Hierarchical Regime-Conditioned Dynamics for Spatiotemporal Graphs

Jeremy Baffou
EPFL
jeremy.baffou@epfl.ch

Pascal Frossard
EPFL
pascal.frossard@epfl.ch

Dorina Thanou
EPFL
dorina.thanou@epfl.ch

Abstract

Forecasting the behavior of spatio-temporal systems often requires more than accurate predictions: models should uncover the dynamical modes governing future evolution. Existing temporal graph and latent dynamical models achieve strong forecasting performance, but their latent representations often entangle multiple dynamical patterns, limiting interpretability. Conversely, regime-inference methods explicitly model such modes but typically do not support forecasting over graph-structured systems. We introduce HERON, a self-supervised regime-aware generative framework that unifies graph-structured trajectory generation with regime identification. HERON factorizes the latent state into discrete system-level regimes, capturing global dynamical modes, and continuous entity-level dynamics, yielding a modular regime interface compatible with existing spatio-temporal frameworks. Crucially, the inferred regime actively conditions system dynamics, enabling regime tracking and controlled simulations under alternative dynamical modes. Without regime annotations, HERON jointly learns forecasts and regime dynamics in a self-supervised manner, recovering meaningful modes across controlled synthetic and real-world datasets while maintaining competitive forecasting performance. These results open a path toward interpretable, controllable generative modeling of spatio-temporal dynamics.

1 Introduction

Accurate forecasting of the evolution of dynamical systems is essential in a wide range of applications, including healthcare [Chen et al., 2023, Liu et al., 2025, Shukla and Marlin, 2020], finance [Fan and Shen, 2024, Li et al., 2024a, Qian et al., 2024], and autonomous motion systems [Tang and Salakhutdinov, 2019, Girgis et al., 2021]. However, when similar observations arise from different underlying mechanisms, predictive accuracy alone is insufficient to characterize the system's dynamics. For example, two patients may display similar tumor growth patterns while belonging to different disease phases, requiring distinct monitoring strategies and interventions [Basler et al., 2020, Gabryś et al., 2022]. In such cases, identifying the underlying dynamical mode provides the context necessary to interpret, monitor, and adapt the forecast. There is therefore a need for models that not only forecast future outcomes, but also track the dynamical patterns that generate them. Such patterns can often be represented as discrete dynamical regimes such as disease phases [Höglinger et al., 2024, Persigehl et al., 2020] or market conditions [Oliveira et al., 2025, Chen et al., 2022], providing an interpretable abstraction of otherwise complex interaction-driven processes. However, existing approaches typically address regime inference [Linderman et al., 2016, Fox et al., 2008] and trajectory generation [Sestak et al., 2025, Xu et al., 2022, Janner et al., 2022] in isolation, leading to a lack of unified frameworks that can simultaneously identify regimes and generate regime-aware dynamics.

In many spatiotemporal settings like multi-agent motion or biological networks, dynamics arise from interactions between entities rather than from independent trajectories, motivating models that

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

explicitly couple temporal evolution with relational structure. Despite recent progress in modeling relational dynamics, existing approaches remain primarily focused on prediction rather than explicit regime characterization. Temporal graph neural networks [Souza et al., 2022, Longa et al., 2023, Huang et al., 2023, You et al., 2022, Pareja et al., 2020, Rossi et al., 2020] and graph state-space models [Li et al., 2024b, Behrouz and Hashemi, 2024] address this by combining relational aggregation with temporal memory, achieving strong results in graph-structured forecasting. However, these models are typically optimized for accurate rollouts, with latent states serving as predictive features rather than explicit descriptions of the underlying dynamical mode. This limits their applicability when the question is not only *where will the system go?*, but also *which dynamics govern its evolution?* Conversely, switching dynamical systems and regime-inference methods explicitly model discrete modes [Linderman et al., 2016, Fox et al., 2008, Caldarelli et al., 2022, Altamirano et al., 2023, Madrid Padilla et al., 2023], but are rarely designed to generate future trajectories over interacting entities. This creates a gap between forecasting models that encode regimes implicitly in continuous representations and regime-based models that lack graph-structured generative capabilities. This gap raises the central question of this work: can discrete regimes be learned as operational conditioning variables that track the current dynamical mode, govern future graph rollouts, and preserve the forecasting performance of latent trajectory models?

In this work, we introduce **HERON**[1], a self-supervised, regime-aware generative framework for spatio-temporal dynamics that models discrete, interpretable dynamical regimes as latent states governing interacting entities. Concretely, HERON treats each time step as a graph snapshot: given a context of past observations, it forecasts future trajectories over a target topology while inferring, at each step, which of K latent regimes governs the system evolution. HERON employs a hierarchical state representation in which a discrete system-level regime explicitly conditions the evolution of entity-level states. This formulation enables a bidirectional flow of information: local spatio-temporal observations are aggregated to infer a global regime, which in turn governs the generation of future trajectories. By coupling regime inference with autonomous rollout, HERON produces forecasts that are both accurate and dynamically actionable. Crucially, the inferred regime is not merely descriptive but it explicitly parametrizes the transition dynamics via a regime-conditioned mixture-of-experts generator, such that the active regime governs future trajectory generation. Across synthetic and real-world datasets with fixed and time-varying regimes, we show that HERON jointly discovers latent regimes and forecasts trajectories, achieving competitive predictive performance while recovering meaningful and interpretable structure. Furthermore, we show that by intervening on the inferred regimes, HERON can expose plausible alternative future trajectories, enabling decision making through the exploration of multiple possible futures rather than a single predicted trajectory. The key contributions of this work are:

- **Regime-aware framework for spatio-temporal dynamics.** We introduce a modular hierarchical latent-state formulation that separates system-level regimes from entity-level dynamics and integrates with existing spatio-temporal backbones.
- **Self-supervised regime-aware forecasting.** We show that jointly learning regimes and forecasts through our self-supervised objective recovers meaningful dynamical modes without annotations while maintaining competitive predictive performance.
- **Regime-conditioned rollouts.** We use regimes as active conditioning variables for generation, enabling regime tracking and controlled simulations of alternative futures under different dynamical modes rather than optimizing endpoint forecasts alone.

2 Problem formulation

Setup. We consider a spatio-temporal system of entities $\{n_i\} \subset \mathcal{N}$ observed at discrete time steps $t \in \{1, \dots, T\}$ with sampling interval Δt. To capture pairwise relations among entities, we model the system as a sequence of graph snapshots $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$, where $\mathcal{V}_t$ is a set of N_t entities, $\mathcal{E}_t \subseteq \mathcal{V}_t \times \mathcal{V}_t$ is a set of M_t relations, and $\mathbf{X}_t = (\mathbf{X}_t^V, \mathbf{X}_t^E)$ collects node and edge features, with $\mathbf{X}_t^V \in \mathbb{R}^{N_t \times d_V}$ and $\mathbf{X}_t^E \in \mathbb{R}^{M_t \times d_E}$. The topology $(\mathcal{V}_t, \mathcal{E}_t)$ is given and may vary with t. Node features $\mathbf{X}_t^V$ typically include spatial coordinates and other entity-level covariates.

We model the evolution of the system through a collection of K *regimes*, each encoding a distinct mechanism by which node and edge features evolve on a given topology. Formally, each regime is

[1] Code is available at `https://github.com/JerBaf/HERON`.

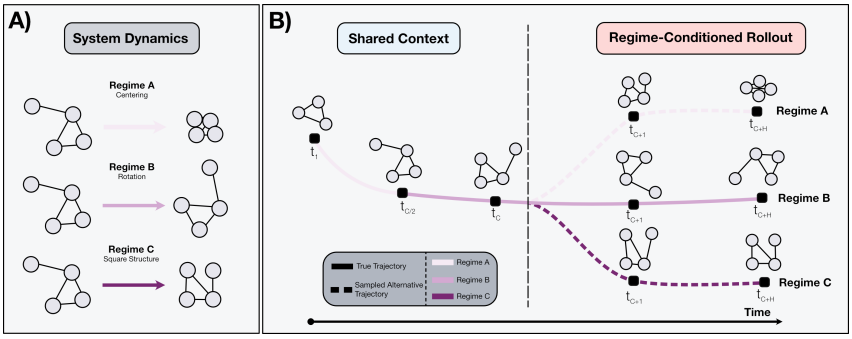


Figure 1: **Example of regime-conditioned graph rollouts. A)** The system can evolve according to distinct dynamics referred to as regimes. **B)** Given an observed graph sequence acting as *context*, we forecast future node features while inferring the latent regime trajectory that conditions the *rollout*. The regime exposes which dynamical mode drives the predictions and allows alternative regime-aware futures beyond forecasting alone.

represented by a function f_k that maps the graph history up to time t to the next feature state:

$$f_k : (\mathcal{V}_{1:t}, \mathcal{E}_{1:t}, \mathbf{X}_{1:t}) \mapsto \mathbf{X}_{t+1},$$

yielding the set $\{f_k\}_{k=1}^{K}$. We assume that the number of regimes K is known and that exactly one regime is active at each time step. When the number of regimes is not known a priori, we select K from the elbow of held-out predictive NLL. We detail sensitivity to K and extensions to overlapping regimes in Appendices 12.5.8, 12.5.9, and 12.5.4.

Objective. Given a *context* $(\mathcal{V}_{1:C}, \mathcal{E}_{1:C}, \mathbf{X}_{1:C})$ of length C and a target topology $(\mathcal{V}_\star, \mathcal{E}_\star)$, we aim to (i) learn a set of regimes $\{f_k\}_{k=1}^{K}$, and (ii) jointly infer the regime trajectory, i.e., the sequence of active regimes $(r_t)_{t=1}^{C+H}$ with $r_t \in \{1, \ldots, K\}$, while generating the future features $\mathbf{X}_{C+1:C+H}$ over a *horizon* of length H on the target topology. By default, $(\mathcal{V}_\star, \mathcal{E}_\star) = (\mathcal{V}_C, \mathcal{E}_C)$, i.e. the last observed topology. Figure 1 illustrates this setting, showing how future graph features are rolled out while jointly inferring the latent regime trajectory that conditions their evolution.

3 Unifying regime inference and forecasting on spatio-temporal graphs

Building on the spatio-temporal setting introduced in Section 2, we now present **HERON**, (HiErarchical Regime-cOnditioned dyNamics), a hierarchical generative graph state-space framework for joint regime inference and trajectory forecasting. HERON combines a discrete system-level latent variable capturing global regimes with continuous entity-level states for local dynamics. At each time step, the model infers the current regime from graph-temporal context, uses it to condition the dynamics, and predicts future system features. We detail the state representation, architecture, and training procedure below.

3.1 State representation

To capture both system-level regimes and entity-specific dynamics, we model a hierarchical latent state separating global and node-level processes. At each level, the state combines a deterministic recurrent component, which accumulates temporal evidence, with a stochastic component that captures uncertainty and promotes diverse trajectories. Formally, at time step t, the hierarchical state decomposes as follows:

System level. A deterministic recurrent state $h_t^G \in \mathbb{R}^{d_H}$ captures global temporal context, while a discrete stochastic variable r_t, parameterized by a probability distribution π_t over K regimes with realization represented as one-hot vector, encodes the active mode of collective behavior.

Entity level. Each node v_i maintains a deterministic recurrent state $h_t^i \in \mathbb{R}^{d_h}$ and a continuous stochastic state $z_t^i \in \mathbb{R}^{d_z}$, jointly encoding entity-specific dynamics.

The full hierarchical state at time t is thus:

$$h_t = \left(h_t^G,\ \{h_t^i\}_{i\in\mathcal{V}_t}\right), \qquad s_t = \left(r_t,\ \{z_t^i\}_{i\in\mathcal{V}_t}\right).$$

While recurrent state-space models (RSSMs) [Hafner et al., 2019, 2021, 2025] also combine deterministic and stochastic components, they use a single latent state. This limits their ability to model structured dynamics. In contrast, our hierarchy explicitly disentangles global regimes from entity-level dynamics, enabling structured modeling that cannot be captured by a single latent state.

3.2 Hierarchical regime-conditioned architecture

HERON follows a hierarchical encoder-decoder architecture as shown in Figure 2. At each time step, recurrent state updates summarize past dynamics at both the system and entity levels. These states are then combined with graph information to produce node- and graph-level embeddings. From these embeddings, the encoder infers a system-level regime together with regime-conditioned latent states at the entity level, which are in turn passed to the decoder to predict next-step node and edge features. We detail these components below following the progression from recurrent state updates to relational embeddings, hierarchical regime inference, and decoding. Figure 2 summarizes the complete HERON forward-pass.

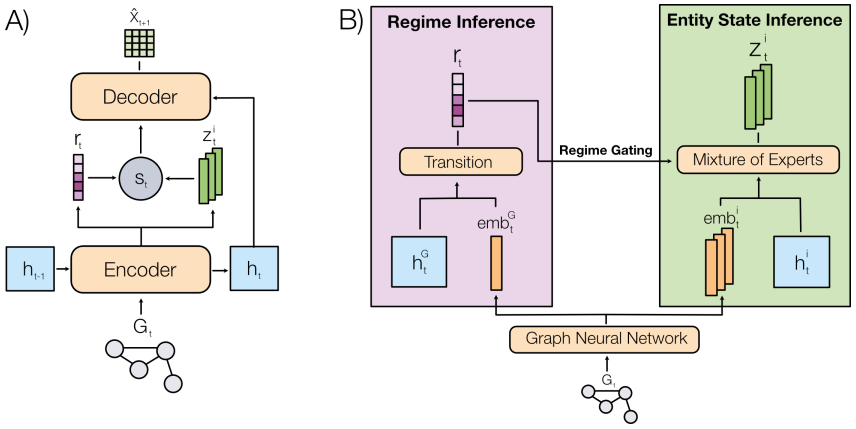


Figure 2: **Model architecture. A) Framework:** At each time step, the model first updates the deterministic recurrent state h_t. Conditioned on this state, the encoder infers the regime r_t and the entity latents z_t^i. Together with h_t, they are then used by the decoder to predict the next-step features X_{t+1}. **B) Encoder:** The encoder proceeds hierarchically. A GNN first extracts node- and graph-level embeddings (emb_t^i, emb_t^G), which are used to infer the regime r_t via a transition module. This regime then conditions a mixture-of-experts (MoE) to produce the entity-level latent states z_t^i.

3.2.1 Inference paths

HERON represents the stochastic state $s_t = \left(r_t,\ \{z_t^i\}_{i\in\mathcal{V}_t}\right)$ via two distributions over the same variables. The **posterior** $q_\phi(s_t \mid h_t, X_t, E_t)$ conditions on the current node and edge features and topology and is used over the context window. The **prior** $p_\psi(\hat{s}_t \mid h_t, E_t)$ conditions only on the deterministic state and topology and is used during rollout, when features are unavailable. We use hats throughout to denote prior quantities. The two paths share the recurrent state updates and the decoder; only the relational embeddings and the stochastic transition heads are duplicated as separate ϕ- and ψ-modules. This dual-path design enables latent rollouts to be generated directly from test-time inputs, rather than feeding predicted features back into a single encoder, which can lead to error accumulation. The training objective (Section 3.3) couples the two distributions through a Kullback–Leibler (KL) term, so that the prior remains consistent with observation-conditioned inference. We describe below the encoder from the posterior path; the prior follows the same structure with its own modules and without X_t as input. See the Appendix 12.1.3 for more details.

