0.7$ pp, p=0.678 at 60 min; $+2.0$ pp, p=0.131 at 120 min), whereas the gain over the weaker control (Back Garden) is larger and reaches significance at the late cutoff ($+3.5$ pp, p=0.084 at 60 min; $+5.3$ pp, p=0.006 at 120 min). In the 3-class setting the gain is non-significant against Bungalow but significant against Back Garden at the two later cutoffs (Bungalow: $+1.7$ pp, p=0.105 and $+2.2$ pp, p=0.086; Back Garden: $+3.5$ pp, p=0.011 and $+4.4$ pp, p=0.004). We therefore read both results as directional: the gain is significant against Back Garden at the later cutoffs but not against the stronger Bungalow

Table 12: Ablation: fusion head capacity × number of source tasks, at an early and a late observation cutoff in each domain. Accuracy (%) under a linear and a nonlinear head applied to the same frozen trace embeddings. The lower panel reports the gain from adding the second source task under each head, and the difference between those gains, computed from unrounded accuracies. Wilcoxon paired signed-rank test p-values in parentheses. The nonlinear single-task and both-task rows correspond to the Bungalow single-task and PLVM rows of Table 2.

		Fire Station		Simulation	
Source tasks	Head	60 min	120 min	100 steps	200 steps
Single	Linear	62.7 ± 3.3	63.1 ± 3.4	53.0	56.6
	Nonlinear	62.3 ± 4.2	62.9 ± 4.0	62.8	68.0
Both (PLVM)	Linear	64.3 ± 3.1	64.6 ± 2.9	59.5	58.8
	Nonlinear	65.6 ± 4.8	65.7 ± 4.5	80.2	86.6
Δ from second task, linear head		$+1.6$ (0.139)	$+1.5$ (0.173)	$+6.5$	$+2.2$
Δ from second task, nonlinear head		$+3.3$ (0.010)	$+2.8$ (0.017)	$+17.4$	$+18.6$
Interaction (nonlinear − linear gain)		$+1.7$ (0.322)	$+1.3$ (0.260)	$+11.0$	$+16.4$

Table 13: Future navigation-style prediction with Storybook House as the held-out target (mean ± SD across 10 folds. **Bolded** values represent the best method). Linear fusion applies a linear head to the same concatenated frozen embeddings as the PLVM. Both single-task controls are reported, as the two source tasks differ in standalone predictive strength. Δ rows give paired differences in percentage points (computed from unrounded accuracies) with Wilcoxon p-values in parentheses.

	Method	20 min	60 min	120 min
2-class (%)	Outcome	56.6 ± 3.3	56.1 ± 3.4	53.8 ± 2.9
	Single-task (Bungalow)	57.9 ± 2.9	64.9 ± 4.8	65.1 ± 4.7
	Single-task (Back Garden)	57.4 ± 3.0	62.1 ± 4.6	61.8 ± 4.2
	Linear fusion	$\mathbf{62.3 \pm 2.5}$	65.4 ± 4.4	65.6 ± 4.1
	PLVM	59.1 ± 4.0	$\mathbf{65.6 \pm 2.2}$	$\mathbf{67.1 \pm 2.1}$
Δ (pp)	PLVM − Single (Bungalow)	$+1.2$ (0.432)	$+0.7$ (0.678)	$+2.0$ (0.131)
	PLVM − Single (Back Garden)	$+1.7$ (0.322)	$+3.5$ (0.084)	$+5.3$ (0.006)
	PLVM − Linear	-3.2 (0.027)	$+0.2$ (1.000)	$+1.6$ (0.123)
	PLVM − Outcome	$+2.5$ (0.193)	$+9.5$ (0.002)	$+13.4$ (0.002)
3-class (%)	Outcome	38.7 ± 4.8	38.0 ± 3.3	36.9 ± 3.3
	Single-task (Bungalow)	39.9 ± 2.5	45.0 ± 2.4	45.1 ± 2.0
	Single-task (Back Garden)	40.7 ± 3.2	43.2 ± 2.8	42.9 ± 2.5
	Linear fusion	$\mathbf{42.3 \pm 1.7}$	46.1 ± 2.0	46.2 ± 2.4
	PLVM	41.2 ± 3.2	$\mathbf{46.7 \pm 2.5}$	$\mathbf{47.3 \pm 3.0}$
Δ (pp)	PLVM − Single (Bungalow)	$+1.3$ (0.160)	$+1.7$ (0.105)	$+2.2$ (0.086)
	PLVM − Single (Back Garden)	$+0.6$ (0.846)	$+3.5$ (0.011)	$+4.4$ (0.004)
	PLVM − Linear	-1.0 (0.492)	$+0.6$ (0.260)	$+1.2$ (0.066)
	PLVM − Outcome	$+2.5$ (0.232)	$+8.7$ (0.002)	$+10.5$ (0.002)

control. The two source tasks differ in standalone strength here (65.1 vs. 61.8 at 120 min, 2-class). As in Table 2, Table 13 reports both.

