META-CAUSAL PREDICTIVE STATES: A WORLD MODEL FOR HIGH-DIMENSIONAL EVENT SEQUENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

While current temporal foundation models transfer across domains through a shared numerical observation space, discrete event sequences describe system dynamics through symbols whose identities are local to each domain. This limits direct vocabulary sharing even when different systems exhibit similar predictive structure. To address this constraint, we propose a world model for high-dimensional discrete event sequences organized around *meta-causal predictive states*, which groups equivalence classes of histories whose conditional future distributions coincide up to event relabeling. Our main hypothesis is that transferable event dynamics can be expressed through a shared predictive vocabulary, while context binds this vocabulary to local event identities. To realize this hypothesis, we propose two models: (i) DES-JEPA learns discrete event representations from causal occurrence statistics using predictive teacher-student training, (ii) DES-PFN learns in-context dynamics over the frozen representations and decodes event types, enabling autoregressive forecasting without target-task parameter updates. Both stages are pretrained using a mixture of six synthetic process families and evaluated on zero-shot forecasting tasks with hundreds of event types on six real-world domains. Our analysis demonstrates that DES-PFN is competitive for low cardinality dataset and, to our knowledge, the first method to scale zero-shot to hundreds of event types.

1 INTRODUCTION

Event sequences arise ubiquitously across domains, from vehicle diagnostic logs (Math et al., 2025) to electronic health records (Rasmy et al., 2021; Johnson et al., 2023; Labach et al., 2023). Learning across these domains requires handling event vocabularies whose labels have no consistent meaning across datasets; the same event type may denote a machine error in one system and a clinical observation in another. While concurrent work on temporal foundation models (TFMs) transfers across domains through numerical observations (Chronos (Ansari et al., 2024) and TimesFM (Das et al., 2024)), discrete event types provide no equivalent basis for such an aligned observation space. Permuting all event types consistently leaves the underlying prediction problem unchanged; only the labels of the output events are permuted accordingly. A reusable or foundational model, should respect this symmetry while also accommodating a variable number of event types without replacing globally trained output heads whenever a new vocabulary is encountered.

Temporal point processes (TPPs) provide an established framework for event prediction (Mei & Eisner, 2017; Zuo et al., 2020; Bacry et al., 2020). Conventional methods therefore rely on fitting expensive statistical models (e.g., Hawkes processes Laub et al. (2025)) that are known to be impractical at hundreds or thousands of distinct event types. Importantly, they must be fitted from scratch for every new batch of data. Berghaus et al. (2026) address this with a foundation inference model (FIM-PP), enabling *in-context* learning of TPPs without fine-tuning. However, FIM-PP supports at most $K = 22$ event types limiting its scope as a foundation model by restricting its direct application to datasets with larger, more realistic, event vocabularies. Therefore, extending current methodologies to more realistic real-world domains with hundreds of discrete event types remains an important open challenge. We hypothesize that filling this void requires a representation that transfers across processes, rather than a better fit to any individual process. For anchor-free data-generating processes (DGPs), we therefore propose learning a shared vocabulary of recurring dynamical patterns, rather than representations tied to local symbol identities.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

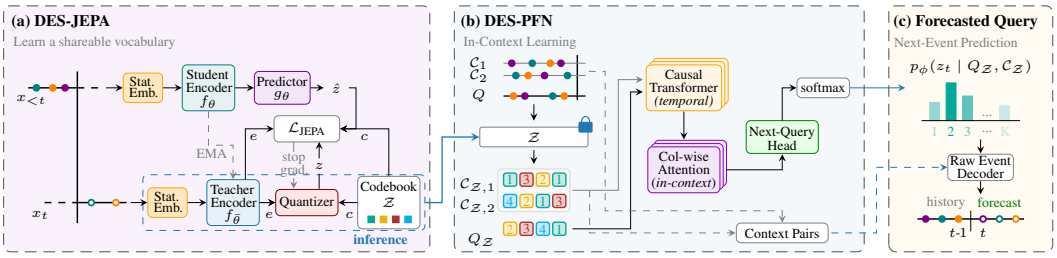

Figure 1: **Pipeline overview.** *(a):* an event sequence is tokenized into a shared codebook $\mathcal{Z}$ via **DES-JEPA**. **DES-PFN** (b) learns in-context to predict the next code of a query Q. *Right (c):* a sequence of interest is forecasted, the resulting code is then decoded into the event types.

To address this gap, we ground our approach in *meta-causal predictive states* (MCPSs), which characterize predictive structure shared by histories across relabeled processes. MCPSs provide an idealized target for transferable representations that preserve the structure governing future evolution independently of local event identities. We approximate this structure through a shared vocabulary of event-level building blocks and a forecasting model that composes them into a contextual predictive states. We propose with **DES-JEPA**, a **j**oint-**e**mbedding **p**redictive **a**rchitecture (LeCun, 2022; Assran et al., 2023; 2025) for **d**iscrete **e**vent **s**equences. DES-JEPA encodes recurring dynamical patterns in a learned shared discrete vocabulary, or *codebook*, reusable across downstream tasks. On top of this codebook, we introduce **DES-PFN**, which from the query history paired with reference sequences constructs a context-dependent representation of evolving dynamics. This allows to perform desired downstream tasks, such as next-event prediction. Both models are pretrained exclusively on synthetic sequences sampled from diverse DGPs. This enables zero-shot forecasting on real-world event sequences across unaligned alphabets without target-task parameter updates. Together, they define an observational world model of future event trajectories, where "causal" denotes predictive-state equivalence in computational mechanics (Shalizi & Crutchfield, 2001). Therefore our contributions are fourfold:

- We formalize a **meta-causal predictive state (MCPS)** as a cross-process predictive abstraction and characterize the information that an approximate representation must preserve, which connects shared event vocabularies to predictive tasks sufficiency.
- We introduce **DES-JEPA**, combining predictive latent learning, vector quantization, and meta-pretraining to learn a **shared codebook** from any discrete event sequences.
- We introduce **DES-PFN**, which uses the shared codebook computed by the teacher encoder of DES-JEPA, to autoregressively predict unseen event types without task-specific fine-tuning.
- We formulate an evaluation of zero-shot transfer, high-cardinality prediction up to 700 event types, and sequence generation, together with targeted tests of shared predictive structure, representation capacity, and dependency preservation of each DGP.

2 Related Work

Event sequence modeling. Neural TPPs model history-dependent event distributions through recurrent or attention-based parameterizations (Mei & Eisner, 2017; Zhang et al., 2020; Li & Sun, 2023). Intensity-free approaches directly model conditional inter-event-time distributions (Shchur et al., 2020), while generative approaches address multi-event forecasting through interacting diffusion processes over event types and times (Zeng et al., 2024), or flow matching over event times (Kerrigan et al., 2026). `EasyTPP` provides a standardized framework for evaluating neural TPP architectures (Xue et al., 2024). Our setting emphasizes forecasting across processes through reference sequences, without target-task parameter updates.

Transfer across temporal processes. Continuous TFMs such as Chronos, TimesFM, and UniTok transfer forecasts through a shared numerical observation space (Ansari et al., 2024; Das et al., 2024; Zhang et al., 2026). PFNs amortize posterior predictive inference over sampled tasks (Müller

et al., 2022). For event sequences, MetaTPP introduces context-based meta-learning (Bae et al., 2023), while FIM-PP learns conditional intensity inference from synthetic processes and transfers to real-world sequences without target-task training (Berghaus et al., 2026). LaT-PFN combines joint-embedding prediction with prior-fitted in-context forecasting in a learned latent space (Verdenius et al., 2024). Event-sequence models such as CoLES, MOTOR, and PinFM learn reusable representations within their respective application domains (Babaev et al., 2022; Steinberg et al., 2024; Chen et al., 2025). In contrast, our approach focuses on learning a shared discrete predictive vocabulary across processes with unaligned event alphabets.

Vocabulary symmetry and predictive discretization. EquiTabPFN and lexinvariant language models address target-label and vocabulary permutations, respectively, but neither learns a shared predictive codebook across stochastic-process families (Arbel et al., 2026; Huang et al., 2023). Vector-quantized and predictive self-supervised methods learn discrete or latent representations whose alignment is inherited from a common visual or acoustic space (van den Oord et al., 2017; Hsu et al., 2021; Assran et al., 2023; Baek et al., 2025); predictive-state, causal-state, and information-bottleneck approaches instead motivate grouping histories by their future behavior (Shalizi & Crutchfield, 2001; Creutzig et al., 2009). DES-JEPA connects these directions by learning a shared discrete prediction space across processes whose native event alphabets are unaligned.

3 Background and Problem Formulation

Notation. We use uppercase letters to denote random variables, e.g., X, and lowercase letters for their realizations, e.g., $x \in \mathcal{X}$, where $\mathcal{X}$ is the event-type alphabet and $P(X)$ denotes the corresponding probability distribution. A stochastic event process is denoted by $\{X_t\}_{t\in\mathbb{N}}$ (Cover & Thomas, 2006), with realization $x_{<t} = (x_0, \ldots, x_{t-1})$ denoting the history before event t. An event sequence with length $L+1$ is written as $S = \{(x_0, t_0), \ldots, (x_L, t_L)\}$, where $x_i \in \mathcal{X}$ is the event type and $t_i \in \mathbb{R}$ its timestamp. We use $H_t = x_{<t}$ for the observed history and $F_t = X_{\geq t}$ for the future. For notational convenience, we omit the subscript t and write H and F when the time step is clear from context. A query is an event sequence whose next event is to be predicted, denoted by Q.

3.1 Local Event Alphabet

We consider a meta-distribution $\mathcal{M}$ over stochastic processes: each draw $m \sim \mathcal{M}$ instantiates a process $\{X_t^{(m)}\}_{t\in\mathbb{N}}$ over its own, process-specific alphabet $\mathcal{X}_m$. The predictive object associated with a history h is its conditional future law $P_m(F \mid H = h)$. Computational mechanics (Shalizi & Crutchfield, 2001) answers the single-process version of this paper. Two histories are *predictively equivalent* when they imply the same distribution over futures:

Definition 1 (Causal state) *For a process P_m, histories $h_1, h_2 \in \mathcal{H}_m$ are causally equivalent, $h_1 \sim_\epsilon h_2$, iff $P_m(F \mid H{=}h_1) = P_m(F \mid H{=}h_2)$. The causal states are the equivalence classes $\mathcal{S}_m = \mathcal{H}_m/\!\sim_\epsilon$, and $\epsilon_m : \mathcal{H}_m \to \mathcal{S}_m$ denotes the resulting (deterministic) causal-state map.*

3.2 Meta-Causal Predictive States

Definition 1 presupposes a single, fixed alphabet $\mathcal{X}_m$. Reconstructing a world-model for any P_m is harder: every $m \sim \mathcal{M}$ has its own alphabet $\mathcal{X}_m$, with no shared meaning between them. Whereas the classical construction defines a family of process-dependent maps $\epsilon_m : \mathcal{H}_m \to \mathcal{S}_m$, we instead seek a single structure over the entire class $\mathcal{M}$. Accordingly, the quotient is taken not only with respect to within-process relabelings, but also to relabelings across distinct processes.