3.2.2 Encoder

The encoder updates the deterministic states and infers the stochastic variables of the current hierarchical state (h_t, s_t), as defined in Section 3.1, from two complementary sources of information: temporal memory carried by the recurrent states, and relational evidence extracted from the current graph snapshot when available. Concretely, the deterministic states h_t summarize past dynamics, while the embeddings $\mathrm{emb}_t = (\mathrm{emb}_t^i, \mathrm{emb}_t^G)$ provide relational summaries of the current graph. The encoder therefore first propagates the deterministic recurrent states forward in time, and then infers the stochastic latent variables through a hierarchical pipeline.

Recurrent state update. We first propagate temporal information forward through the deterministic part of the state. At both levels of the hierarchy, gated recurrent units (GRUs) [Chung et al., 2014] update the recurrent states from the previous latent state:

$$h_t^i = GRU_\theta^{\mathcal{V}}(h_{t-1}^i, z_{t-1}^i), \qquad h_t^G = GRU_\theta^G(h_{t-1}^G, r_{t-1}), \tag{1}$$

where $GRU_\theta^{\mathcal{V}}$ and GRU_θ^G denote two distinct GRU transition functions for the entity- and system-level recurrent states, respectively.

Relational embeddings. Conditioned on the recurrent states, a GNN[2] computes node embeddings (emb_t^i) that summarize the current local relational configuration. Then, an attention-based pooling operator [Lee et al., 2019] aggregates node embeddings into a graph-level representation (emb_t^G) :

$$\mathrm{emb}_t^i = \mathrm{GNN}_\phi\big(h_t^i, \mathbf{X}_t, \mathcal{E}_t\big), \qquad \mathrm{emb}_t^G = \mathrm{Pool}_\phi\big(\{\mathrm{emb}_t^i\}, \mathcal{E}_t\big). \tag{2}$$

Regime inference. We model the discrete latent regime variable r_t as a categorical distribution within a variational framework. Sampling is enabled via the Gumbel–Softmax reparameterization [Jang et al., 2017, Maddison et al., 2017] with a straight-through estimator [Bengio et al., 2013], which approximates hard regime assignments while preserving a soft representation for gradient-based optimization. The regime transitions are conditioned on the graph-level recurrent state and the corresponding embedding:

$$r_t \sim q_\phi\Big(r_t \mid h_t^G, \mathrm{emb}_t^G\Big). \tag{3}$$

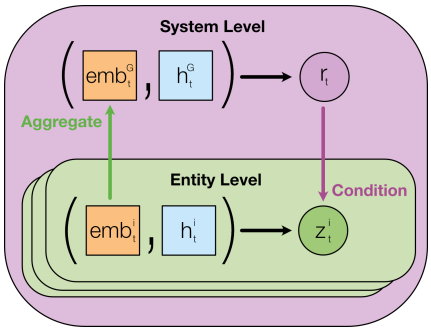


Figure 3: **Hierarchical state representation.** Entity-level embeddings are *aggregated* and combined with the global state h_t^G to infer a latent regime r_t that governs system-wide dynamics. This regime then *conditions* a MoE module to produce entity-level latent states z_t^i, linking global behavior and individual evolution.

Regime-conditioned entity state inference. The inferred regime explicitly modulates entity-level dynamics rather than acting as a passive system-level label. To enforce this, we use the regime to gate a mixture of experts (MoE) [Jacobs et al., 1991, Fedus et al., 2022] during entity-level stochastic state inference, with one expert assigned to each discrete regime dynamics. Each expert can then be seen as one of the dynamics modes f_k described in Section 2. This induces regime-specific dynamical mappings that generate the node-level latent states, tightly coupling collective behavior with individual dynamics:

$$z_t^i \sim q_\phi\big(z_t^i \mid h_t^i, \mathrm{emb}_t^i, r_t\big). \tag{4}$$

Prior path. Starting from the updated deterministic states h_t, the prior path embeddings are computed from recurrent states and topology alone:

$$\widehat{\mathrm{emb}}_t^{\,i} = \mathrm{GNN}_\psi\big(h_t^i, \mathcal{E}^\star\big), \qquad \widehat{\mathrm{emb}}_t^{\,G} = \mathrm{Pool}_\psi\Big(\{\widehat{\mathrm{emb}}_t^{\,i}\}_{i\in\mathcal{V}^\star}, \mathcal{E}^\star\Big). \tag{5}$$

[2]In this work, we instantiate the relational encoder with GNNs to model interactions among entities. However, HERON is agnostic to the specific relational architecture, meaning that any encoder that maps a set of interacting entities to relational representations, such as Set Transformers or point-cloud architectures, can be used in place of the GNN.

Then, the hierarchical latent states are sampled sequentially in two steps: the global regime is first sampled from the graph-level prior, after which each entity-level latent state is sampled from its node-level prior conditioned on the sampled regime, i.e.,

$$\hat{r}_t \sim p_\psi\Big(\hat{r}_t \mid h_t^G, \widehat{\mathrm{emb}}_t^{\,G}\Big), \quad \hat{z}_t^i \sim p_\psi\Big(\hat{z}_t^i \mid h_t^i, \widehat{\mathrm{emb}}_t^{\,i}, \hat{r}_t\Big), \qquad i \in \mathcal{V}^\star. \tag{6}$$

Overall, the hierarchical inference pipeline, illustrated in Fig. 3, aggregates local spatio-temporal signals into a global regime and propagates this information back to entity-level states, enabling explicit interaction between collective and individual dynamics. A summary of the algorithmic procedure is given in Algorithm 1.

Integration to existing frameworks. Importantly, this formulation is agnostic to the underlying temporal graph or relational backbone. It provides a lightweight, regime-aware interface for latent relational models with entity-level temporal states: node-level latent variables are reused as entity states, while a system-level discrete variable, conditions entity-level transitions or decoding. GraphSSM Li et al. [2024b] provides a natural example of this interface, as it models entity-level latent dynamics through graph-based relational mixing. HERON extends it with a graph-level regime inference module, regime-conditioned entity transitions, and the self-supervised objective of Section 3.3, equipping the base model with explicit regime tracking and regime-conditioned rollouts while preserving its original inductive biases. We detail the corresponding implementation in Appendix 12.5.7.

3.2.3 Decoder

Given the inferred hierarchical state (h_t, s_t), the decoder g_θ predicts the next set of features $\mathbf{X}_{t+1}$. We use a probabilistic decoder to parameterize the conditional uncertainty over future spatio-temporal dynamics. Concretely, a multi-layer perceptron (MLP) [Bengio et al., 2017] maps (h_t, s_t) to the parameters of a Gaussian predictive distribution with diagonal covariance:

$$\mathbf{X}_{t+1} \sim g_\theta\Big(\mathbf{X}_{t+1} \mid h_t, s_t\Big) = \mathcal{N}\Big(\mu_\theta\big(h_t, s_t\big),\, \mathrm{diag}\big(\sigma_\theta^2(h_t, s_t)\big)\Big),$$

where $\mu_\theta(\cdot)$ denotes the predicted mean and $\sigma_\theta^2(\cdot)$ the predicted marginal variances. The complete encoder–transition–decoder pipeline is summarized in Fig. 2.

3.3 Training procedure

HERON is trained end-to-end in a self-supervised manner, enabling regime discovery directly from spatio-temporal data without annotations. The training objective combines complementary components: accurate features forecasting, alignment between posterior and prior to support autonomous rollout, and structural regularization to prevent degenerate or uninformative regime assignments.

Prediction loss. The loss is defined as a negative Gaussian log-likelihood applied to both the posterior and prior paths, i.e.,

$$\mathcal{L}_{\mathrm{pred}} = -\ln g_\theta(\mathbf{X}_{t+1} \mid h_t, s_t) - \gamma \ln g_\theta(\mathbf{X}_{t+1} \mid h_t, \hat{s}_t).$$

This encourages accurate predictions and consistency between observed and autonomous rollouts.

Posterior-Prior alignment. We align the observation-conditioned posterior with the observation-free prior so that latent trajectories inferred during training can be reproduced at inference time. Following the KL balancing principle [Hafner et al., 2021], we decompose this KL objective into two asymmetric terms. The dynamics loss $\mathcal{L}_{\mathrm{dyn}}$ trains the prior to match a stopped-gradient posterior ($\mathrm{sg}(q_\phi)$), while the representation loss $\mathcal{L}_{\mathrm{rep}}$ regularizes the posterior toward a stopped-gradient prior ($\mathrm{sg}(p_\psi)$):

$$\mathcal{L}_{\mathrm{dyn}} = \mathrm{KL}[\mathrm{sg}(q_\phi(s_t \mid h_t, \mathbf{X}_t)) \,\|\, p_\psi(s_t \mid h_t)], \qquad \mathcal{L}_{\mathrm{rep}} = \mathrm{KL}[q_\phi(s_t \mid h_t, \mathbf{X}_t) \,\|\, \mathrm{sg}(p_\psi(s_t \mid h_t))].$$

Together, these terms train the prior without observation access while preventing the posterior from drifting into regions the prior cannot reproduce.

Structural loss. We regularize the regime pathway with three terms enforcing balanced regime usage, sharp assignments, and decorrelated experts. Let $Q_t^{(b)} = q_\phi(r_t \mid h_{t,b}^G, \mathrm{emb}_{t,b}^G)$ be the posterior regime distribution for sample b, and $z_{k,t}^i$ the output of expert k for entity i at time t. We define

$$\mathcal{L}_{\mathrm{struct}} = \mathcal{L}_{\mathrm{bal}} + \mathcal{L}_{\mathrm{ent}} + \mathcal{L}_{\mathrm{spec}}$$

where individual loss components can be writen as the following

$$\mathcal{L}_{\text{bal}} = \text{KL}\Big[\mathbb{E}_b[Q_t^{(b)}] \,\|\, \mathcal{U}(K)\Big], \quad \mathcal{L}_{\text{ent}} = \mathbb{E}_b\Big[\mathcal{H}[Q_t^{(b)}]\Big], \quad \mathcal{L}_{\text{spec}} = \sum_{i \in V_t} \sum_{j \neq k} \frac{\langle z_{j,t}^i, z_{k,t}^i \rangle^2}{\|z_{k,t}^i\|_2^2 + \epsilon}, \tag{7}$$

where $\epsilon > 0$ ensures numerical stability. $\mathcal{L}_{\text{bal}}$ matches the batch-averaged regime distribution to the uniform prior $\mathcal{U}(K)$, preventing collapse onto a few experts; $\mathcal{L}_{\text{ent}}$ minimizes per-instance regime entropy to encourage confident assignments compatible with straight-through gating; $\mathcal{L}_{\text{spec}}$ minimizes pairwise expert-output projections per entity, encouraging orthogonal expert representations.

Final formulation. The overall objective minimizes the four loss terms over the training window $W = C + H$:

$$\mathcal{L}(\theta, \phi, \psi) = \mathbb{E}_{q_\phi, p_\psi}\left[\frac{1}{W}\sum_{t=1}^{W}\Big(\beta_{\text{pred}}\mathcal{L}_{\text{pred}} + \beta_{\text{dyn}}\mathcal{L}_{\text{dyn}} + \beta_{\text{rep}}\mathcal{L}_{\text{rep}} + \beta_{\text{struct}}\mathcal{L}_{\text{struct}}\Big)\right].$$

This self-supervised objective enables regime discovery directly from spatio-temporal data while supporting stable rollout and structured modeling of latent dynamics.

4 Experimental setup

Baselines We evaluate HERON against baselines spanning diverse inductive biases for spatio-temporal trajectory prediction. We first include two persistence baselines, namely *Static* : $\hat{x}_{t+1} = x_t$ and *Velocity* : $\hat{x}_{t+1} = x_t + (x_t - x_{t-1})$. We include S4 Gu et al. [2022] as a non-relational state-space baseline. To assess whether performance gains follow from discrete switching alone or MoE conditioning, we compare against rSLDS Glaser et al. [2020], RED-SDS Ansari et al. [2021] and TESTAM Lee and Ko [2024]. To evaluate the role of relational modeling, we further consider LaM-SLidE Sestak et al. [2025] and GraphSSM Li et al. [2024b], which extend state-space models with relational inductive or graph-based biases, and EvolveGCN Pareja et al. [2020] as a temporal GNN baseline. Finally, we evaluate HERON on top of GraphSSM and compare its regime inference against a hidden Markov model (HMM) applied to the GraphSSM embeddings for regime retrieval. Implementation details are provided in Appendix 12.5.7.

Evaluation metrics We evaluate HERON along two axes: **1) Forecasting:** We measure forecasting quality using Average Displacement Error (ADE) Xu et al. [2023] and minimum Final Displacement Error (minFDE) Xu et al. [2022]. **2) Regime inference:** We measure the quality of the inferred regime assignments using Macro-F1 Manning et al. [2008]. We report test results, with training and validation scores are provided in Appendix 12.5.1.

Datasets To assess HERON's ability to capture dynamical regimes, relational structure, and feature evolution, we evaluate it on both synthetic and real-world multi-agent datasets. The synthetic datasets cover three controlled settings: linear dynamics, which removes interactions to isolate temporal modeling and regime recovery; Gompertz dynamics [Gompertz, 1825, Laird, 1964], which introduce non-linear feature evolution over an evolving topology; and network interactions Harush and Barzel [2017], which focus on interaction-driven processes to test regime recovery from relational structure. The real-world datasets, SportVU NBA [Linou et al., 2016, Yue et al., 2014] and AMASS [Mahmood et al., 2019a] with BABEL annotations [Punnakkal et al., 2021a], evaluate regime-aware forecasting in more complex settings. SportVU captures multi-agent trajectories with stable collective regimes defined by play outcomes, while for AMASS, we pool the original annotations into a coarser set of dynamical regimes Implementation details are provided in Appendix 12.4, and dataset properties summarized in Table 4.

5 Forecasting performance with regime conditioning

We first evaluate how regime conditioning supports autonomous trajectory forecasting. Table 1 shows that HERON remains competitive across all datasets and achieves the best ADE on the synthetic benchmarks, which cover complementary challenges: independent trajectories under a shared regime, non-linear dynamics with evolving topology, and interaction-driven network processes. In particular, strong performance on the network benchmark, where regimes are defined by relational propagation, indicates that HERON captures structured dynamics beyond simple temporal extrapolation.

Model	Synthetic			Real-world	
	Linear	Gompertz	Network	NBA	AMASS
Persistence (Static)	5.595	2.521	0.105	4.48/7.86	0.093/0.135
Persistence (Velocity)	2.896	2.906	0.058	2.55/5.75	0.129/0.271
S4 Gu et al. [2022]	4.072	4.307	0.162	2.09/5.11	0.422/0.616
rSLDS Glaser et al. [2020]	2.744	3.171	0.052	2.93/5.40	0.074/0.144
RED-SDS Ansari et al. [2021]	3.287	3.158	0.051	4.84/8.57	0.077/0.120
TESTAM Lee and Ko [2024]	2.975	2.661	<u>0.042</u>	2.70/4.52	<u>0.055</u>/0.094
EvolveGCN Pareja et al. [2020]	3.454	2.623	0.057	1.72/4.92	0.060/0.088
LaM-SLidE Sestak et al. [2025]	<u>2.077</u>	N/A	0.054	**0.73/1.32**	0.061/0.084
GraphSSM Li et al. [2024b]	2.537	2.178	0.049	1.65/4.59	0.056/0.082
GraphSSM+HERON Li et al. [2024b]	2.494	<u>2.042</u>	0.047	1.59/4.43	<u>0.055</u>/<u>0.080</u>
HERON	**1.937**	**1.938**	**0.038**	<u>0.81</u>/<u>1.49</u>	**0.047/0.066**

Table 1: **Regime-conditioned forecasting performance.** Test ADE is reported for generated datasets, and ADE/FDE for motion datasets. Best results are in bold and second-best underlined. N/A means that the method was not applicable.