Fusion head: As on Fire Station (Table 12), the nonlinear head is numerically ahead of linear fusion at the later cutoffs on Storybook House: $+1.6$ pp at 120 minutes in the 2-class setting (p=0.123) and $+0.6$ and $+1.2$ pp at 60 and 120 minutes in the 3-class setting (p=0.260, 0.066). None of these differences is significant, and linear fusion over the same concatenated embeddings also exceeds both single-task models at 60 and 120 minutes on both targets, so the cross-task gain does not depend on the capacity of the fusion head. At 20 minutes the linear head is ahead (-3.2 pp, p=0.027; -1.0 pp, n.s.), consistent with the instability of higher-capacity heads on short prefixes noted in Appendix D.7.

D.9 Feature-ladder construction and shuffle control

Table 14: Feature ladder and shuffle control (accuracy %, mean ± SD across 10 folds). Each row adds the named block to the previous one. Δ rows give paired differences in percentage points with Wilcoxon p-values in parentheses.

Features	20 min	60 min	120 min
+ Zone composition (order-free)	55.0 ± 2.7	48.1 ± 3.6	50.7 ± 2.8
+ Navigation statistics (ordered)	62.5 ± 5.7	66.5 ± 3.4	67.4 ± 3.4
+ Navigation statistics (shuffled)	58.2 ± 3.8	49.5 ± 1.4	51.0 ± 1.5
PLVM (reference)	61.0 ± 5.1	65.6 ± 4.8	65.7 ± 4.5
Δ PLVM − Nav. stats (ordered)	$-1.5\,(0.56)$	$-0.8\,(0.51)$	$-1.8\,(0.28)$
Δ Ordered − shuffled	$+4.3\,(0.010)$	$+16.9\,(0.002)$	$+16.4\,(0.002)$

Table 14 reports logistic regressions fit on the same prefixes, folds, and players as the PLVM. Per source task, the base features are the number of completed subtasks and level progression (a minimal subset of the outcome-baseline features in Table 6), and each row adds a block to the previous one. The zone-composition row adds the fraction of completions in each of the K=6 zones (order-free); the navigation-statistics row further adds the four quantities that define the target label (Appendix B.2), computed on the observed prefix (ordered). Zones are fit by k-means on each level's subtask coordinates, never on player data. The *shuffled* condition permutes each prefix's zone sequence before computing the navigation statistics and refits the model (averaged over 10 permutations); the base and composition features are unchanged by construction, so any loss is due to ordering alone. Ladder models are fit separately at each cutoff, whereas the PLVM is a single model across cutoffs.

The signal is in ordering: Ordering adds $+4.3$ pp at 20 min (p=0.010) and $+16.9$ and $+16.4$ pp at 60 and 120 min (p=0.002). The effect is smaller early because zone composition still varies within a level. Once a level is finished, each subtask has been completed exactly once, so composition converges to the level's fixed zone sizes and only ordering distinguishes players; with no distinguishing signal, the order-free and shuffled models fall to roughly the outcome-baseline level.

Relation to the PLVM: The ordered navigation statistics are statistically tied with the PLVM at every cutoff ($p \geq 0.28$). They act as a construct-informed reference rather than a competing baseline: they apply the target label's own definition to the source levels, which presumes the construct is known in advance and expressed the same way in earlier tasks. The PLVM matches this reference without access to the definition, using encoders trained without the label. This matters for targets that have no source-level counterpart: in simulation, where the phenotype is a latent configuration that cannot be computed from either task, no analogous engineered feature exists and outcome summaries remain near chance, while the PLVM recovers the phenotype (Figure 7). Real-world targets without an engineered counterpart are the natural next test.

E Simulation Details

This section provides the complete specification of the paired-task simulation. The class-conditional mixture distributions below specify how shared latent persistence and opportunism drives are sampled for each simulated type. Table 15 reports the target-selection-policy parameters for Equation 2 under the complementary, redundant, and weak-shared conditions.

Latent variables are sampled from class-conditional mixtures:

$$\text{Type A}: \tfrac{1}{2}\mathcal{N}((0.8, 0.8), 0.35^2 I) + \tfrac{1}{2}\mathcal{N}((-0.8, -0.8), 0.35^2 I),$$
$$\text{Type B}: \tfrac{1}{2}\mathcal{N}((0.8, -0.8), 0.35^2 I) + \tfrac{1}{2}\mathcal{N}((-0.8, 0.8), 0.35^2 I).$$

Type A agents have same-sign persistence and opportunism drives, whereas Type B agents have opposite-sign drives.

At each decision point, an agent computes the probability of selecting unfinished target j as

$$p^{(m)}(j) = \text{softmax}_j\big(\text{score}^{(m)}(j)/\tau\big)$$

where $\tau > 0$ is the softmax temperature.