Definition 2 (Admissible alphabet isomorphism) *For $m, m' \sim \mathcal{M}$, an admissible isomorphism is a bijection $\sigma : \mathcal{X}_m \to \mathcal{X}_{m'}$, extended pointwise to sequences and by pushforward to distributions over $\mathcal{X}_m^{\mathbb{N}}$, under which predictive laws are compared.*

Definition 3 (Meta-predictive equivalence and the MCPS space) *Pairs (m, h) and (m', h'), with $h \in \mathcal{H}_m$, $h' \in \mathcal{H}_{m'}$, are meta-predictively equivalent, $(m, h) \sim_{\mathrm{MPS}} (m', h')$, iff there exists an admissible σ (Definition 2) with*

$$\sigma_{\#} P_m(F \mid H{=}h) \;=\; P_{m'}(F \mid H{=}h').$$

*The **meta-causal predictive state space** is the quotient $\mathcal{S}_{\mathcal{M}} = (\mathcal{M} \times \mathcal{H})/\sim_{\mathrm{MPS}}$, with $P_m(F \mid h)$ the predictive law representing an element of $\mathcal{S}_{\mathcal{M}}$ before quotienting.*

Example (Two processes, one MCPS point) Let m have alphabet $\mathcal{X}_m = \{\texttt{login}, \texttt{logout}\}$ and m' have the unrelated alphabet $\mathcal{X}_{m'} = \{\texttt{read}, \texttt{write}\}$. Suppose both are memoryless order-1 chains in which the event simply alternates: after a `login` the next event is always `logout` and vice versa, and after a `read` the next is always `write` and vice versa. Take $\sigma : \mathcal{X}_m \to \mathcal{X}_{m'}$ with $\sigma(\texttt{login})=\texttt{read}$, $\sigma(\texttt{logout})=\texttt{write}$; extended pointwise to sequences, σ maps the $\mathcal{X}_m$-sequence $\texttt{login}, \texttt{logout}, \texttt{login}, \ldots$ to $\texttt{read}, \texttt{write}, \texttt{read}, \ldots$. For $h = (\texttt{login})$ and $h' = (\texttt{read})$, $P_m(F \mid h)$ places all mass on futures starting `logout`, `login`,..., and its *pushforward* $\sigma_{\#}P_m(F \mid h)$ places all mass on the σ-relabeled futures starting `write`, `read`,..., exactly $P_{m'}(F \mid h')$. So $\sigma_{\#}P_m(F \mid h) = P_{m'}(F \mid h')$, hence $(m, h) \sim_{\mathrm{MPS}} (m', h')$ by Definition 3: they collapse to the *same* point of $\mathcal{S}_{\mathcal{M}}$, even though m and m' never share a symbol, because σ ("the event I just saw" $\mapsto$ "its alternating partner") carries one predictive law exactly onto the other. By contrast, $h'' = (\texttt{read}, \texttt{read})$ is impossible under m''s own alternating rule (probability 0), so no admissible σ can relate it to h or h'; it lies in a different, unrelated point of $\mathcal{S}_{\mathcal{M}}$. This illustrates that $\sim_{\mathrm{MPS}}$ merges histories only when an alphabet relabeling makes their *entire future-distributions* coincide, not merely their surface symbols.

Predictive objective. The MCPS provides the process-agnostic predictive object: histories that map to the same state induce the same future law up to admissible event relabeling. A transferable representation should therefore preserve this predictive structure while discarding information tied only to local event identities. In practice, we seek a finite-valued mapping $R : \mathcal{H} \to \mathcal{Z}$ such that the conditional law of future events depends on the history H_t only through $R(H_t)$, while R remains invariant to consistent relabelings of the process-specific alphabet. The following sections instantiate this objective by learning a shared discrete representation of predictive structure and applying it for in-context forecasting on previously unseen processes.

4 METHODOLOGY

The proposed methodology is anchored around the following hypothesis:

Hypothesis 4.1 (Quantization of the MCPS and shared finite vocabulary) *We propose that for a family of redundant, predictable process underlying $\mathcal{M}$, the meta-causal predictive state space $\mathcal{S}_{\mathcal{M}}$ (Def. 3) is finite. Concretely, there exist a finite vocabulary of K symbols and a quantizer q_K*

$$q_K : \mathcal{S}_{\mathcal{M}} \to [K], \qquad Z = q_K \circ \epsilon_{\mathcal{M}}.$$

We hypothesize that, across the process families of interest, predictive structure admits a compact approximation through a shared finite vocabulary of recurring dynamical patterns which we call a *codebook*. It is first learned by **DES-JEPA**. Then, **DES-PFN** uses it for in-context next-event prediction. Both models are pretrained exclusively on synthetic event sequences, enabling zero-shot inference on real-world processes without parameter updates. Fig. 1 summarizes the architecture.