On real-world datasets, HERON shows the intended forecasting–interpretability trade-off. On NBA, it remains close to LaM-SLidE, a strong latent trajectory model specialized for forecasting, while exposing an explicit system-level regime that can be tracked and evaluated. On AMASS, where motion regimes are fine-grained and may change within a sequence, HERON obtains the strongest ADE/FDE among the evaluated models.

Furthermore, capacity-matched ablations show that gains are not explained by expert count alone. As shown in section 7, removing regime conditioning or using fixed/random routing degrades forecasting and regime recovery, while removing the structural losses primarily weakens recovery. Overall, these results show that HERON preserves strong rollout accuracy while providing an explicit latent variable for modeling and analyzing dynamical regimes.

6 HERON recovers meaningful regime

Recovery performance We evaluate whether the discrete states learned by HERON correspond to meaningful dynamical regimes rather than arbitrary latent clusters. Regime labels are not used during training and are only introduced for evaluation. Table 2 shows strong alignment with ground-truth modes across synthetic datasets and NBA trajectories, supporting the role of the discrete state as an interpretable system-level variable. The network benchmark provides the most direct test of this claim. In this setting, regimes are defined by interaction mechanisms, rather than isolated node trajectories. HERON nevertheless recovers these modes with high Macro-F1 and AUC, indicating that the learned regime captures how information propagates through the graph. Together with the forecasting results, this shows that the regime variable is both predictive and aligned with the underlying dynamical mechanism.

Method	Synthetic			Real-world	
	Linear	Gompertz	Network	NBA	AMASS
Latent-HMM	0.945	0.897	<u>0.657</u>	<u>0.593</u>	<u>0.318</u>
rSLDS Glaser et al. [2020]	<u>0.953</u>	0.860	0.357	0.358	0.294
RED-SDS Ansari et al. [2021]	0.947	<u>0.926</u>	0.557	0.451	0.299
HERON	**0.961**	**0.927**	**0.913**	**0.891**	**0.412**

Table 2: **Self-supervised regime recovery.** Test macro-F1 quantifies how well the learned regimes align with ground-truth dynamical labels, which are used only for evaluation. Best results are in bold and second-best results are underlined.

The real-world datasets further evaluate fixed and time-varying regimes. On NBA, where regimes remain stable within each sequence, the learned states recover meaningful play patterns: the high macro-F1 shows that the model distinguishes rebound from scoring scenarios, which induce distinct motion patterns such as counterattacks and throw-ins. AMASS presents a more challenging setting, as action regimes may change within a sequence and the BABEL annotations provide only coarse proxies for the underlying motion dynamics reflected in the joint trajectories. Our extended analysis shows that 38% of regime errors occur within five frames of a label transition, while macro-F1 increases from 0.412 to 0.566 on stable segments. This suggests that smooth transitions and mismatch between semantic lables and trajectory dynamics account for part, but not all, of the remaining errors (Appendix 12.5.5). Nevertheless, the learned states support strong forecasting and produce coherent regime-conditioned rollouts.

Interventional analysis of regimes Finally, we probe the functional role of the regimes by intervening on the generator. Given the same context and recurrent state, we clamp the regime assignment to different experts and compare the resulting rollouts. Figure 4 shows that cchanging the clamped regime produces qualitatively distinct and recognizable motion patterns, such as walk or turn motions, rather than perturbing a single trajectory. Quantitatively, the matched regime yields the lowest ADE across datasets: clamping to an alternative regime increases ADE by 15–83% on Network and 16–23% on NBA, while physical and kinematic validity remain comparable under alternative clamps 12.5.2. These results support the central interpretation of HERON: the learned regime is not only recoverable from observed data, but also acts as an explicit conditioning variable for generating and probing alternative dynamical modes.

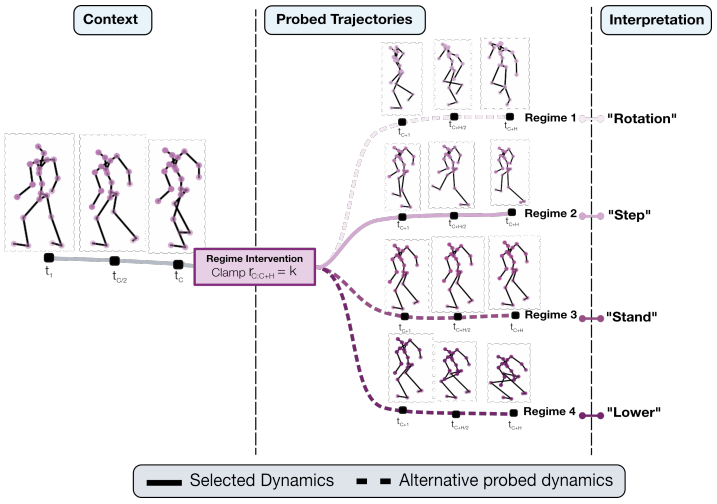


Figure 4: **Regime-conditioned rollout probes.** From a shared context, we clamp regimes during rollout. Dashed lines show regime-specific trajectories; the solid line is the observed future. Regime names are assigned post hoc from dominant motion patterns.

7 Component ablations

To determine whether HERON's performance gains arise from regime-aware modeling rather than additional capacity, we ablate the regime variable, its role in conditioning generation, the learned regime assignment, and the structural objectives. We evaluate both forecasting and regime recovery performances, and Table 3 reports test-set results under the same evaluation protocol as the main experiments.

The capacity-matched no-regime variant (A1) retains the same number and sizes of expert networks but averages their outputs without a discrete routing variable. Its degraded performance indicates that the gains cannot be explained by parameter count alone. When regimes are still inferred but no longer condition generation (A2), recovery remains substantial, while forecasting performance declines considerably. This contrast highlights the role of the inferred regime in governing prediction rather than merely providing descriptive labels. Consistently, replacing the learned assignment with fixed (A3a) or random routing (A3b) degrades both metrics, confirming the importance of context-dependent regime selection. Finally, removing the structural losses (A4) primarily weakens regime recovery but also reduces forecasting accuracy, indicating that these objectives help establish distinct and predictive regime representations. Overall, these results show that HERON's gains arise from learning context-dependent regimes and using them to govern generation, linking regime discovery to the dynamics the model predicts.

Model	Forecasting			Regime recovery (Macro-F1)		
	Network ADE	NBA ADE/FDE	AMASS ADE/FDE	Network	NBA	AMASS
HERON	**0.038**	**0.81/1.49**	**0.047/0.066**	**0.913**	**0.891**	**0.412**
A1: no-regime, capacity-matched	0.048	1.52/2.97	0.055/0.078	0.248	0.546	0.290
A2: passive regime	0.049	1.52/2.99	0.056/0.084	<u>0.786</u>	<u>0.782</u>	<u>0.369</u>
A3a: fixed regime	0.052	1.35/2.09	<u>0.053</u>/0.075	0.163	0.357	0.217
A3b: random regime	0.049	1.38/2.09	0.055/0.075	0.371	0.501	0.307
A4: no structural loss	<u>0.045</u>	<u>1.12</u>/<u>1.83</u>	0.054/<u>0.068</u>	0.557	0.527	0.317

Table 3: **Component ablations.** Test forecasting performance (ADE on Network and ADE/FDE on NBA and AMASS) and test regime-recovery performance (Macro-F1). Lower forecasting errors and higher Macro-F1 are better. Best results are in bold and second-best results are underlined.

8 Prior work

Temporal graph methods. Temporal graph models extend GNNs to evolving relational systems, where node features, edges, or topology change over time. Recurrent and evolution-based GNNs update node embeddings or graph parameters across snapshots [Pareja et al., 2020, Rossi et al., 2020], while continuous-time temporal graph networks maintain event-driven memories [Souza et al., 2022, Longa et al., 2023, Huang et al., 2023]. Spatio-temporal GNNs combine relational message passing with temporal modeling for structured forecasting [Cini et al., 2023, Li et al., 2017, Chen et al., 2021], and recent graph state-space models further integrate relational mixing within latent dynamical systems [Li et al., 2024b, Behrouz and Hashemi, 2024]. However, these representations are primarily optimized for prediction, with spatial aggregation, temporal memory, and rollout dynamics tightly coupled, making the system-level dynamical mode difficult to identify.

Latent dynamical models. Latent dynamical and neural state-space models represent trajectories through hidden states and transition dynamics. Classical and neural formulations learn compact latent dynamics for sequential prediction [Kalman, 1960, Ho and Kálmán, 1966, Liu et al., 2024], while structured models such as S4 Gu et al. [2022] and Mamba Gu and Dao [2024] scale these ideas to long-range dependencies, and world models support autonomous latent rollouts and planning [Hafner et al., 2019, 2021, 2025, Sestak et al., 2025]. These models achieve strong predictive performance, but their latent states are generally not exposed as interpretable regimes that directly govern generation. Conversely, switching dynamical systems [Linderman et al., 2016, Fox et al., 2008] and change-point models [Caldarelli et al., 2022, Altamirano et al., 2023, Madrid Padilla et al., 2023] explicitly represent discrete modes, but typically focus on identifying them in observed sequences rather than generating future graph-structured trajectories.

HERON bridges this gap by unifying graph-structured trajectory generation with self-supervised regime identification, enabling forecasts that are predictable and regimes that are actionable.

9 Conclusion and future directions

We introduced HERON, a self-supervised hierarchical framework that unifies regime inference and spatio-temporal feature generation on graphs through discrete system-level regimes that condition entity-level trajectory generation. Across controlled and real-world datasets, HERON recovers meaningful regimes while remaining competitive with strong forecasting baselines. Moreover, clamping the regime during rollout from a shared context produces distinct yet coherent alternative trajectories. These results support the central premise of this work: regimes are not merely post-hoc explanations but explicit conditioning variables that enable forecasting, tracking, and controlled simulation of alternative dynamical modes. Adaptive regime discovery and modeling topological events, such as node appearance and disappearance, as part of the generative process remain promising directions (see Appendix 12.8 for a more detailed discussion). Furthermore, extending HERON across different relational architectures and data modalities represents an important direction for future work, particularly in domains such as biomedicine, where regime inference could reveal meaningful latent states associated with disease progression or patient dynamics.

10 Acknowledgements

This work was partially supported by the Swiss National Science Foundation (SNSF) under grant IZVSZ2_229539 and the PHRT-SDSC project under grant 2022-811/C22-16P. We thank the anonymous reviewers for their insightful feedback and constructive discussion.

References

Yuqi Chen, Kan Ren, Yansen Wang, Yuchen Fang, Weiwei Sun, and Dongsheng Li. ContiFormer: continuous-time transformer for irregular time series modeling. In *Advances in Neural Information Processing Systems*, 2023.

Chen Liu, Ke Xu, Liangbo L Shen, Guillaume Huguet, Zilong Wang, Alexander Tong, Danilo Bzdok, Jay Stewart, Jay C Wang, Lucian V Del Priore, et al. ImageFlowNet: forecasting multiscale image-level trajectories of disease progression with irregularly sampled longitudinal medical images. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2025.

Satya Narayan Shukla and Benjamin Marlin. Multi-Time attention networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2020.

Jinyong Fan and Yanyan Shen. StockMixer: a simple yet strong MLP-based architecture for stock price forecasting. In *AAAI Conference on Artificial Intelligence*, 2024.

Tong Li, Zhaoyang Liu, Yanyan Shen, Xue Wang, Haokun Chen, and Sen Huang. MASTER: market-guided stock transformer for stock price forecasting. In *AAAI Conference on Artificial Intelligence*, 2024a.

Hao Qian, Hongting Zhou, Qian Zhao, Hao Chen, Hongxiang Yao, Jingwei Wang, Ziqi Liu, Fei Yu, Zhiqiang Zhang, and Jun Zhou. MDGNN: multi-relational dynamic graph neural network for comprehensive and dynamic stock investment prediction. In *AAAI Conference on Artificial Intelligence*, 2024.

Charlie Tang and Russ R Salakhutdinov. Multiple futures prediction. In *Advances in Neural Information Processing Systems*, 2019.

Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. *arXiv preprint arXiv:2104.00563*, 2021.

Lucas Basler, Hubert S Gabryś, Sabrina A Hogan, Matea Pavic, Marta Bogowicz, Diem Vuong, Stephanie Tanadini-Lang, Robert Förster, Ken Kudura, Martin W Huellner, et al. Radiomics, tumor volume, and blood biomarkers for early prediction of pseudoprogression in patients with metastatic melanoma treated with immune checkpoint inhibition. *Clinical Cancer Research*, 2020.

HS Gabryś, L Basler, S Burgermeister, S Hogan, M Ahmadsei, M Pavic, M Bogowicz, D Vuong, S Tanadini-Lang, R Förster, et al. PET/CT radiomics for prediction of hyperprogression in metastatic melanoma patients treated with immune checkpoint inhibitors. *Frontiers in Oncology*, 2022.

Günter U Höglinger, Charles H Adler, Daniela Berg, Christine Klein, Tiago F Outeiro, Werner Poewe, Ronald Postuma, A Jon Stoessl, and Anthony E Lang. A biological classification of Parkinson's disease: the SynNeurGe research diagnostic criteria. *The Lancet Neurology*, 2024.

Thorsten Persigehl, Simon Lennartz, and Lawrence H Schwartz. iRECIST: how to do it. *Cancer Imaging*, 2020.

Daniel Cunha Oliveira, Dylan Sandfelder, André Fujita, Xiaowen Dong, and Mihai Cucuringu. Tactical asset allocation with macroeconomic regime detection. *arXiv preprint arXiv:2503.11499*, 2025.

Rongbo Chen, Mingxuan Sun, Kunpeng Xu, Jean-Marc Patenaude, and Shengrui Wang. Clustering-based cross-sectional regime identification for financial market forecasting. In *International Conference on Database and Expert Systems Applications*, 2022.

Scott W Linderman, Andrew C Miller, Ryan P Adams, David M Blei, Liam Paninski, and Matthew J Johnson. Recurrent switching linear dynamical systems. *arXiv preprint arXiv:1610.08466*, 2016.

Emily Fox, Erik Sudderth, Michael Jordan, and Alan Willsky. Nonparametric Bayesian learning of switching linear dynamical systems. In *Advances in Neural Information Processing Systems*, 2008.

Florian Sestak, Artur Toshev, Andreas Fürst, Günter Klambauer, Andreas Mayr, and Johannes Brandstetter. LAM-SLIDE: latent space modeling of spatial dynamical systems via linked entities. *arXiv preprint arXiv:2502.12128*, 2025.

Pei Xu, Jean-Bernard Hayet, and Ioannis Karamouzas. SocialVAE: human trajectory prediction using timewise latents. In *European Conference on Computer Vision*, 2022.

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Amauri Souza, Diego Mesquita, Samuel Kaski, and Vikas Garg. Provably expressive temporal graph networks. In *Advances in Neural Information Processing Systems*, 2022.

Antonio Longa, Veronica Lachi, Gabriele Santin, Monica Bianchini, Bruno Lepri, Pietro Liò, Andrea Passerini, et al. Graph neural networks for temporal graphs: state of the art, open challenges, and opportunities. *Transactions on Machine Learning Research*, 2023.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. In *Advances in Neural Information Processing Systems*, 2023.

Jiaxuan You, Tianyu Du, and Jure Leskovec. ROLAND: graph learning framework for dynamic graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. EvolveGCN: evolving graph convolutional networks for dynamic graphs. In *AAAI Conference on Artificial Intelligence*, 2020.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. In *International Conference on Machine Learning*, 2020.