Table 15: Simulation parameters used in the paired-task grid-world environment. The complementary condition is the main simulation; the redundant and weak-shared conditions are used in Table 9.

Parameter	Complementary	Redundant	Weak shared
Number of agents	8,000	8,000	8,000
Grid size	20×20	20×20	20×20
Foraging targets	32	32	32
Cleaning targets	48	48	48
High-value food items	6	6	6
Low-value food reward	1.0	1.0	1.0
High-value food reward	3.0	3.0	3.0
Cleaning target reward	1.0	1.0	1.0
Movement cost	0.01	0.01	0.01
β_P^{forage}	0.35	1.4	0.2
β_O^{forage}	2.8	1.6	1.5
β_P^{clean}	1.8	1.4	1.0
β_O^{clean}	0.6	1.6	0.35
β_D	0.05	0.05	0.05
Noise SD (σ)	0.5	0.5	0.8
τ	1.0	1.0	1.0
Observation cutoffs	20, 40, …, 200	20, 40, …, 200	20, 40, …, 200

F Computation Details

All experiments were run on a computing cluster with 40 CPU cores, an Intel Xeon Gold 6148 CPU at 2.40GHz, one NVIDIA Tesla V100 SXM2 GPU with 32GB memory, and up to 80GB system memory. Experiments used Python 3.9, PyTorch for model training, and NumPy/pandas for data processing.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims are stated in the Abstract and Introduction and are evaluated empirically in Sections 4–5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in the "Limitations and Future Work" paragraph in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions.
   - The authors should reflect on the scope of the claims made.
   - The authors should reflect on the factors that influence the performance of the approach.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present theoretical results or formal proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material.
   - Any informal proof provided in the core of the paper should be complemented by formal proofs.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper?

   Answer: [Yes]

   Justification: The experimental setup, model architecture, evaluation protocol, datasets, labels, and hyperparameters are described in Sections 3–5 and Appendices B–F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Reproducibility can be accomplished in various ways, including detailed instructions, released code/data, hosted models, or checkpoints.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code used to generate the experiments and figures is publicly available at `https://github.com/NMIL230/nmil-p-plvm-neurips2026`, with instructions to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines.
   - While release of code and data is encouraged, [No] is acceptable with proper justification.
   - The instructions should contain the exact command and environment needed to reproduce the results.
   - The authors should provide instructions on data access and preparation.
   - The authors should provide scripts to reproduce experimental results for the method and baselines.
   - At submission time, anonymized versions should be released if applicable.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details necessary to understand the results?

   Answer: [Yes]

   Justification: Training objectives, observation cutoffs, model inputs, cross-validation, and hyperparameters are described in Sections 3–4 and Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail necessary to appreciate the results.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Tables and figures report mean accuracy with standard deviations across folds; key comparisons report paired Wilcoxon tests, with bootstrap CIs and McNemar tests in Appendix D.2.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests.
- The factors of variability that the error bars capture should be clearly stated.
- The method for calculating the error bars should be explained.
- The assumptions made should be given.
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- If error bars are reported in tables or plots, the authors should explain how they were calculated.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources, including CPU, GPU, memory, and software environment, are reported in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for individual runs and total compute.
   - The paper should disclose whether the full research project required more compute than the reported experiments.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses synthetic simulations and an existing research dataset, with misuse and privacy risks discussed in the Broader Impact paragraph.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 6 discusses positive uses for adaptive systems and potential harms such as unethical profiling, surveillance, and identification.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include malicious or unintended uses, fairness considerations, privacy considerations, and security considerations.
- The authors should consider possible harms from intended use, incorrect results, and misuse.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse?

    Answer: [N/A]

    Justification: The paper does not release a high-risk pretrained model.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards.
    - Datasets scraped from the Internet could pose safety risks.
    - We encourage authors to make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets used in the paper properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The PowerWash Simulator dataset is cited in the main text and references; the supplemental code is released under the MIT License, and any shared derived data are released under the same CC0 license as the original dataset.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and include a URL if possible.
    - The name of the license should be included for each asset.
    - For scraped data, the copyright and terms of service should be provided.
    - If assets are released, license and copyright information should be provided.
    - For existing datasets that are re-packaged, both original and derived licenses should be provided.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate dataset/code/model details using structured templates.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets if applicable.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation?

Answer: [N/A]

Justification: The paper analyzes an existing PowerWash Simulator gameplay telemetry dataset and does not conduct new crowdsourcing or human-subject data collection.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed, and whether IRB approvals or equivalent review were obtained?

Answer: [N/A]

Justification: The paper does not conduct new human-subject experiments and instead uses an existing research dataset.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- IRB approval or equivalent may be required for human subjects research.
- Authors are expected to adhere to the NeurIPS Code of Ethics and institutional guidelines.
- For initial submissions, do not include information that would break anonymity.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the core methodology.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to the NeurIPS LLM policy for what should or should not be described.