4.1 SYNTHETIC PRETRAINING AND INPUT EMBEDDING

A synthetic prior over event dynamics. Both DES-JEPA and DES-PFN are pretrained *exclusively* on synthetic data covering 6 distinct stochastic process family. Following the recipe of PFNs (Müller et al., 2022; Hollmann et al., 2023; 2025) and Tabular foundation models (QU et al., 2026), every task draws a fresh family with random parameters, such that **no** two tasks, even from the same family, share an identical generator.

For instance, event-types $|\mathcal{X}|$ sampled uniformly from 3 to 1024 event types. Through six distinct processes, we try to cover a wide range of exciting, repeating and deterministic causal mechanisms found in a wide range of sequential data so that the learned codebook generalizes across qualitatively different dynamics rather than memorize any one family. We used a combination of high-order Markov chains (Koller & Friedman, 2009), Hawkes processes Kagan & Knopoff (1978), untrained LSTM Hochreiter & Schmidhuber (1997), hidden markov models Koller & Friedman (2009), bursty renewal processes Barabási (2005) and deterministic Markov chains. A more exhaustive description is provided in Appendix A.

Statistical Embedding. Conventional language models use token-indexed embedding tables (Radford et al., 2018; Touvron et al., 2023), whereas event identities in $\mathcal{X}_m$ have no shared semantic correspondence across processes. We therefore represent each event x_t through causal occurrence statistics computed from the observed history H_t, including the current event. The feature vector $v_t = v(x_t; H_t) \in \mathbb{R}^{d_u}$ comprises empirical surprisal, global and local-window frequencies, a first-occurrence indicator, and relative occurrence times. An MLP R_θ projects these features to the encoder dimension:

$$u_t = R_\theta(v_t) \in \mathbb{R}^d. \tag{1}$$

Because these statistics depend on event occurrences rather than identifier values, the resulting embeddings are invariant to consistent relabeling of event types. We attach no further event-level covariates beyond event type and occurrence time. Handcrafted parameterizations, such as classical Hawkes intensity functions (Laub et al., 2025), impose specific assumptions on event dynamics, limiting their applicability across heterogeneous process families. Additional statistics of the input features feeding the event embedding, including pairwise correlations, are provided in Appendix B.1.

4.2 DES-JEPA: Learning a shared vocabulary

DES-JEPA realizes our hypothesis that modeling discrete event sequences requires a representation that transfers across processes, rather than a better fit to any individual process.

4.2.1 Joint-Embedding Predictive Architecture (JEPA)

We adopt a joint-embedding predictive architecture (JEPA) (LeCun, 2022; Assran et al., 2023; 2025) to learn representations organized by predictive structure rather than predicting individual event types for individual processes. We propose a **d**iscrete **e**vent **s**equence-JEPA (DES-JEPA), which by predicting teacher-derived target codes from preceding observations learns a shared latent space (codebook) without requiring process-specific event labels as prediction targets.

Let $\mathcal{Z} = \{1, \ldots, K\}$ denote the *codebook* of Hypothesis 4.1, and let

$$\mathcal{E} = (c_1, \ldots, c_K) \in \mathbb{R}^{K \times d}, \qquad c : \mathcal{Z} \to \mathbb{R}^d, \;\; c(k) := c_k,$$

denote the associated **embedding table**; a learned continuous vector $c_k \in \mathbb{R}^d$ for every code $k \in \mathcal{Z}$. Although $\mathcal{Z}$ and $\mathcal{E}$ are in bijection by construction, they play different representational roles: $\mathcal{Z}$ is the discrete code space that the model predicts and indexes into, while $\mathcal{E}$ is the associated embedding space on which nearest-neighbor retrieval is performed.

A **student** encoder f_θ maps past observations to a representation $h = f_\theta(u(x_{<t}))$, which are mapped via a predictor g_θ to logits $\hat{z} = g_\theta(h)$ over $\mathcal{Z}$. A **teacher** encoder $f_{\bar{\theta}}$, updated only by an exponential moving average (EMA) (Assran et al., 2023; 2025), encodes the next raw event x_t into a continuous target $e = f_{\bar{\theta}}(u(x_t)) \in \mathbb{R}^d$, quantized against the embedding table $\mathcal{E}$ by nearest neighbor:

$$z \;=\; k^*(x_t) \;:=\; \arg\min_{k \in \mathcal{Z}} \|e - c(k)\|_2^2 \;\in\; \mathcal{Z}. \tag{2}$$

The resulting code $z \in \mathcal{Z}$ is what g_θ is trained to predict; its embedding $c(z) \in \mathcal{E}$ is what downstream modules (DES-PFN) consume whenever a continuous representation of the code is required.