Jintang Li, Ruofan Wu, Xinzhou Jin, Boqun Ma, Liang Chen, and Zibin Zheng. State space models on temporal graphs: a first-principles study. In *Advances in Neural Information Processing Systems*, 2024b.

Ali Behrouz and Farnoosh Hashemi. Graph Mamba: towards learning on graphs with state space models. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024.

Edoardo Caldarelli, Philippe Wenk, Stefan Bauer, and Andreas Krause. Adaptive Gaussian process change point detection. In *International Conference on Machine Learning*, 2022.

Matias Altamirano, François-Xavier Briol, and Jeremias Knoblauch. Robust and scalable Bayesian online changepoint detection. In *International Conference on Machine Learning*, 2023.

Carlos Misael Madrid Padilla, Haotian Xu, Daren Wang, Oscar Hernan Madrid Padilla, and Yi Yu. Change point detection and inference in multivariate non-parametric models under mixing conditions. In *Advances in Neural Information Processing Systems*, 2023.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 2025.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *International Conference on Machine Learning*, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: a continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 1991.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 2022.

Yoshua Bengio, Ian Goodfellow, Aaron Courville, et al. *Deep learning*. MIT Press, Cambridge, MA, USA, 2017.

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

Joshua Glaser, Matthew Whiteway, John P Cunningham, Liam Paninski, and Scott Linderman. Recurrent switching dynamical systems models for multiple interacting neural populations. In *Advances in Neural Information Processing Systems*, 2020.

Abdul Fatir Ansari, Konstantinos Benidis, Richard Kurle, Ali Caner Turkmen, Harold Soh, Alexander Smola, Bernie Wang, and Tim Januschowski. Deep explicit duration switching models for time series. In *Advances in Neural Information Processing Systems*, 2021.

Hyunwook Lee and Sungahn Ko. TESTAM: a time-enhanced spatio-temporal attention model with mixture of experts. In *International Conference on Learning Representations*, 2024.

Chenxin Xu, Robby T Tan, Yuhong Tan, Siheng Chen, Yu Guang Wang, Xinchao Wang, and Yanfeng Wang. EqMotion: equivariant multi-agent motion prediction with invariant interaction reasoning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

Benjamin Gompertz. XXIV. on the nature of the function expressive of the law of human mortality, and on a new mode of determining the value of life contingencies. in a letter to Francis Baily, esq. FRS &c. *Philosophical Transactions of the Royal Society of London*, 1825.

Anna Kane Laird. Dynamics of tumour growth. *British Journal of Cancer*, 1964.

Uzi Harush and Baruch Barzel. Dynamic patterns of information flow in complex networks. *Nature Communications*, 2017.

K. Linou, D. Linou, and M. de Boer. NBA player movements, 2016. URL `https://github.com/linouk23/NBA-Player-Movements`.

Yisong Yue, Patrick Lucey, Peter Carr, Alina Bialkowski, and Iain Matthews. Learning fine-grained spatial models for dynamic sports play prediction. In *IEEE International Conference on Data Mining*, 2014.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. AMASS: archive of motion capture as surface shapes. In *IEEE/CVF International Conference on Computer Vision*, 2019a.

Abhinanda R Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J Black. BABEL: bodies, action and behavior with English labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021a.

Andrea Cini, Ivan Marisca, Daniele Zambon, and Cesare Alippi. Taming local effects in graph-based spatiotemporal forecasting. In *Advances in Neural Information Processing Systems*, 2023.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

Yuzhou Chen, Ignacio Segovia, and Yulia R Gel. Z-GCNETs: time zigzags at graph convolutional networks for time series forecasting. In *International Conference on Machine Learning*, 2021.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 1960.

BL Ho and Rudolf E Kálmán. Effective construction of linear state-variable models from input/output functions: Die konstruktion von linearen modellen in der darstellung durch zustandsvariable aus den beziehungen für ein- und ausgangsgrößen. *at-Automatisierungstechnik*, 1966.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: visual state space model. In *Advances in Neural Information Processing Systems*, 2024.

Albert Gu and Tri Dao. Mamba: linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: archive of motion capture as surface shapes. `https://amass.is.tue.mpg.de/`, 2019b. Dataset homepage. Accessed: 2026-05-06.

Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F. Troje. MoVi: a large multipurpose motion and video dataset. *arXiv preprint arXiv:2003.01888*, 2020.

Carnegie Mellon University. CMU MoCap dataset, 2016. URL `http://mocap.cs.cmu.edu`.

Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The KIT whole-body human motion database. In *International Conference on Advanced Robotics*, 2015.

Abhinanda R. Punnakkal, Arjun Chandrasekaran, Nikos Athanasiou, Alejandra Quiros-Ramirez, and Michael J. Black. BABEL: bodies, action and behavior with English labels. `https://babel.is.tue.mpg.de/`, 2021b. Dataset download portal. Accessed: 2026-05-06.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: a skinned multi-person linear model. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 2015.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI Conference on Artificial Intelligence*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Hongcan Guo, Haolang Lu, Guoshun Nan, Bolun Chu, Jialin Zhuang, Yuan Yang, Wenhao Che, Xinye Cao, Sicong Leng, Qimei Cui, et al. Advancing expert specialization for better MoE. *arXiv preprint arXiv:2505.22323*, 2025.

Contents

1 Introduction **1**

2 Problem formulation **2**

3 Unifying regime inference and forecasting on spatio-temporal graphs **3**

3.1 State representation 3

3.2 Hierarchical regime-conditioned architecture 4

3.2.1 Inference paths 4

3.2.2 Encoder 5

3.2.3 Decoder 6

3.3 Training procedure 6

4 Experimental setup **7**

5 Forecasting performance with regime conditioning **7**

6 HERON recovers meaningful regime **8**

7 Component ablations **9**

8 Prior work **10**

9 Conclusion and future directions **10**

10 Acknowledgements **11**

11 Impact Statement **16**

12 Appendix **16**

12.1 HERON 16

12.1.1 Hierarchical states modeling 16

12.1.2 Mixture-of-experts as a transition model. 17

12.1.3 Posterior and Prior Inference Paths 18

12.1.4 Loss Implementations 19

12.2 Experimental Protocol and Metrics 20

12.3 Baselines 21

12.4 Datasets 23

12.4.1 Generated Dataset Construction 23

12.4.2 Real-World Datasets 25

12.5 Complementary Experiments 26

12.5.1 Training results and statistical significance 26

12.5.2 Quantitative regime interventions 26

12.5.3 Imbalanced regime frequencies . . . 27
12.5.4 Overlapping regimes and soft routing . . . 28
12.5.5 AMASS regime-recovery errors . . . 29
12.5.6 Regime reproducibility across runs . . . 29
12.5.7 GraphSSM enhanced with HERON . . . 29
12.5.8 Sensitivity to mis-specification of K . . . 30
12.5.9 Selecting the number of regimes without labels . . . 31
12.6 Hyperparameters . . . 32
12.7 Model Size and Computational Budget . . . 32
12.8 Limitations . . . 33
12.8.1 MoE scales linearly with K. . . . 33
12.8.2 Losses requires balance between weights. . . . 33
12.8.3 Uniform regime usage is restrictive under severe imbalance. . . . 34
12.8.4 One active regime at rollout. . . . 34
12.8.5 Fixed target topology. . . . 34
12.8.6 Number of regimes K is a model-selection hyperparameter. . . . 34

11 Impact Statement

HERON aims to improve the forecasting and understanding of evolving relational systems by identifying the dynamical regimes governing their evolution. This approach could support more informed decision-making in sensitive domains such as healthcare, epidemiology, and finance. While we do not currently foresee any immediate societal concerns arising from the proposed methodology, inaccurate regime identification or forecasts could lead to misleading conclusions. Moreover, identified regimes should not be interpreted as causal explanations without further validation. We therefore emphasize the importance of rigorous evaluation and human oversight when deploying HERON in sensitive domains.

12 Appendix

This appendix provides the implementation and evaluation details that are most relevant for assessing HERON as a regime-aware latent modeling framework. The main text presents the conceptual model and empirical results; here we clarify how the method can be instantiated on top of existing latent graph-temporal models, how posterior and prior paths differ, how the generated datasets are constructed, and how evaluation is performed.

12.1 HERON

12.1.1 Hierarchical states modeling

Latent state initialization.

We initialize the hierarchical latent state as follows. For each entity i, let τ_i denote the first time step at which the entity is observed. The entity-level latent state and deterministic recurrent state are initialized as

$$z^i_{\tau_i} = \mathbf{0}, \qquad h^i_{\tau_i} = \mathbf{0}.$$

Thus, entities present at the beginning of the sequence are initialized at the first time step, while entities entering later are initialized when they first appear.

At the system level, the deterministic recurrent state is initialized analogously:

$$h^G_0 = \mathbf{0}.$$

The initial regime distribution is set to the uniform categorical prior over the K regimes,

$$p(r_0 = k) = \frac{1}{K}, \qquad k \in \{1, \dots, K\},$$

or equivalently,

$$\boldsymbol{\pi}_0 = \frac{1}{K}\mathbf{1}_K.$$

This initialization encodes a non-informative prior over regimes at the first time step, assigning equal probability to each regime. Alternative initial regime priors can be incorporated in the same manner as the load-balancing procedure described in Appendix 12.1.4.

Regime discrete variable.

For a graph snapshot index by b in the batch, the system-level transition module produces regime logits ℓ_b. During training, we use the Gumbel–Softmax reparameterization to sample regimes while retaining gradient information. First, we generate Gumbel noise from uniform random variables:

$$g_{b,k} = -\log(-\log u_{b,k}), \qquad u_{b,k} \sim \mathrm{U}(0,1). \tag{8}$$

We then add this noise to the regime logits and apply a temperature-scaled softmax to obtain a differentiable, relaxed regime assignment:

$$\tilde{r}_{b,k} = \frac{\exp((\ell_{b,k} + g_{b,k})/\tau)}{\sum_{j=1}^{K} \exp((\ell_{b,j} + g_{b,j})/\tau)}, \qquad k \in \{1, \dots, K\}. \tag{9}$$

The temperature τ is annealed during training, typically from 1.0 to 0.3 over 60% of training, encouraging exploration early and sharper assignments later. Finally, we obtain a one-hot regime assignment using the argmax, while the straight-through estimator preserves gradient flow through the relaxed assignment:

$$r_{b,k}^{\mathrm{hard}} = \mathbb{1}\left[k = \arg\max_j \tilde{r}_{b,j}\right], \qquad r_b = r_b^{\mathrm{hard}} - \mathrm{sg}(\tilde{r}_b) + \tilde{r}_b. \tag{10}$$

Here, $\ell_{b,k}$ denotes the logit for regime k in graph b, and $g_{b,k}$ denotes the corresponding Gumbel noise. The relaxed assignment $\tilde{r}_b$ supports gradient propagation, while r_b^{hard} provides the one-hot assignment used in the forward pass.

Entity-level latent. The entity states are modeled as Gaussian stochastic variables. For each entity i, the posterior and prior transition heads output the parameters of a diagonal Gaussian distribution:

$$q_\phi(z_t^i \mid h_t^i, \mathrm{emb}_t^i, r_t) = \mathcal{N}\big(\mu_{\phi,t}^i, \mathrm{diag}\big((\sigma_{\phi,t}^i)^2\big)\big)\,, \tag{8}$$

with an analogous parameterization for the prior distribution defined as

$$p_\psi(\hat{z}_t^i \mid h_t^i, \widehat{\mathrm{emb}}_t^{\,i}, \hat{r}_t) = \mathcal{N}\big(\mu_{\psi,t}^i, \mathrm{diag}\big((\sigma_{\psi,t}^i)^2\big)\big)\,. \tag{9}$$

The regime variable therefore provides a discrete system-level assignment, while the Gaussian entity latents represent continuous node-level uncertainty conditioned on the active regime.

12.1.2 Mixture-of-experts as a transition model.

Our formulation in Section 2 assumes that the modeled system evolves according to K distinct transition functions $\{f_k\}_{k=1}^K$, each corresponding to a different dynamical regime. This assumption naturally motivates the use of Mixture-of-Experts models [Jacobs et al., 1991, Fedus et al., 2022], since they explicitly parameterize a collection of candidate transition functions and select or combine them through a gating mechanism. In our implementation, we use a one-hot regime assignment rather than a dense mixture. This design choice encourages a clearer separation between regimes and reduces the risk that multiple experts jointly encode indistinguishable dynamics.

The MoE parameterization also provides a direct mechanism for controlled interventions in latent rollouts. At generation time, the selected expert can be clamped, allowing the model to simulate trajectories under a fixed regime rather than allowing the gating network to select regimes autonomously. This enables controlled visualization of alternative plausible futures and is a direct consequence of the explicit regime-conditioned transition structure.

Algorithm 1 HERON hierarchical inference and rollout

Input: Context sequence $(\mathbf{X}_{1:C}, \mathcal{V}_{1:C}, \mathcal{E}_{1:C})$, target topology $(\mathcal{V}^\star, \mathcal{E}^\star)$, context size C, horizon H
Output: Forecasts $\hat{\mathbf{X}}_{C+1:C+H}$ and regime trajectory $r_{1:C+H}$

$T \leftarrow C + H$
Initialize $h_0^G, \{h_0^i\}_{i\in\mathcal{V}_1}, r_0, \{z_0^i\}_{i\in\mathcal{V}_1}$
for $t = 1$ to T **do**
  $h_t^i \leftarrow GRU_\theta^{\mathcal{V}}(h_{t-1}^i, z_{t-1}^i), \quad h_t^G \leftarrow GRU_\theta^G(h_{t-1}^G, r_{t-1})$ ▷ *Recurrent state update*
  if $t \le C$ **then**
    $\mathrm{emb}_t^i \leftarrow \mathrm{GNN}_\phi(h_t^i, \mathbf{X}_t, \mathcal{E}_t), \quad \mathrm{emb}_t^G \leftarrow \mathrm{Pool}_\phi(\{\mathrm{emb}_t^i\}, \mathcal{E}_t)$ ▷ *Posterior embeddings*
    $r_t \sim q_\phi(r_t \mid h_t^G, \mathrm{emb}_t^G), \quad z_t^i \sim q_\phi(z_t^i \mid h_t^i, \mathrm{emb}_t^i, r_t)$ ▷ *Posterior state inference*
    $\hat{\mathbf{X}}_{t+1} \sim g_\theta(\mathbf{X}_{t+1} \mid h_t^G, \{h_t^i\}_{i\in\tilde{\mathcal{V}}_t}, r_t, \{z_t^i\}_{i\in\tilde{\mathcal{V}}_t})$ ▷ *Posterior decoding*
  else
    $\hat{\mathrm{emb}}_t^i \leftarrow \mathrm{GNN}_\psi(h_t^i, \mathcal{E}^\star), \quad \hat{\mathrm{emb}}_t^G \leftarrow \mathrm{Pool}_\psi(\{\hat{\mathrm{emb}}_t^i\}, \mathcal{E}^\star)$ ▷ *Prior embeddings*
    $\hat{r}_t \sim p_\psi(\hat{r}_t \mid h_t^G, \hat{\mathrm{emb}}_t^G), \quad \hat{z}_t^i \sim p_\psi(\hat{z}_t^i \mid h_t^i, \hat{\mathrm{emb}}_t^i, \hat{r}_t)$ ▷ *Prior state inference*
    $\hat{\mathbf{X}}_{t+1} \sim g_\theta(\mathbf{X}_{t+1} \mid h_t^G, \{h_t^i\}_{i\in\tilde{\mathcal{V}}_t}, \hat{r}_t, \{\hat{z}_t^i\}_{i\in\tilde{\mathcal{V}}_t})$ ▷ *Prior decoding*

12.1.3 Posterior and Prior Inference Paths

HERON maintains two inference paths. The *posterior* path is used when the current graph observation is available; the *prior* path is used during rollout. Variables without hats denote posterior quantities, and variables with hats denote prior quantities. Unless explicitly stated, parameters indexed by ϕ and ψ are separate modules with no shared weights. In our implementation, the deterministic recurrent updates and the decoder are shared across both paths, while the posterior and prior embedding networks and stochastic transition heads are distinct. Algorithm 1 show this hierarchical process in details.