The student never observes x_t and predicts $\hat{z}$ autoregressively purely from past observations $x_{<t}$. The teacher is evaluated under stop-gradient $\mathrm{sg}[\cdot]$ and never receives gradients directly. This asymmetry prevents both encoders from collapsing to a constant map (Assran et al., 2023). For a single trajectory $x_{<t}$ drawn from $m \sim \mathcal{M}$, we train f_θ, g_θ by minimizing

$$\mathcal{L}_{\text{JEPA}} = \mathbb{E}_{m \sim \mathcal{M}} \, \mathbb{E}_{(x_{<t}, x_t) \sim m} \Big[\, \mathrm{CE}_\epsilon\big(g_\theta(h),\, z\big) \;+\; \big\|\mathrm{sg}[e] - c_z\big\|_2^2 \;+\; \beta \big\| e - \mathrm{sg}[c_z] \big\|_2^2 \, \Big] \tag{3}$$

where CE_ϵ is label-smoothed (ϵ=0.1) cross-entropy. The first term is the main training signal, predicting the next latent code $\hat{z}$ from history $x_{<t}$. The remaining two are the standard VQ-VAE codebook and commitment terms (van den Oord et al., 2017): $\|\mathrm{sg}[e] - c_z\|_2^2$ moves the selected embedding c_z toward the teacher's target e, while the commitment term $\beta\|e - \mathrm{sg}[c_z]\|_2^2$ (β=0.25) would move e toward c_z, but since $e = f_{\bar{\theta}}(u(x_t))$ and $f_{\bar{\theta}}$ is updated only by EMA, this gradient never actually reaches the teacher; only the codebook term has an effective trainable target.

4.2.2 DEAD CODE REVIVAL

Naive VQ training is well known to collapse onto a small subset of indices (van den Oord et al., 2017; Razavi et al., 2019). Thus, we actively enforce codebook utilization by reviving dead codes during DES-JEPA pretraining stage. Following the restart strategy of VQ-VAE-2 (Razavi et al., 2019; Dhariwal et al., 2020), we track an EMA of each code's usage share across training steps τ. Let $\hat{p}_k^{(\tau)}$ denotes the fraction of training step τ's batch assignments landing on code k,

$$\bar{p}_k^{(\tau)} \leftarrow \mu\, \bar{p}_k^{(\tau-1)} + (1-\mu)\, \hat{p}_k^{(\tau)}, \qquad \bar{p}_k^{(0)} = \tfrac{1}{K}, \tag{4}$$

and flag code k as *underused* whenever $\bar{p}_k^{(\tau)}$ falls below a fraction of its fair share $1/K$. A per-code counter d_k accumulates consecutive underused steps and resets to zero the moment k recovers. The code k becomes reset-eligible once it has remained underused for τ_{pat} consecutive training steps (a patience threshold). Unlike a per-step presence test, $\bar{p}_k^{(\tau)}$ is barely perturbed by a single rare assignment, so a code that only survives on noise-level picks is still correctly identified as dead (due to the momentum carried by the EMA). Rather than re-seeding each dead code the moment it becomes reset-eligible, we wait and act on all of them simultaneously. Therefore, a batch reset is triggered only when the fraction of reset-eligible codes exceeds a predefined threshold $\rho = 0.01$:

$$\frac{1}{K}\sum_{k=1}^{K} \mathbb{1}[d_k \geq \tau_{\text{pat}}]; \geq; \rho. \tag{5}$$

4.3 DES-PFN: IN-CONTEXT AUTOREGRESSIVE NEXT-EVENT PREDICTION

Following the creation of the shared codebook $\mathcal{Z}$, we use it as a common predictive vocabulary across processes with otherwise unaligned event alphabets. DES-PFN is a PFN that learns to infer the predictive dynamics of an unseen process from a finite reference context, without updating its parameters at inference time.

4.3.1 IN-CONTEXT FORECASTING

For a process $m \sim \mathcal{M}$, let $\mathcal{C} = \{S^{(j)}\}_{j=1}^{N_c}$ be a reference context consisting of sequences drawn from the same process. At prediction time t, the observed query is the prefix $Q_t = \{(x_i, t_i)\}_{i=1}^{t}$ which corresponds to the history $H_{t+1} = X_{\leq t}$ in the notation of Section 3. The forecasting objective is therefore to approximate the conditional future law for a given process m with associated P_m

$$P_m(F \mid Q_t, \mathcal{C}), \qquad F = X_{\geq t+1}. \tag{6}$$

From a PFN perspective Müller et al. (2022); Hollmann et al. (2023), the context provides information about the unknown process m, yielding the posterior predictive distribution

$$p(F \mid Q_t, \mathcal{C}) = \int p_m(F \mid Q_t)\, p(m \mid \mathcal{C})\, dm, \tag{7}$$

which DES-PFN learns to approximate amortized over the process prior $\mathcal{M}$. After pretraining of DES-JEPA, the codebook $\mathcal{Z}$ is frozen and used to tokenize the observed query and reference sequences via the teacher path of equation 2. We obtain

$$Q_{t,\mathcal{Z}} = (z_1, \ldots, z_t), \qquad \mathcal{C}_{\mathcal{Z}} = \{S_{\mathcal{Z}}^{(j)}\}_{j=1}^{N_c}, \qquad z_i \in \mathcal{Z}.$$

DES-PFN maps these observations to a context-conditioned predictive representation $h_\phi = f_\phi(Q_{t,\mathcal{Z}}, \mathcal{C}_{\mathcal{Z}})$. For the temporal component, we discretize the inter-arrival time $\Delta t_{t+1} = t_{t+1} - t_t$ using a typical logarithmic binning scheme (Ansari et al., 2024; Math et al., 2025),

$$\tau_{t+1} = \text{bin}\left(\log\left(1 + \frac{\Delta t_{t+1}}{\text{scale}}\right)\right), \qquad \tau_{t+1} \in \{1, \ldots, B\}. \tag{8}$$