Posterior path. Starting from the updated deterministic states h_t, the posterior path then uses the current observed node features and graph topology to form relational embeddings:

$$\mathrm{emb}_t^i = \mathrm{GNN}_\phi\big(h_t^i, \mathbf{X}_t, \mathcal{E}_t\big), \qquad \mathrm{emb}_t^G = \mathrm{Pool}_\phi\big(\{\mathrm{emb}_t^i\}_{i\in\mathcal{V}_t}, \mathcal{E}_t\big). \tag{10}$$

The graph-level embedding produces the categorical regime posterior:

$$r_t \sim q_\phi\Big(r_t \mid h_t^G, \mathrm{emb}_t^G\Big). \tag{11}$$

The sampled regime then conditions the entity-level latent posterior:

$$z_t^i \sim q_\phi\big(z_t^i \mid h_t^i, \mathrm{emb}_t^i, r_t\big), \qquad i \in \mathcal{V}_t. \tag{12}$$

This path is observation-conditioned. It is used to infer latent states from the context and to provide a training target for the prior.

Prior path. Starting from the updated deterministic states h_t, the prior path embeddings are computed from recurrent states and topology alone:

$$\hat{\mathrm{emb}}_t^i = \mathrm{GNN}_\psi\Big(h_t^i, \hat{\mathcal{E}}_t\Big), \qquad \hat{\mathrm{emb}}_t^G = \mathrm{Pool}_\psi\Big(\{\hat{\mathrm{emb}}_t^i\}_{i\in\hat{\mathcal{V}}_t}, \hat{\mathcal{E}}_t\Big). \tag{13}$$

The prior regime is sampled from the graph-level prior:

$$\hat{r}_t \sim p_\psi\Big(\hat{r}_t \mid h_t^G, \hat{\mathrm{emb}}_t^G\Big). \tag{14}$$

Finally, the regime-conditioned node-level prior produces entity latent states:

$$\hat{z}_t^i \sim p_\psi\Big(\hat{z}_t^i \mid h_t^i, \hat{\mathrm{emb}}_t^i, \hat{r}_t\Big), \qquad i \in \hat{\mathcal{V}}_t. \tag{15}$$

12.1.4 Loss Implementations

Prediction loss. To encourage the latent regime to capture dynamical evolution rather than static feature configurations, the decoder is trained to predict node-level feature velocities,

$$\mathbf{Y}_t = \mathbf{X}_{t+1} - \mathbf{X}_t. \tag{16}$$

This velocity target is used only as the training target. During inference, predictions are accumulated and evaluated in the original feature space. The decoder formulation in the main text is kept in terms of feature predictions for consistency with the rollout objective and evaluation protocol. Note that prediction loss is only applied to time-varying features. Any static node or edge features will be seen by the model as input but will not be reconstructed.

During the context window, the prediction loss is applied to both posterior and prior paths. This parallel decoding reduces the mismatch between the observation-conditioned states used during context and the autonomous prior states used after the context window. During the forecast horizon, the loss is applied only to the prior path, since future observations are unavailable and the posterior path cannot be instantiated.

Balanced KL losses. The KL terms are computed analytically for both components of the hierarchical stochastic state. For the discrete regime variable, the KL divergence is evaluated between categorical distributions over the K regimes as follows:

$$\mathrm{KL}[q_\phi(r_t) \,\|\, p_\psi(\hat{r}_t)] = \sum_{k=1}^{K} q_\phi(r_t = k) \log \frac{q_\phi(r_t = k)}{p_\psi(\hat{r}_t = k)}. \tag{17}$$

For the entity-level latent states, the KL divergence is evaluated in closed form between diagonal Gaussian distributions:

$$\mathrm{KL}\big[q_\phi(z_t^i) \,\|\, p_\psi(\hat{z}_t^i)\big] = \frac{1}{2} \sum_{d=1}^{d_z} \left[\log \frac{(\sigma^i_{\psi,t,d})^2}{(\sigma^i_{\phi,t,d})^2} + \frac{(\sigma^i_{\phi,t,d})^2 + (\mu^i_{\phi,t,d} - \mu^i_{\psi,t,d})^2}{(\sigma^i_{\psi,t,d})^2} - 1 \right]. \tag{18}$$

These terms are used in the balanced KL formulation described in the main text, where the dynamics term $\mathcal{L}_{\mathrm{dyn}}$ trains the prior to match the stopped-gradient posterior and the representation term $\mathcal{L}_{\mathrm{rep}}$ regularizes the posterior toward the stopped-gradient prior. We define the two losses as the following:

$$\mathcal{L}_{\mathrm{dyn}} = \mathrm{KL}[\mathrm{sg}(q_\phi(r_t)) \,\|\, p_\psi(\hat{r}_t)] + \frac{1}{|\mathcal{V}_t|} \sum_{i \in \mathcal{V}_t} \mathrm{KL}\big[\mathrm{sg}\big(q_\phi(z_t^i)\big) \,\|\, p_\psi(\hat{z}_t^i)\big], \tag{19}$$

$$\mathcal{L}_{\mathrm{rep}} = \mathrm{KL}[q_\phi(r_t) \,\|\, \mathrm{sg}(p_\psi(\hat{r}_t))] + \frac{1}{|\mathcal{V}_t|} \sum_{i \in \mathcal{V}_t} \mathrm{KL}\big[q_\phi(z_t^i) \,\|\, \mathrm{sg}\big(p_\psi(\hat{z}_t^i)\big)\big]. \tag{20}$$

Structural loss. The structural regularizers are applied primarily to the posterior path. The balanced KL objective then transfers these constraints to the prior by aligning the prior latent distributions with the posterior ones. This design avoids imposing all structural constraints directly on the autonomous rollout path while still encouraging the prior to reproduce structured regime assignments.

The **load-balancing** term encourages the model to use its K experts in a balanced manner across the batch. We first obtain the regime probabilities by applying a softmax to the regime logits without Gumbel noise. For graph with index b in the batch and regime k, this gives

$$\pi_{b,k} = \mathrm{softmax}(\ell_b)_k. \tag{21}$$

We then average these probabilities over the B graphs in the batch to estimate the empirical usage of each regime, denoted by $\bar{\pi}_k$. We compare this distribution against a uniform target u, which assigns equal probability to each of the K regimes:

$$\bar{\pi}_k = \frac{1}{B} \sum_{b=1}^{B} \pi_{b,k}, \qquad u_k = \frac{1}{K}. \tag{22}$$

The load-balancing loss is defined as the KL divergence between the empirical regime distribution and this uniform target. Substituting $u_k = 1/K$ gives the following equivalent expression:

$$\begin{aligned}\mathcal{L}_{\text{balancing}} &= \mathrm{KL}[\bar{\pi} \,\|\, u] \\ &= \sum_{k=1}^{K} \bar{\pi}_k \log \frac{\bar{\pi}_k}{u_k} \\ &= \sum_{k=1}^{K} \bar{\pi}_k \log \left(K \bar{\pi}_k\right).\end{aligned} \tag{23}$$

The uniform target reflects the assumption that regimes are approximately balanced in the datasets considered in this work. Under severe regime imbalance, however, this assumption becomes restrictive. In Appendix 12.5.3, we therefore describe an extension in which the regime prior is learned from the data rather than fixed to be uniform. In a controlled setting with a 70%/20%/10% regime distribution, enforcing the uniform prior causes minority-regime recall to fall to zero. Replacing the fixed uniform target with an unsupervised learned categorical prior recovers minority-regime recall to 0.70 and improves Macro-F1 from 0.60 to 0.78.

The **entropy regularizer** is computed on the sample-level regime probabilities before Gumbel perturbation. Minimizing this entropy encourages low-entropy regime assignments, making the inferred regime distribution closer to one-hot and improving the confidence of expert selection:

$$\mathcal{L}_{\text{entropy}} = \mathcal{H}[Q_t] = \frac{1}{B} \sum_{b=1}^{B} H(\pi_{b,t}) = -\frac{1}{B} \sum_{b=1}^{B} \sum_{k=1}^{K} \pi_{b,t,k} \log \pi_{b,t,k}. \tag{24}$$

Finally, the **expert-specialization** term is computed using all expert outputs, even though the effective gating selects a single expert through the straight-through regime assignment. The loss is applied to the mean component of the Gaussian entity latent distribution, rather than to sampled latent vectors, to reduce sensitivity to sampling noise. This penalizes alignment between different expert outputs for the same entity and encourages regime-specific latent mappings. The loss can formally be writen as

$$\mathcal{L}_{\text{ortho}} = \sum_{i \in V_t} \sum_{j=1}^{K} \sum_{\substack{k=1 \\ k \neq j}}^{K} \left\| \frac{\left\langle \mu^i_{\phi,j}, \mu^i_{\phi,k} \right\rangle}{\left\langle \mu^i_{\phi,k}, \mu^i_{\phi,k} \right\rangle + \epsilon} \mu^i_{\phi,k} \right\|_2^2, \tag{25}$$

where $\langle \cdot \rangle$ denotes the inner product between two vectors, and $\mu^i_{\phi,k}$ represents the mean component of the Gaussian output of expert k for entity i after the MoE (see section 12.1.1 for mode detail on notation). We found this latent-space formulation preferable to weight-space orthogonality alternatives, which were less effective in our experiments. Future work may consider lower-cost specialization mechanisms, for example by parameterizing experts as basis functions, although such restrictions may reduce expressivity.

12.2 Experimental Protocol and Metrics

Sequences are split at the sequence level, where windows from the same original sequence are never allowed to appear in different splits. We use fixed windows of length 10 for generated datasets, with $C = 5$ context steps and $H = 5$ rollout steps. For real-world datasets NBA [Linou et al., 2016, Yue et al., 2014] and AMASS [Mahmood et al., 2019a], we use longer windows ($C = 8, H = 12$); the context and horizon lengths are set according to the preprocessing protocol used for the corresponding dataset.

Hyperparameters are selected with four-fold cross-validation on the training split. All reported forecasting metrics are computed on the horizon using the prior rollout path, not the posterior path. This ensures that evaluation matches the intended deployment setting where future observations are unavailable.

Let $\mathcal{B}$ be the set of test windows, and let $\mathcal{V}_{b,t}$ be the nodes evaluated in window b at time t. The Average Displacement Error is

$$\mathrm{ADE} = \frac{1}{|\mathcal{B}|H} \sum_{b\in\mathcal{B}} \sum_{t=C+1}^{C+H} \frac{1}{|\mathcal{V}_{b,t}|} \sum_{i\in\mathcal{V}_{b,t}} \left\|\hat{x}^i_{b,t} - x^i_{b,t}\right\|_2 . \tag{26}$$

The Final Displacement Error is

$$\mathrm{FDE} = \frac{1}{|\mathcal{B}|} \sum_{b\in\mathcal{B}} \frac{1}{|\mathcal{V}_{b,C+H}|} \sum_{i\in\mathcal{V}_{b,C+H}} \left\|\hat{x}^i_{b,C+H} - x^i_{b,C+H}\right\|_2 . \tag{27}$$

More precisely, we report the minimum FDE across several stochastic rollouts samples ($M = 20$):

$$\mathrm{minFDE} = \frac{1}{|\mathcal{B}|} \sum_{b\in\mathcal{B}} \min_{m\in\{1,\dots,M\}} \frac{1}{|\mathcal{V}_{b,C+H}|} \sum_{i\in\mathcal{V}_{b,C+H}} \left\|\hat{x}^{i,(m)}_{b,C+H} - x^i_{b,C+H}\right\|_2 . \tag{28}$$

For regime evaluation, labels are used only after training. Because discrete latent states are identifiable only up to permutation, we first align predicted regime IDs with ground-truth labels using the Hungarian algorithm on the validation confusion matrix. Macro-F1 is then computed after this alignment. For AUC, we permute the predicted regime probabilities according to the same assignment and compute one-vs-rest AUC for multi-class settings.

12.3 Baselines

The baselines are chosen to contextualize forecasting quality, not to claim that all methods solve the same regime-inference problem. Persistence baselines test whether the task can be solved by short-term extrapolation. S4 evaluates a strong non-relational sequence model. EvolveGCN represents recurrent temporal graph modeling. GraphSSM and LaM-SLidE provide latent spatio-temporal forecasting references.

All baselines are evaluated under the same data splits, context length C, horizon H, autoregressive rollout protocol, and forecasting metrics as HERON. Learned baselines are tuned using the four-fold cross-validation protocol in Section 12.2. For methods that do not expose an explicit regime variable, we report forecasting metrics only.

Persistence (Static and Velocity). We include two parameter-free temporal heuristics: Static, $\hat{X}_{t+1} = X_t$, and Velocity, $\hat{X}_{t+1} = 2X_t - X_{t-1}$. Both are applied to the time-varying channels only, while static descriptors are copied forward unchanged. Rollouts are autoregressive, using previous predictions as inputs after the context window. For Velocity, the first predicted step uses the last observed difference $X_C - X_{C-1}$; newly appearing nodes in Gompertz default to Static at their entry step.

S4 Gu et al. [2022]. S4 is used as a non-relational state-space baseline. It models temporal dependencies with structured state-space layers but does not use graph topology or edge features. Given node features $X \in \mathbb{R}^{B\times N\times C\times d_V}$, we apply shared S4 blocks independently to each node trajectory by reshaping the input as $(BN) \times C \times d_V$. To allow limited graph-level context without using edges, the resulting node states are aggregated with a masked mean over the entities present in each graph and injected back into each node representation through a linear residual followed by layer normalization:

$$H = \mathrm{S4Block}(X.\mathrm{reshape}(BN, C, d_V)) \,.\mathrm{reshape}(B, N, C, d_h),$$

$$c_{b,t} = \frac{1}{|\mathcal{V}_{b,t}|} \sum_{i\in\mathcal{V}_{b,t}} H_{b,i,t}, \qquad \tilde{H}_{b,i,t} = \mathrm{LayerNorm}(H_{b,i,t} + W_c c_{b,t}) .$$

A regression head maps $\tilde{H}$ to the next-step target distribution, and horizon predictions are produced autoregressively.

rSLDS Glaser et al. [2020]. rSLDS is the canonical classical switching baseline, and reports both forecasting and regime recovery. We use the Linderman lab `ssm` implementation in its recurrent-only variant, in which the discrete regime is driven by the continuous state rather than by its own past:

$$p(z_t = k \mid x_{t-1}) \propto \exp\big(R_k^\top x_{t-1} + r_k\big), \qquad x_t = A_{z_t} x_{t-1} + b_{z_t} + \epsilon_t, \qquad y_t = C x_t + d + \eta_t,$$

fitted by Laplace-EM on the training fold. Being non-relational, the model needs a flat observation: NBA and AMASS frames are flattened node-major into a single velocity vector (22 and 66 dimensions), the latter reduced to 15 PCA components; on Network, where the node set differs across sequences, one shared model is fitted to all per-node trajectories. Forecasts are autonomous rollouts seeded from the posterior-filtered terminal context state, and regimes are read from the discrete posterior over the context.

RED-SDS Ansari et al. [2021]. RED-SDS is the considered deep switching baseline. It augments the regime z_t with a count variable c_t, so dwell times are modelled explicitly rather than implied by a self-transition probability. With $u_k(c)$, we define the probability of remaining in regime k after c steps. The joint regime-duration transition is then:

$$\begin{aligned} p(z_t, c_t \mid z_{t-1}, c_{t-1}, x_{t-1}) = u_{z_{t-1}}(c_{t-1})\, \delta[z_t{=}z_{t-1}]\, \delta[c_t{=}c_{t-1}{+}1] \\ + \big(1 - u_{z_{t-1}}(c_{t-1})\big)\, p(z_t \mid z_{t-1}, x_{t-1})\, \delta[c_t{=}1]. \end{aligned}$$

over nonlinear transitions $x_t \sim \mathcal{N}(f_{z_t}(x_{t-1}), \cdot)$ and emissions $y_t \sim \mathcal{N}(g(x_t), \cdot)$, trained by maximising an ELBO with annealed relaxations of both z and c. We use the authors' implementation unmodified, adding only a dataset adapter that reuses the flattening and PCA front-end of rSLDS Glaser et al. [2020], with the maximum duration set to the window length. Forecasts come from the open-loop unroll, regimes from the smoothed posterior marginals.

TESTAM Lee and Ko [2024]. TESTAM is the graph mixture-of-experts baseline: it routes implicitly and locally, where HERON commits to an explicit system-level regime. Three experts defined as temporal, adaptive-graph (supports generated from a learned memory), and spatial-attention are combined by a gate selecting one expert per node and time step,

$$g = \mathrm{softmax}\big(\mathrm{Gate}(h^{\mathrm{tmp}}, h^{\mathrm{adp}}, h^{\mathrm{att}})\big), \qquad \hat{y} = o_{e^\star}, \quad e^\star = \arg\max_e g_e,$$

trained with the authors' pseudo-label routing losses, which push the gate away from the worst expert and towards the best. We use the model unchanged. As our windows carry no time-of-day signal, its time channel becomes a within-window positional index, and multi-channel node features are emitted through the expert output width, one routing decision being shared across channels. On Network, whose node set varies, sequences are zero-padded to a fixed node table. TESTAM is forecasting-only.

EvolveGCN Pareja et al. [2020]. EvolveGCN is a recurrent temporal graph baseline in which the GCN parameters evolve over time. We use the EvolveGCN-H variant, where the layer weights are updated by a GRU driven by the current node embeddings:

$$W_t^{(\ell)} = \mathrm{GRU}\Big(H_t^{(\ell-1)},\, W_{t-1}^{(\ell)}\Big), \qquad H_t^{(\ell)} = \sigma\Big(\hat{A}_t H_t^{(\ell-1)} W_t^{(\ell)}\Big). \tag{29}$$

We attach a two-layer MLP regression head to the final node embeddings $H_t^{(L)}$ and train the model with the same negative Gaussian log-likelihood as HERON. The model is unrolled over the context and prediction windows, consuming observed features during the context and its own predictions during the horizon. We use a two-layer EvolveGCN-H stack, matching the depth of the graph encoder in HERON.

LaM-SLidE Sestak et al. [2025]. LaM-SLidE is a latent generative model for spatial dynamical systems. It first trains an entity-aware autoencoder using learned entity identifiers, then trains a flow-matching prior over latent trajectories conditioned on a context window. We use the authors' public implementation with task-specific tokenization. NBA entities are represented by fixed player/ball identifiers together with coordinates and type indicators; AMASS joints are represented by joint identifiers and anatomical descriptors. The latent prior is trained to generate H future latent steps from the C context latents. At evaluation, we draw $M = 20$ stochastic samples per window, matching the minFDE protocol in Section 4. We evaluate LaM-SLidE on the fixed-entity datasets only; synthetic datasets with variable entity sets are marked as not applicable.

GraphSSM Li et al. [2024b]. GraphSSM extends structured state-space models to temporal graphs by combining temporal SSM updates with graph mixing through a Laplacian-based GHiPPO formulation. We use GraphSSM-S4, the strongest variant reported by the original authors on temporal-graph benchmarks. For comparison with HERON, we stack the same number of GraphSSM-S4 blocks as the HERON encoder, replace the original task head with the same MLP velocity decoder, and train with the identical negative Gaussian log-likelihood. The Laplacian is recomputed at every snapshot for Gompertz, where topology changes over time, and once per window otherwise. GraphSSM-S4 serves as the closest non-regime-aware graph state-space comparator.

Latent-HMM. The HMM baseline adds memory to the simple regime-discovery controls, which cluster one pooled vector per sequence. A K-state Gaussian HMM with diagonal covariances is fitted by EM on per-timestep, node-pooled feature sequences $f_{1:T}$. Its likelihood can be written as:

$$p(f_{1:T}) = \sum_{s_{1:T}} \pi_{s_1} \prod_{t>1} A_{s_{t-1}s_t} \prod_t \mathcal{N}\left(f_t; \mu_{s_t}, \mathrm{diag}, \sigma^2_{s_t}\right),$$

so neighbouring frames are tied by a learned transition law rather than treated independently. We use the GraphSSM latent as the velocity target, meaning the per-graph mean of the pre-head hidden state of a frozen GraphSSM trained only to forecast the next step. Features are standardised on the training fold.

12.4 Datasets

	Synthetic			Real-world	
	Linear	Gompertz	Network	NBA	AMASS
Number of entities per graph	50-100	50-100	50-100	11	22
Number of sequences	$\sim 10^3$	$\sim 10^3$	$\sim 10^3$	$\sim 10^5$	$\sim 10^5$
Number of regimes	3	3	3	2	4
Metrics	ADE	ADE	ADE	ADE/FDE	ADE/FDE
Dynamic topology	–	✓	–	–	–
Dynamic edge features	–	–	–	✓	–
Dynamic regime	–	–	✓	–	✓

Table 4: **Summary of dataset properties.** The table highlights complementary dynamics and scales. ✓ indicates presence; – indicates absence.

12.4.1 Generated Dataset Construction

All generated datasets follow the same high-level protocol. We generate 2000 graph sequences. For each sequence, the number of nodes is sampled uniformly between 50 and 100. Edges are sampled randomly following the network construction procedure used in [Harush and Barzel, 2017]: topology is controlled through the degree structure, and each generated graph is constrained to form a single connected component. This produces random graphs that remain closer to realistic network structure than unconstrained independent edge sampling.

For each node, we create a feature vector of dimension 8. The node features are initialized independently from a uniform distribution, $x_i^{(0)} \sim \mathcal{U}(0.1, 0.6)$. To remain faithful to real world scenarios, we add a Gaussian observational noise ϵ_t^i at each time step to all node features, with $(\epsilon_t \sim \mathcal{N}(\mu = 0, \sigma = 0.03)$. For the Linear and Gompertz datasets, each generated graph sequence is assigned a single regime label $r \in \{1, \ldots, K\}$, and the graph is simulated for 10 time steps according to the corresponding dynamical rule. For the Network dataset, the initial regime is sampled uniformly at the beginning of the simulation. The regime then evolves according to a persistence-based switching process: if the regime has remained unchanged for at least the previous three time steps and that we did not reached yet the horizon, it switches with probability $p = 0.3$ to one of the other $K - 1$ regimes, sampled uniformly at random; otherwise, the current regime is retained. This construction induces temporally coherent regime segments while allowing stochastic transitions

between dynamical modes. No changes of regime occurs during the horizon. We show the dataset generation procedure in the following Algorithm 2.

Algorithm 2 Synthetic Graph Sequence Generation

Input: Number of sequences $N_{\text{seq}} = 2000$, regimes K, switching probability $p = 0.3$, context size C, horizon H

$T \leftarrow C + H$
for $s = 1, \dots, N_{\text{seq}}$ **do**
  Sample $n_s \sim \text{Unif}(\{50, \dots, 100\})$
  Sample edges according to the network construction procedure of [Harush and Barzel, 2017]
  Initialize $x_i^{(0)} \sim \mathcal{U}(0.1, 0.6)$ independently for all nodes i
  if dataset is Linear or Gompertz **then**
    Sample $r \sim \text{Unif}(\{1, \dots, K\})$
    Set $r_t \leftarrow r$ for all $t = 0, \dots, T-1$
  else if dataset is Network **then**
    Sample $r_0 \sim \text{Unif}(\{1, \dots, K\})$
    for $t = 1, \dots, T-1$ **do**
      if $t < C$ **and** $t \geq 3$ **and** $r_{t-1} = r_{t-2} = r_{t-3}$ **then**
        With probability p, sample $r_t \sim \text{Unif}\big(\{1, \dots, K\} \setminus \{r_{t-1}\}\big)$
      else
        Set $r_t \leftarrow r_{t-1}$
  for $t = 0, \dots, T-1$ **do**
    Simulate the graph dynamics at time t under regime r_t

Linear Dynamics The linear dataset removes interaction effects so that regime recovery depends on aggregating temporal evidence across independent entities. Each node follows

$$x_{t+1}^i = a_k x_t^i + b_k, \qquad k \in \{1, \dots, K\}, \tag{30}$$

where (a_k, b_k) are regime-specific coefficients. All nodes in a sequence share the same regime. This dataset serves as a sanity check: a model should recover the shared dynamical mode without relying on relational information. Note that the linearity is in the dynamics, not in the raw features evolution. If we unravel the equation and neglect the bias term, the features evolve according to an exponential law: $x_t^i = x_0 (a_k)^t$.

Regime type	a_k	b_k	Behavior
Decay	0.8	0	Exponential decay
Stable	0.0	1	Constant drift independent of features value
Increase	1.2	0	Exponential increase

Table 5: **Linear dataset regimes.** The three dynamics corresponds to different linear evolution patterns.

Gompertz Dynamics with Topological Events The Gompertz dataset introduces non-linear growth and local topological changes. Node features represent scalar volumes and evolve according to a discrete Gompertz-type update:

$$x_{t+1}^i = a_k^{1-\exp(-c_k)} \left(x_t^i\right)^{\exp(-c_k)}, \qquad k \in \{1, \dots, K\}. \tag{31}$$

The regime controls the growth parameters. When a node reaches the saturation threshold $\theta_{\text{thr}} = 0.5$, it is removed and replaced by two newly spawned entities. The two entities connectivity is the same as the one of its parent. This induces a local change in the node set and graph topology. The setting tests whether regime inference remains stable under non-linear feature evolution and changing graph support. We list the set of considered regimes and their associated parameters in Table 6.

Regime type	a_k	c_k	Behavior
Tumor regression	0.05	0.08	Shrinks toward small volume
Stable disease	0.5	0.00	Approximately constant
Progressive disease	1	0.08	Grows beyond initial features range

Table 6: **Gompertz regimes.** The three dynamics corresponds to biological relevant tumor growth patterns, namely regression, stable and progression.

Network Interaction Dynamics The network dataset is designed to test whether the model can identify regimes defined by interaction mechanisms rather than by isolated node trajectories. The same graph topology can produce different trajectories depending on how node states propagate over edges. We use three dynamical families from [Harush and Barzel, 2017], corresponding to degree-averting, degree-driven, and homogeneous information flow.

For a weighted adjacency matrix A, the node state $x_i(t)$ evolves according to one of three continuous-time systems.

The regulatory dynamics ($\mathbb{R}_2$) combine linear decay with a saturating input from neighboring nodes:

$$\frac{dx_i(t)}{dt} = -x_i(t) + \sum_{j=1}^{N} A_{ij} \frac{x_j(t)^2}{1 + x_j(t)^2}. \tag{32}$$

The saturating input limits the influence of large neighboring states and corresponds to degree-averting flow, in which information tends to avoid hubs and favor peripheral pathways.

The population dynamics ($\mathbb{P}$) combine self-limiting decay with positive reinforcement from neighboring nodes:

$$\frac{dx_i(t)}{dt} = -x_i(t)^3 + \sum_{j=1}^{N} A_{ij} x_j(t)^2. \tag{33}$$

This positive reinforcement yields degree-driven flow, in which hubs dominate propagation.

The epidemic dynamics ($\mathbb{E}$) combine recovery with infection pressure from neighboring nodes, modulated by the susceptible fraction $(1 - x_i(t))$:

$$\frac{dx_i(t)}{dt} = -x_i(t) + \sum_{j=1}^{N} A_{ij}(1 - x_i(t))x_j(t). \tag{34}$$

The susceptibility factor modulates the influence of infected neighbors, yielding homogeneous flow in which nodes contribute more uniformly across the graph.

We discretize the continuous flow equations by integrating the dynamics with a $\Delta t = 0.01$. A summary of each dynamics is provided in Table 7.

Family	Flow class	Regime signal	Propagation pattern
$\mathbb{R}_2$	Saturating regulatory influence	Degree-averting	Peripheral pathways dominate
$\mathbb{P}$	Self-limited growth with neighbor reinforcement	Degree-driven	Hubs dominate propagation
$\mathbb{E}$	Recovery and susceptibility-limited contagion	Homogeneous	Propagation is broadly distributed

Table 7: **Network interaction regimes.** The three network dynamics differ in the mechanism by which information propagates over the same graph topology.

This dataset is the most direct synthetic test of graph-aware regime inference. A velocity-only method may miss these regimes because the distinction lies in how the graph mediates propagation, not only in marginal feature changes.

12.4.2 Real-World Datasets

SportVU NBA We use the SportVU NBA player-tracking dataset to evaluate regime-aware forecasting in a multi-agent setting with strong interaction structure. The dataset consists of professional

basketball tracking sequences annotated with play outcomes. We consider two outcome categories: scoring plays, which end in a made basket, and rebounding plays, which follow a missed shot. These outcome categories define sequence-level regimes, which remain fixed over the prediction horizon.

The dataset is available in [Linou et al., 2016]. We use the train/test split provided by the authors of [Yue et al., 2014]. To construct the final benchmark, we pool scoring and rebounding sequences while preserving the original train/test partition. No additional outcome relabeling is performed. The interaction graph is defined as a fully connected graph. For each node, we use the spatial coordinates (x, y) together with a static entity-type descriptor indicating whether the node corresponds to a player or the ball. Edge features encode pairwise relational attributes, including the $\mathcal{L}_2$ distance between entities and a team-membership indicator: 1 for entities on the same team, 0 for player–ball edges, and -1 for players on opposing teams. This indicator is treated as a $\mathbb{Z}_2$-symmetric relational field. The edge $\mathcal{L}_2$ distance and spatial coordinates are the only time-varying features. The preprocessing pipeline used to construct the trajectories and regime labels is provided in our code.

AMASS with BABEL Annotations We use AMASS with BABEL annotations to evaluate regime-aware forecasting under time-varying action structure. AMASS provides articulated human motion-capture trajectories, while BABEL provides temporally aligned action annotations. In contrast to SportVU NBA, where the regime is fixed for an entire sequence, BABEL action labels may change within a sequence. This setting therefore evaluates whether a model can track changes in the latent action regime over time while forecasting joint-level motion.

The AMASS motion data are available from [Mahmood et al., 2019b]. Since AMASS aggregates multiple motion-capture sources, we restrict our experiments to four large and commonly used subsets: BMLmovi and BMLrub [Ghorbani et al., 2020], CMU [Carnegie Mellon University, 2016], and KIT [Mandery et al., 2015]. We pair these motion sequences with BABEL annotations, available from [Punnakkal et al., 2021b], and map fine-grained action labels to a smaller set of dominant coarse regimes. Specifically, as shown in Table 8 we retain four action groups: step/walk, stand, turn, and sit/lower. We drop all the time steps or motion sequences that had no actions falling in the associated pooling. The stemming, grouping, filtering, and trajectory preprocessing steps are provided in our code. Unless otherwise stated, action labels are used only for evaluation. The graphs are created using the SMPL 22 joints skeleton [Loper et al., 2015]. For each node, we use the spatial coordinates (x, y) together with static anatomical descriptors: joint type, left/right laterality, and distance to the pelvis. Edge features represent static relational attributes, including limb membership (e.g., arm or leg), fixed inter-joint distance, and left/right laterality. The spatial coordinates are the only time-varying features.