Thus, the one-step predictive distribution is represented by two heads sharing the same predictive representation h_ϕ:

$$\begin{aligned} p_\phi(z_{t+1} \mid Q_{t,\mathcal{Z}}, \mathcal{C}_{\mathcal{Z}}) &= \text{softmax}(W_z h_\phi), \\ p_\phi(\tau_{t+1} \mid Q_{t,\mathcal{Z}}, \mathcal{C}_{\mathcal{Z}}) &= \text{softmax}(W_\tau h_\phi). \end{aligned} \tag{9}$$

The model can then be rolled out autoregressively to approximate the distribution of futures $P(F|.)$ Holtzman et al. (2020).

4.3.2 ARCHITECTURE

The shared representation h_ϕ alternates two attention operations at each layer. A causal Transformer block models temporal dependencies within each sequence, while a column-wise induced-attention block aggregates information across reference sequences. The former avoids quadratic complexity in sequence length through RoPE (Su et al., 2024) and a sliding attention window (Beltagy et al., 2020); the latter reduces the complexity of cross-context aggregation using a small set of learnable inducing points (Lee et al., 2019; QU et al., 2026). Further architectural details are provided in Appendix B.2.

4.3.3 DECODING LATENT CODES TO EVENT TYPES

To approximate the conditional future law $P_m(F \mid Q_t, \mathcal{C})$, DES-PFN first forecasts the immediate next step's latent code $z_{t+1} \in \mathcal{Z}$. However, the target inference task requires forecasting the actual next event x_{t+1} within its process-specific alphabet $\mathcal{X}_m$. Because the quantization in equation 2 is a many-to-one projection, we must decode DES-PFN's predictive outputs back into $\mathcal{X}_m$. We achieve this using a trained *prototype-pooling* attention head inspired by few-shot learning architectures Snell et al. (2017). Crucially, this head utilizes the context $\mathcal{C}$ as a reference set and operates directly on DES-PFN's contextual predictive representation h_ϕ. Every event occurrence in the context that shares the raw identifier $v \in \mathcal{X}_m$ is mean-pooled into a single prototype p_v (identifiers absent from $\mathcal{C}$ receive no prototype). The query's hidden state g then attends over the available prototypes $\{p_v\}_{v \in \mathcal{X}_m}$ to predict the first step of the future, which directly yields a conditional distribution over $\mathcal{X}_m$, as:

$$\hat{P}(X_{t+1} = v \mid h_\phi, \mathcal{C}) = \mathrm{softmax}_v\left(\frac{\langle W_q g,\, W_k p_v \rangle}{\sqrt{d}}\right). \tag{10}$$

The projection matrices W_q and W_k are trained jointly with DES-PFN via an auxiliary decoding cross-entropy loss. Consequently, rather than learning a fixed, global raw-identifier table, the decoding head is dynamically rebuilt from $\mathcal{C}$ for every new context.

5 EXPERIMENTS

In the following experiments, we evaluate the zero-shot capability of **DES-PFN** on next-event prediction for multiple horizons (N=1,5,20) and interpret the codebook usage under **DES-JEPA**.

Baselines, Datasets & Experimental Setup. We evaluate **DES-PFN** against FIM-PP (Berghaus et al., 2026) and three gradient-trained neural TPP baselines, RMTPP (Du et al., 2016), NHP (Mei & Eisner, 2017), and SAHP (Zhang et al., 2020), on next-event prediction tasks for different event horizons. We benchmark the model performance on six real event-type datasets: Twitter retweet cascades, StackOverflow badge awards, Taobao and Amazon user actions (Xue et al., 2024), HDFS system logs (Zhu et al., 2023), and Vehicle event logs (Math et al., 2025) reported at two vocabulary caps ($|\mathcal{X}|$=500 and $|\mathcal{X}|$=700). Every zero-/few-shot method, i.e., DES-PFN, FIM-PP and each neural baseline's *few-shot* variant (refitted from scratch on only the in-context set), sees the identical n_{context}=128 example sequences. DES-PFN and FIM-PP receive no gradient updates. Each *fully-trained* neural baseline variant is trained for 5 epochs. To ensure a practical and standardized evaluation framework, we impose a strict wall-clock time budget of 600 seconds per task. This constraint highlights the operational limits of traditional neural point processes, whose parameter counts and computation scale aggressively with vocabulary size. We report the validation top-1 accuracy, as a reference upper bound. In the experiments, DES-JEPA and DES-PFN are pretrained on the six described DGPs with a maximum codebook cardinality of K=768 and contain 60 million and 40 million parameters, respectively. For further details, please refer to Appendix A.

5.1 NEXT-EVENT PREDICTIONS

Results. Table 1 reports DES-PFN's zero-shot (no-gradient-update) next-event prediction accuracy across six real datasets spanning three to seven hundred event types. Fully-trained specialists on each dataset's entire training split still outperform DES-JEPA on four lower-cardinality datasets (`easytpp`). However, this specialism does not generalize well and is expensive, since each needs a real held-out test split and per-dataset retraining from scratch to be accurate. In low cardinality, FIM-PP (same setting as DES-PFN, zero-shot, in-context inference) outperforms DES-PFN in three

Table 1: **Real-world next-event predictions results** reported as the mean±std top-1 accuracy (↑) averaged over 5 seeds for n_{context}=128, n_{query}=20 and L=32. Baselines are reported in two regimes: *few-shot* and *full* (fully-trained). Since no real train/test split is available for HDFS , the 'full' column reports a *local train/test split* (§-marked), whereas 'few-shot' is not executed ("–"). FIM-PP only supports $|\mathcal{X}| \leq 22$ marks, thus, if $|\mathcal{X}| > 22$, it is not executed ("–"). Bold/underline mark the best/second-best score per row, † marks a method that exceeded a 600s time budget.

		RMTPP		NHP		SAHP		FIM-PP	**DES-PFN (ours)**
Dataset	$\lvert\mathcal{X}\rvert$	few-shot	full	few-shot	full	few-shot	full		
Retweet	3	0.464±0.021	0.533	0.235±0.011	**0.591**	0.531±0.020	0.471	0.462±0.022	0.560±0.031*
StackOverflow	22	0.009±0.003	0.411	0.007±0.002	0.420	0.413±0.035	0.420	0.441±0.032	**0.450±0.042***
Raobao	17	0.227±0.049	**0.559**	0.490±0.049	0.522	0.433±0.041	0.465	0.516±0.040*	0.411±0.072
Rmazon	16	0.151±0.028	0.299	0.089±0.006	**0.319**	0.193±0.031	0.254	0.207±0.041	0.230±0.033*
HDFS	14	–	0.074§	–	0.063§	–	0.103§	**0.266±0.018***	0.161±0.040
Vehicle	500	–	†	–	†	–	†	–	**0.163±0.014***
Vehicle	700	–	†	–	†	–	†	–	**0.102±0.008***

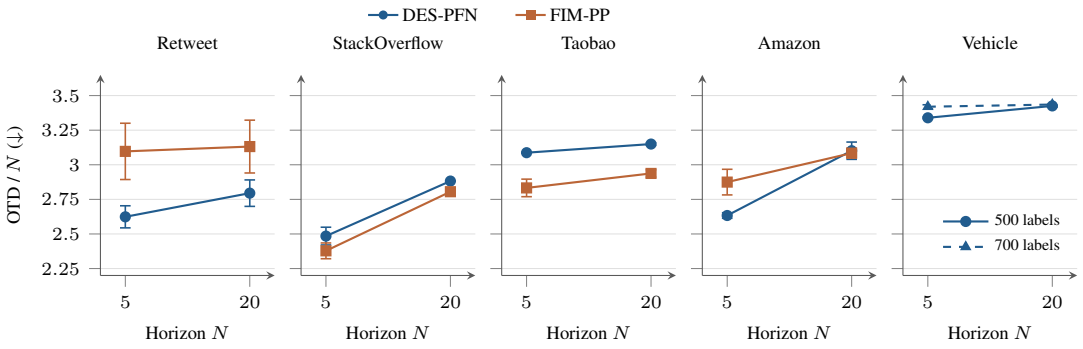

Figure 2: **Per-event-normalized long-horizon rollout error** vs. rollout horizon $N \in \{5, 20\}$ for FIM-PP and DES-PFN. Each panel presents a dataset. FIM-PP is not executable on Vehicle, thus we only display DES-PFN (at both vocabulary caps). See Appendix C.1.1 for extended results.

out of five datasets. However, its fixed 22-event type architecture precludes evaluation on the Vehicle dataset. DES-PFN supports significantly larger event vocabularies and is the only method able to report results for Vehicle; all gradient-trained baselines genuinely exceed the 600s training-time budget. This timeout is an expected architectural limitation rather than an artifact of the experimental setup: fitting tens of millions of parameters to accommodate 500 or 700 event types from scratch creates a severe computational bottleneck, demonstrating the scalability advantages of foundation inference models for high-cardinality event streams.

5.2 Long-Horizon Rollout Quality in Few Shots Settings

Settings. We assess long-horizon rollout quality via the optimal transport distance (OTD) Karpukhin & Savchenko (2026) generated and ground-truth continuations, evaluated at forecast horizons of $N \in \{5, 20\}$ events and averaged over three random seeds. DES-PFN decodes its own rollout via nucleus sampling (p=0.5). A full description of OTD is given in Appendix C.1.1.

Results. Fig. 2 presents the one-to-one comparison between the two foundation inference models (FIMs) evaluated: FIM-PP and DES-PFN. We normalize the OTD by the horizon length to provide a comparison of the reported results per step. A flat line means error accumulates linearly with rollout length (no extra degradation per step beyond N=5's rate), while an upward slope means later steps are individually harder than earlier ones. DES-PFN outperforms FIM-PP on Retweet (on both horizons) and Amazon at N=5, while performance is on-par for Amazon at N=20. FIM-PP remains stronger performance on StackOverflow and Taobao at both horizons. Generally, OTD increases significantly when moving from short N=5 to longer horizons N=20 (e.g., on Vehicle, 16.7→68.5 at $|\mathcal{X}|$=500). This is consistent with our expectation; larger event-type vocabulary makes longer autoregressive rollouts progressively harder to keep calibrated. Interestingly, on Vehicle, the gap

between the two vocabulary caps (500 vs. 700) stays small across both horizons, suggesting the OTD degradation is driven primarily by the rollout length rather than the vocabulary size alone.