Regime type	Example of Action Names	Global Motion Pattern
Step/Walk	‘walk away‘, ‘slow walk‘, ‘step forward‘, ...	Directional feet movement
Stand	‘look‘, ‘poses‘, ‘wait‘, ...	Most body static
Turn	‘twist body to the left‘, ‘rotate‘, ‘twirl‘, ...	Body rotating
Sit/Lower	‘sit‘, ‘lean‘, ‘crouch‘, ...	Body joints lowering to the ground

Table 8: **AMASS/BABEL regimes.** The four dynamics corresponds to dominant motion patterns in the AMASS [Mahmood et al., 2019a] dataset aligned with BABEL labels [Punnakkal et al., 2021a].

12.5 Complementary Experiments

12.5.1 Training results and statistical significance

Training reported as the results of the four folds cross validation with best hyperparameters.

12.5.2 Quantitative regime interventions

We test whether HERON's learned regimes encode functionally distinct dynamics by intervening on the regime used during rollout. For each ground-truth regime (j), we roll out the corresponding sequences while clamping the model to regime (i), average the resulting ADE, and normalize each column by the matched intervention $(i = j)$ (diagonal $= 1$; lower is better). The matched regime achieves the lowest ADE on every dataset. Incorrect interventions increase error by 15–83% on

Model	Synthetic			Real-world	
	Linear	Gompertz [Gompertz, 1825]	Network [Harush and Barzel, 2017]	NBA [Linou et al., 2016]	AMASS [Mahmood et al., 2019a]
Persistence (Static)	5.518 (±0.055)	2.498 (±0.025)	0.103 (±0.001)	4.51 (±0.07) / 7.78 (±0.13)	0.092 (±0.002) / 0.133 (±0.003)
Persistence (Velocity)	2.911 (±0.029)	2.881 (±0.029)	0.057 (±0.001)	2.51 (±0.05) / 5.81 (±0.10)	0.131 (±0.003) / 0.276 (±0.006)
S4 Gu et al. [2022]	4.156 (±0.125)	4.378 (±0.131)	0.165 (±0.006)	2.13 (±0.08) / 5.04 (±0.18)	0.428 (±0.018) / 0.624 (±0.028)
rSLDS Glaser et al. [2020]	2.720 (± 0.058)	3.133 (± 0.047)	0.054 (± 0.001)	2.93 (± 0.02) / 5.40 (± 0.05)	0.073 (± 0.004) / 0.133 (± 0.006)
RED-SDS Ansari et al. [2021]	3.388 (± 0.031)	3.152 (± 0.128)	0.046 (± 0.002)	4.72 (± 0.07) / 8.36 (± 0.13)	0.071 (± 0.002) / 0.119 (± 0.002)
TESTAM Lee and Ko [2024]	2.923 (± 0.198)	2.679 (± 0.165)	0.042 (± 0.002)	2.55 (± 0.09) / 4.57 (± 0.21)	0.051 (± 0.004) / 0.093 (± 0.003)
EvolveGCN [Pareja et al., 2020]	3.398 (±0.071)	2.654 (±0.054)	0.056 (±0.001)	1.74 (±0.06) / 4.97 (±0.15)	0.059 (±0.002) / 0.087 (±0.003)
LaM-SLidE Sestak et al. [2025]	2.083 (±0.043)	N/A	0.053 (±0.001)	**0.74** (±0.03) / **1.30** (±0.06)	0.062 (±0.004) / 0.085 (±0.003)
GraphSSM Li et al. [2024b]	2.521 (±0.052)	2.156 (±0.046)	0.050 (±0.001)	1.66 (±0.06) / 4.62 (±0.15)	0.057 (±0.002) / 0.083 (±0.003)
GraphSSM+HERON Li et al. [2024b]	2.481 (±0.049)	2.061 (±0.041)	0.046 (±0.001)	1.58 (±0.05) / 4.39 (±0.13)	0.054 (±0.002) / 0.079 (±0.003)
HERON	**1.921** (±0.038)	**1.945** (±0.038)	**0.038** (±0.001)	0.80 (±0.03) / 1.51 (±0.06)	**0.046** (±0.003) / **0.067** (±0.002)

Table 9: **Regime-conditioned forecasting performance.** Train ADE is reported for generated datasets, and ADE/FDE for motion datasets. Values are mean (± standard deviation) over 4 folds. Best results are in bold and second-best underlined. N/A means that the method was not applicable.

Method	Synthetic			Real-world	
	Linear	Gompertz	Network	NBA	AMASS
Latent-HMM	**0.979** (±0.051)	**0.965** (±0.044)	0.716 (±0.023)	0.604 (±0.016)	0.329 (±0.017)
rSLDS Glaser et al. [2020]	0.958 (±0.059)	0.946 (±0.016)	0.331 (±0.024)	0.362 (±0.019)	0.330 (±0.025)
RED-SDS Ansari et al. [2021]	0.946 (±0.026)	0.939 (±0.019)	0.570 (±0.007)	0.493 (±0.023)	0.318 (±0.017)
HERON	0.948 (±0.031)	0.931 (±0.012)	**0.920** (±0.009)	**0.878** (±0.011)	**0.419** (±0.018)

Table 10: **Self-supervised regime recovery.** Train macro-F1 quantifies how well the learned regimes align with ground-truth dynamical labels, which are used only for evaluation. Best results are in bold and second-best results are underlined. Values are mean (± standard deviation) over 4 folds.

Network and 16–23% on NBA, indicating that the experts encode non-interchangeable relational dynamics rather than redundant parameterizations. The effect is smaller on AMASS (2–31%), consistent with its smooth and partially overlapping motion dynamics. Nevertheless, clearly distinct modes remain well separated: for example, intervening between Stand and Lower increases ADE by 28–31%. See Table 11 for the extensive list of all results. Overall, changing the inferred regime systematically alters the predicted dynamics, while the matched regime yields the best single-regime rollout, providing quantitative evidence that HERON's regimes meaningfully govern the generated trajectories.

We further evaluate whether regime interventions preserve physically plausible rollouts, i.e., whether alternative regimes modify the dynamics without producing anatomically inconsistent human motion on AMASS or implausible player trajectories on NBA (see Table 12 and 13 respectively). On AMASS, we measure relative bone-length violation, which captures deviations from skeletal consistency, and per-joint jerk normalized by real motion, which measures whether predicted trajectories remain comparably smooth. On NBA, we report the fraction of predicted positions outside the court and player acceleration relative to real trajectories, capturing spatial validity and kinematic realism. All reported statistics are computed over the test set. Although incorrect regime interventions increase ADE, their constraint statistics remain close to those of matched-regime rollouts: bone-length violation changes from 2.51% to 2.66% and normalized jerk from 2.89 to 2.87 on AMASS, while the out-of-court rate changes from 3.37% to 3.65% and normalized acceleration from 1.19 to 1.20 on NBA. Thus, clamping an alternative regime changes the predicted dynamics without substantially reducing their physical or kinematic plausibility. Combined with the ADE intervention results, this shows that HERON produces distinct yet similarly constraint-compliant dynamical alternatives. These are interventions on the model's latent regime, not identified causal interventions on the real-world process; they quantify whether the learned experts control distinct predictive dynamics.

12.5.3 Imbalanced regime frequencies

We evaluate HERON on a controlled variant of the Network dataset with regime frequencies 70%/20%/10%. Under the original balance loss, which matches aggregate regime usage to a uniform distribution, HERON fails to recover the rarest regime (minority recall 0.00). We therefore replace the fixed uniform target with a categorical prior learned jointly from the training data, without regime supervision. This modification substantially recovers the minority regime (recall 0.70) and improves

Network

	True regime		
Clamped regime	Regulatory	Epidemic	Mutualistic
Regulatory	1.00	1.42	1.19
Epidemic	1.83	1.00	1.29
Mutualistic	1.21	1.15	1.00

AMASS

	True regime			
Clamped regime	Rotation	Step	Stand	Lower
Rotation	1.00	1.03	1.09	1.04
Step	1.02	1.00	1.15	1.14
Stand	1.07	1.19	1.00	1.31
Lower	1.17	1.11	1.28	1.00

NBA

	True regime	
Clamped regime	Basket	Rebound
Basket	1.00	1.16
Rebound	1.23	1.00

Table 11: **Forecasting results with regime clamping.** Rows indicate the regime imposed during forecasting, while columns indicate the true regime. ADE is normalized column-wise by the matched intervention, such that diagonal entries equal 1.

AMASS condition	Bone-length violation (%)	Motion jerk ($\times$ real)
Ground-truth future	0.00	1.00
Clamp = correct regime	2.51	2.89
Clamp = wrong regime	2.66	2.87

Table 12: **Constraint violation with regime clamping on AMASS.**

macro-F1 from 0.60 to 0.78 and regime AUC from 0.62 to 0.97, while also reducing ADE from 0.049 to 0.044 (see Table 14). These results show that replacing the fixed uniform prior with a learned one substantially improves HERON under strongly imbalanced regimes, suggesting that the uniform regime-distribution assumption can be relaxed.

12.5.4 Overlapping regimes and soft routing

To test whether the one-active-regime assumption is restrictive, we generate sequences from convex mixtures of two regimes with known weights $w = (\lambda, 1-\lambda)$, and compare the original hard rollout gate, $r = \mathrm{onehot}(\arg\max(p))$, with soft routing, $r = p$. The hard- and soft-routing variants are trained separately under the same protocol (see Table 15 for detailed results). Both recover the mixture similarly (posterior–mixture agreement accuracy: 0.85 vs. 0.87), but soft routing better preserves it during generation: rollout-weight error decreases from 0.28 to 0.13 (-54%), and ADE on mixed sequences improves from 2.02 to 1.85 (-8.4%). The trade-off on pure sequences is modest (ADE 2.22 to 2.32, $+4.5\%$), where hard selection correctly matches a single active regime. Thus, HERON's posterior already captures mixed regime membership. The limitation arises from collapsing it to one expert at rollout. Posterior-weighted routing, which combines expert predictions according to the inferred regime probabilities, provides a straightforward extension for settings with coexisting dynamics. This controlled experiment addresses simultaneous mixtures; continuous temporal transitions remain approximated under the current discrete prior, although soft routing offers one practical extension.

NBA condition	Out-of-court (%)	Player acceleration ($\times$ real)
Ground-truth future	2.21	1.00
Clamp = correct regime	3.37	1.19
Clamp = wrong regime	3.65	1.20

Table 13: **Constraint violation with regime clamping on NBA.**

Model	Test macro-F1	Minority recall	Regime AUC	ADE
Default HERON (uniform balance)	0.60	0.00	0.62	0.049
HERON + learnable prior	0.78	0.70	0.97	0.044

Table 14: **Imbalanced regimes.** Controlled Network variant with regime frequencies 70%/20%/10%.

12.5.5 AMASS regime-recovery errors

AMASS recovery is primarily limited near action boundaries, where BABEL's semantic labels are an imperfect proxy for instantaneous motion dynamics. In particular, the labeling frequently uses the Transition label for short and heterogeneous changes between most action pairs and therefore does not define a coherent dynamical mode. In the preprocessing pipeline used for the main-paper results, we exclude this label from the retained action set and map its frames to the preceding action. Moreover, physical motions change smoothly, whereas annotations switch discretely, so HERON generally requires several observations before identifying the new mode. Consistent with this explanation, 38% of errors occur within 5 frames of a label transition, and macro-F1 increases from 0.412 overall to 0.566 on stable motion segments. Manual inspection also reveals semantic boundary ambiguities, such as turning motions being grouped with either rotation or stepping. These results indicate that label-dynamics mismatch is an important contributor to the lower aggregate recovery score, particularly near ambiguous transitions, while the remaining errors on stable segments also reflect residual model limitations.

12.5.6 Regime reproducibility across runs

To test whether HERON recovers reproducible dynamical modes rather than seed-specific partitions, we trained five independent seeds and evaluated all ten run pairs on the same held-out sequences. We report mean $\pm$ standard deviation across these ten pairwise comparisons for: (i) ARI, which measures agreement between frame partitions independently of regime labels and adjusts for chance; (ii) matched frame accuracy, the fraction of frames assigned to the same regime after a single global Hungarian alignment; and (iii) worst-mode Jaccard, the minimum intersection-over-union between the frame sets assigned to any pair of matched regimes, thereby exposing instability in the least reproducible mode. Null ARI is the 95th percentile over 1,000 circular-shift surrogates, which destroy cross-run alignment while preserving regime frequencies and segment durations. Because pairs sharing a run are not statistically independent, these statistics characterize cross-seed variability rather than independent-sample uncertainty. Results are presented in Table 16.

HERON is highly consistent on Network and NBA, reaching ARI 0.92/0.83, matched accuracy 0.98/0.96, and worst-mode Jaccard 0.96/0.90, respectively. Consistency is lower on AMASS, in line with its weaker alignment to the coarse semantic labels and the ambiguity induced by smooth motion transitions. Nevertheless, independent runs recover meaningfully similar partitions: ARI reaches 0.48, compared with a temporal-null threshold of 0.005, while matched accuracy and worst-mode Jaccard remain 0.61 and 0.57. Thus, HERON repeatedly discovers similar dynamical patterns on AMASS, although this structure is only partially aligned with the four annotated action classes.

12.5.7 GraphSSM enhanced with HERON

Motivation. Section 3.2 claims that the HERON regime pathway can wrap any latent graph-temporal backbone exposing an entity-level state. We instantiate this on GraphSSM [Li et al., 2024b], where each node trajectory is propagated by a structured SSM kernel with graph-based mixing. GraphSSM is competitive on local trajectories (Table 9) but offers no system-level mode: in regime-

Setting	ADE (pure)	ADE (mixed)	Posterior calibration	Rollout-weight error
HERON – hard gate	2.22	2.02	0.85	0.28
HERON – soft gate	2.32	1.85	0.87	0.13

Table 15: **Overlapping regimes.** Hard versus posterior-weighted soft routing.

Dataset	K	Inter-run ARI	Matched accuracy	Worst-mode Jaccard	Null ARI (95th %ile)
Network	3	0.92 ± 0.02	0.98 ± 0.02	0.96 ± 0.03	0.005
NBA	2	0.83 ± 0.04	0.96 ± 0.01	0.90 ± 0.02	0.004
AMASS	4	0.48 ± 0.07	0.61 ± 0.08	0.57 ± 0.05	0.005

Table 16: **Cross-seed regime reproducibility.** Statistics over all ten pairs from five independent seeds.

structured data, the active mode is silently absorbed into the per-node SSM state, foreclosing regime tracking and the interventional analysis of Section 6.

Component mapping. We identify the GraphSSM hidden state with the entity-level deterministic state h_t^i of Section 3.1, and let the GraphSSM update play the role of the entity-level transition. Three HERON modules are added around it:

1. A graph-level GRU $h_t^G \in \mathbb{R}^{d_H}$ and a categorical regime $r_t \in \{1, \dots, K\}$.
2. Hierarchical regime inference: attention pooling [Lee et al., 2019] over the GraphSSM node embeddings produces emb_t^G; the posterior $q_\phi(r_t \mid h_t^G, \mathrm{emb}_t^G)$ is sampled via Gumbel–Softmax with straight-through estimator; a separate ψ-module supplies the prior $p_\psi(\hat{r}_t \mid h_t^G, \widehat{\mathrm{emb}}_t^G)$ used at rollout.
3. A regime-gated MoE producing the entity-level latent z_t^i, with one expert per regime.

The forward pass is Algorithm 1 with the GraphSSM kernel substituted for the entity-level transition operator.

Implementation. We use GraphSSM-S4 [Li et al., 2024b] stacked to the depth of our HERON encoder. Regime cardinality matches standalone HERON (K=3 synthetic, K=2 NBA, K=4 AMASS). All non-backbone hyperparameters follow Table 20. We explicitely matched the capacity of GraphSSM and its counterpart with the HERON module on top, so that any gain in performances cannot be attributed to a pure increase in capacity.

Forecasting. GraphSSM+HERON improves over vanilla GraphSSM on every benchmark (Table 9: ADE drops by 1.7% (Linear), 6.2% (Gompertz), 4.1% (Network); ADE/FDE drop by $3.6\%/3.5\%$ (NBA) and $1.8\%/2.4\%$ (AMASS). The largest gains arise on the most clearly regime-structured benchmarks (Gompertz, Network), supporting the central claim that an explicit regime variable pays off when latent dynamical modes are present. GraphSSM+HERON does *not* match standalone HERON (e.g. 0.047 vs. 0.038 on Network), indicating that the SSM kernels and the regime-conditioned MoE supply complementary rather than redundant inductive biases.

Regime tracking. Table 17 shows that plugging HERON on top of GraphSSM allowed to retrieve regimes, even if performances are slightly degraded compared to pure HERON.

Takeaway. The regime interface adds as a lightweight wrapper to a strong latent graph-temporal backbone without rewriting it, yielding concurrent gains in forecasting and regime tracking.

12.5.8 Sensitivity to mis-specification of K

To assess HERON's sensitivity to misspecification of the number of regimes, we retrained the model with K below and above the reference value. Forecasting is substantially less sensitive to K than

	Synthetic			Real-world	
Metric	Linear	Gompertz	Network	NBA	AMASS
Macro-F1	0.766	0.703	0.790	0.872	0.32
AUC	0.703	0.944	0.868	0.899	0.408

Table 17: **Self-supervised regime recovery.** Test macro-F1 and AUC quantify how well learned regimes align with ground-truth dynamical labels, which are used only for evaluation.

regime recovery (see Table 18 for detailed results). On Network, over-specifying $K = 4$ preserves the reference-K ADE of 0.038, while only three experts remain active. Even at $K = 5$, ADE is 0.041, with only three experts used. Similarly, NBA models with $K = 3$ or 4 activate only two experts, obtaining ADE 0.89 and 0.84, respectively, versus 0.81 at $K = 2$. Under-specification affects regime identification more strongly. On Network, $K = 2$ increases ADE moderately from 0.038 to 0.043, but reduces macro-F1 from 0.91 to 0.55. On AMASS, $K = 3$ gives the best ADE/FDE, whereas the reference $K = 4$ gives the best macro-F1/AUC. This likely reflects AMASS's smooth transitions and partially overlapping motion patterns, for which the number of predictive dynamical modes need not coincide with the number of semantic action labels. Overall, forecasting is relatively robust to moderate over-specification, whereas reference-label-aligned recovery remains sensitive to both under- and over-specification. We therefore use active-expert utilization together with the validation criterion described in Appendix 12.5.9 to select K in practice.

	Network					NBA				
K	ADE	macro-F1	AUC	Experts used	K	ADE	FDE	macro-F1	AUC	Experts used
2	0.043	0.55	0.75	2	**2**	**0.81**	**1.49**	**0.89**	**0.91**	2
3	**0.038**	**0.91**	**0.97**	3	3	0.89	1.73	0.83	0.86	2
4	0.038	0.81	0.88	3	4	0.84	1.67	0.87	0.87	2
5	0.041	0.73	0.76	3						

	AMASS				
K	ADE	FDE	macro-F1	AUC	Experts used
2	0.051	0.080	0.27	0.49	2
3	**0.044**	**0.059**	0.29	0.48	3
4	0.047	0.066	**0.41**	**0.57**	4
5	0.051	0.085	0.27	0.52	5

Table 18: **Sensitivity to misspecification of the number of regimes.** Bold K marks the reference regime count; metric bolding follows the best value in each dataset.

12.5.9 Selecting the number of regimes without labels

To select K without regime labels or domain knowledge, we train HERON for $K \in \{1, \ldots, 6\}$ and evaluate the predictive NLL of held-out validation futures. We then select the elbow using a fixed rule: after rescaling the curve, we choose the point of maximum deviation from the line joining its endpoints, marking where further increases in K yield diminishing returns. This recovers the reference $K = 3/2/4$ on Network/NBA/AMASS, whereas the paper's velocity heuristic selects 2/4/5 (results presented in Table 19).

As a complementary diagnostic, we estimate the effective K of an overcomplete HERON model trained with $K = 6$. We first remove experts whose usage falls below 10%, then compare the remaining experts through rollouts clamped from the same validation states. We normalize pairwise forecast disagreements by the median among surviving experts, so 1 denotes a typical pair. Under the primary rule, pairs below 0.5 are connected in a redundancy graph, and connected components define the effective K. This yields 3/2/5 distinct experts. On AMASS, the additional effective mode suggests specialization to locally distinct motion patterns, while held-out likelihood still favors $K = 4$. Thus, predictive likelihood provides a practical label-free selection rule, while effective K identifies inactive or dynamically redundant experts.

Selection rule	Network ($K = 3$)	NBA ($K = 2$)	AMASS ($K = 4$)
Expert usage	3	2	6
Effective-K	3	2	5
Held-out likelihood	**3**	**2**	**4**
Velocity baseline	2	4	5

Table 19: **Label-free K selection and diagnostics.** Reference regime counts are shown in parentheses.

12.6 Hyperparameters

We list in this section the set of optimal hyperparameters for the framework that resulted from the random search performed with the cross-validation.

Hyperparameter	**Synthetic Datasets**	**NBA Dataset**	**AMASS Dataset**
batch size	256	64	128
entity embedding dim	32	64	64
encoder hidden dim	48	32	32
encoder num layers	2	2	4
entity hidden dim	32	48	16
system hidden dim	32	32	32
latent dim	8	32	16
regime hidden dim	48	64	32
expert hidden dim	16	32	32
τ_{max}	1.2	1	1
τ_{min}	0.5	0.3	0.5
prediction weight	5.0	5.0	2.0
prior prediction weight	0.5	0.5	0.5
dynamics kl weight	0.8	0.8	0.8
representation kl weight	0.2	0.8	0.2
structural weight	2.0	1.0	2.0
entropy weight	0.2	0.05	0.05
orthogonality weight	0.05	0.01	0.01

Table 20: **Hyperparameters for all datasets.** List of the relevant hyperparameters and their values per dataset.

12.7 Model Size and Computational Budget

Table 21 reports trainable parameter counts for the implemented models and configurations. Parameter counts are provided to make clear that the forecasting results are not obtained by comparing HERON only against much smaller baselines.

Training was performed on a single A100 40GB GPU using four data-loading workers. Depending on the dataset, training took approximately 10 to 90 minutes.

Cost of expert specialization. Each expert is a small head (3-layer MLPs, the full K-expert bank is around 12% of the model). All K experts are evaluated at every step regardless of the decorrelation term, because the transition forms a straight-through mixture $\sum_k w_k \, \mathrm{expert}_k$ over experts; their outputs are already materialized in the forward pass. The decorrelation regularizer therefore adds no extra expert evaluations; it only reads those existing outputs and computes a $K \times K$ similarity, which is negligible. The cost of evaluating all experts rather than a single hard-selected one is a lightweight-head overhead of approximately 7% wall-clock on GPU.

Model	Dataset/configuration	Trainable parameters
GraphSSM	Synthetic	300K
GraphSSM	NBA	450K
GraphSSM	AMASS	450K
HERON	Synthetic	400K
HERON	NBA	600K
HERON	AMASS	600K
EvolveGCN	Synthetic	300K
EvolveGCN	NBA	800K
EvolveGCN	AMASS	800K
S4	Synthetic	600K
S4	NBA	1.7M
S4	AMASS	1.7M
LaM-SLidE	Synthetic	1.7M
LaM-SLidE	NBA	2.1M
LaM-SLidE	AMASS	2.1M
Persistence	Any dataset	0

Table 21: **Trainable parameter counts.** Counts correspond to the instantiated configurations used in the experiments.

12.8 Limitations

12.8.1 MoE scales linearly with K.

Despite its alignment with our problem formulation as explained in section 12.1.2, a limitation of the choice of MoE as the modulation mechanism is that the number of expert parameters scales linearly with the number of regimes K. This scaling is not specific to our framework, but follows from explicitly representing K separate transition mechanisms. When K is moderate and the regimes have interpretable semantics, this cost is acceptable and provides the benefit of separately parameterized dynamics. However, for settings with a large number of regimes, or with regimes that vary more continuously, lighter conditioning mechanisms may be preferable. One possible alternative is FiLM-style modulation [Perez et al., 2018], where a shared transition model is modulated by the regime variable. Such an approach would preserve regime-dependent dynamics while largely decoupling the size of the main transition model from K. The specialization machinery used in our framework could in principle be retained, since the corresponding objectives are defined on expert or transition outputs rather than on a specific internal parameterization.

This alternative comes with a trade-off. FiLM-style modulation does not provide separately parameterized experts, and therefore cannot directly encode different architectural inductive biases across regimes. For example, one transition function could be parameterized as a GNN while another could use a Transformer architecture [Vaswani et al., 2017]. Similarly, when the form of the dynamics is partially known, or when a known basis of transition functions is available for modeling flows on the network, an explicit MoE parameterization remains a natural choice. We therefore view the MoE design as appropriate for the regime-structured setting considered in this work, while leaving a systematic comparison with lighter modulation mechanisms to future work.

12.8.2 Losses requires balance between weights.

Beyond the prediction objective, our training objective includes terms that encourage expert usage, expert specialization, prior–posterior alignment, and prevention of posterior collapse. These terms introduce additional hyperparameters, and the behavior of the model depends on their relative weighting. If the usage or specialization terms are too weak, the model may fail to assign distinct dynamics to different experts. Conversely, if these terms dominate the predictive objective, specialization can be encouraged at the expense of accurate forecasting. Similarly, insufficient prior–posterior alignment can lead to unreliable open-loop rollouts, while excessive pressure on the posterior may contribute to

collapse. These failure modes are well known in latent dynamics models and structured representation learning [Hafner et al., 2021, 2025, Guo et al., 2025].

We include several safeguards to reduce this sensitivity. First, we adopt the KL balancing mechanism introduced in [Hafner et al., 2021], which stabilizes prior–posterior alignment by discouraging the posterior from moving into regions that the prior cannot reach. Second, in our experiments we observed that the structural objectives require primarily a stable relative balance rather than independent per-dataset tuning. In particular, a fixed ratio between the expert-usage and specialization terms was effective across the datasets considered, with only global scaling needed to balance these structural losses against the prediction loss. From Section 12.6, a ratio of $\frac{1}{4}$ is constant over the entropy and orthogonality weights, where the structural weights tune generally the constraint put on the model to restrict its latent space. This is consistent with the role of these terms: they jointly encourage peaked assignments at the sample level while maintaining balanced expert usage at the population level which naturally comes with a tradeoff and balance. We also observed limited sensitivity around the $0.8/0.2$ KL-balancing weights used in our experiments. They were consistently selected as best threshold in the hyperparameters search 12.6.

These observations do not remove the need for validation when transferring the framework to substantially different domains, larger values of K, or noisier regimes. The loss weights remain hyperparameters, and incorrect balancing can still lead to underused experts, weak specialization, or inaccurate rollouts. In practice, these issues can be monitored through expert-usage statistics, assignment entropy, specialization metrics, and prior–posterior KL values. A more exhaustive study of automatic loss-weight adaptation is left for future work.

12.8.3 Uniform regime usage is restrictive under severe imbalance.

The submitted uniform balance objective is unsuitable under severe imbalance: minority recall falls to zero in the controlled setting of Appendix 12.5.3. Replacing its fixed target with an unsupervised learned categorical prior recovers minority recall to 0.70 and improves macro-F1 from 0.60 to 0.78. We therefore view the learned-prior formulation as an extension and explicitly retain the limitation of the original uniform objective.

12.8.4 One active regime at rollout.

The current hard rollout gate assumes that one regime is active at each time step. The controlled experiment in Appendix 12.5.4 shows that HERON's posterior already captures mixed regime membership, while the limitation arises from collapsing it to one expert at rollout. Posterior-weighted routing provides a straightforward extension for settings with coexisting dynamics. This experiment addresses simultaneous mixtures; continuous temporal transitions remain approximated under the current discrete prior.

12.8.5 Fixed target topology.

HERON currently models regime-dependent dynamics on a fixed target topology and does not generate node/edge birth or death events. Topology-changing generation is outside the scope of this work, but can be envisioned as an interesting future extension

12.8.6 Number of regimes K is a model-selection hyperparameter.

HERON treats the number of regimes K as a fixed model-selection hyperparameter. When regime labels or domain knowledge are unavailable, Appendix 12.5.9 selects K using the predictive NLL of held-out validation futures and a fixed elbow rule. This recovers the reference $K = 3/2/4$ on Network/NBA/AMASS, whereas the velocity heuristic selects 2/4/5. The sensitivity analysis in Appendix 12.5.8 further shows that forecasting is relatively robust to moderate over-specification, whereas reference-label-aligned recovery remains sensitive to both under- and over-specification. Fully adaptive regime discovery, where new regimes can emerge or be pruned during training, remains a natural direction for future work.

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: All the claims found in the abstract and introduction are supported by the results discussed in the paper.

   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: We discuss all the limitations and future work in the dedicated section 9 and Appendix 12.8.

   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 (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - 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.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. 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: [Yes]

Justification: The paper provides a set of detailed definitions that motivate the choice of the developed model. We provide all the necessary information in the Appendix.

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, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- 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 (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The datasets we use are publicly available and we provide details on how to generate the synthetic data in the Appendix. To make our experiments reproducible, the set of all hyperparameters of the models used along with the training specifications are provided in detail in the Appendix.

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: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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: Yes, the full code with detailed instructions for reproducibility have been made public and can be found at `https://github.com/JerBaf/HERON`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide detailed descriptions of all the training parameters, procedure and hyperparameters used for every model that we train in the Appendix.

   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 that is necessary to appreciate the results and make sense of them.
   - 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: We provide all our results in terms of downstream task performance in the format "mean $\pm$ standard deviation from the mean".

   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, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide detailed information about the used resources in the Appendix.

   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 each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

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: We preserve anonymity in our submission and our submission abides by the NeurIPS Code of Ethics.

   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 (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss potential positive and negative societal impacts in Section 11.

    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 potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

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 (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The release of our data and models upon acceptance, does not pose a direct risk of misuse.

    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 to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide citations for all sources of code and/or data used. For the datasets, we reference the original open-source work they are based on, in full accordance with their licensing terms.

    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, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

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: This paper does not release any new assets beyond code, which is adequately documented in order to reproduce all results from the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - 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). You can either create an anonymized URL or include an anonymized zip file.

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 (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved in the experiments conducted for this paper.

    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, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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 to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved in the experiments conducted for this paper.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

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? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not a core component of this paper in terms of methods.

    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 our LLM policy in the NeurIPS handbook for what should or should not be described.