5.3 Codebook Analysis

Rate-Distortion and Codebook Usage. We analyze how DES-JEPA allocates latent codes across different DGPs. Evaluating four distinct families: HAWKES (self-exciting), BURST (sticky repeat), TEMPLATE (deterministic graph walk), and NEURAL (untrained LSTM) at vocabularies $|\mathcal{X}_m| \in [3, 500]$. We observe a strong Pearson correlation (0.73) between distinct code usage and statistical complexity Shalizi & Crutchfield (2001). This indicates that DES-JEPA allocates latent codes proportionally to task complexity. **Rate-distortion (bottom-left).** The quantizer in equation 2 operates over K=768 fixed codebook vectors. We rank all 768 by real usage frequency on a pooled reference batch, then, for each $K' \leq K$, re-quantize the same continuous teacher embeddings against only the top-K' most-used vectors and measure the resulting mean squared quantization error. The distortion remains essentially flat from K'=768 down to $K' \approx 384-512$ (6.19 $\rightarrow$ 6.40), rising sharply only below $K' \approx 256$ (8.89 at 256, 17.86 at 128, 89.61 at 32). Intuitively, one might expect that capturing the dynamics of diverse, randomly initialized DGPs across six distinct families would require a massive vocabulary ($|\mathcal{X}| \gg 1024$).

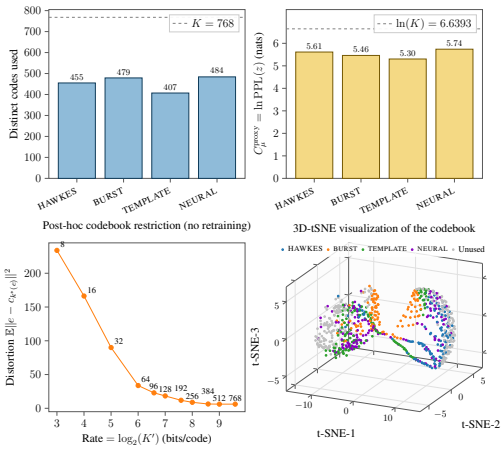

Figure 3: **Qualitative structure of the learned codebook**, real trained checkpoint, 4 maximally-different DGP families (HAWKES, BURST, TEMPLATE, NEURAL), $|X_m| \in [3, 500]$, 200 fresh tasks/family. **Top:** codes used and statistical complexity C_μ^{proxy} per family. **Bottom-left:** rate-distortion curve from post-hoc codebook restriction (no retraining) **Bottom-right:** 3D t-SNE of the 768 codebook vectors.

However, the minimal quality loss observed empirically validates a core premise of our framework: the many-to-one projection of quantization collapses highly diverse surface-level event sequences into a restricted, finite set of recurrent dynamical roles. Because disparate processes ultimately share this underlying predictive structure or MCPS, the *operational vocabulary size is naturally bottlenecked*. We therefore consider lowering the codebook capacity towards $K = 512$ in future work. **Qualitative structure (bottom-right):** We analyze a visual representation, a 3D t-SNE (van der Maaten & Hinton, 2008) plot, of the 768 codebook vectors. Color coding represents distinct process families and are thoroughly intermixed throughout the visualization; no family forms distinct clusters. The embedding also shows two qualitatively different local structures: elongated, curve-like filaments, and denser, rounded local clusters. We interpret the former as consistent with continuous variation in underlying event statistics. The denser local clusters, in contrast, may correspond to a limited set of recurrent dynamical roles, with events from different DGP families mapping to similar codes. We provide a range of perplexity values in Appendix C.2.

6 Conclusion and Limitations

We introduce a framework for unifying high-dimensional discrete event sequences into a shared vocabulary, enabling zero-shot downstream tasks such as next-event prediction. **DES-JEPA** pretrains a single, shared codebook over a meta-distribution of synthetic DGPs, accommodating arbitrary, anchor-free event sequences that share no vocabulary. Applying this frozen codebook allows **DES-PFN** to perform zero-shot next-event prediction. We evaluate the performance of DES-PFN on six real-world datasets spanning three orders of magnitude in cardinality, from K=3 to K=700 event types. Our approach is competitive at low cardinality and establishes state-of-the-art performance as it grows to hundreds of event types. Two **limitations** motivate future work: (i) the fixed codebook size restricts adaptability, motivating dynamically allocated vocabularies, and (ii) the explicit extraction of meta-causal predictive states (MCPS) via causal discovery remains unexplored.

AI USE STATEMENT

Generative AI tools were used to assist in the preparation of this work, including correcting, revising, and formatting the manuscript; supporting implementation used for the experimental evaluation; assisting in the creation of figures and visualizations; and initially identifying potentially relevant literature for consideration. AI was not used to generate synthetic datasets, to independently develop any scientific content of the paper, including the related work, methodology, experimental design, analysis, interpretations, or conclusions, nor was it used to generate, modify, or interpret experimental results. All AI-assisted code was reviewed and tested for correctness by at least one author, while experimental results and their interpretations were verified by at least two authors. Literature suggestions and references were manually checked against the corresponding sources. All AI-assisted text and figures were carefully reviewed, edited, and revised by the at least two authors before inclusion in the manuscript. The authors take full responsibility for the final content of this work, including all text, figures, code artifacts, scientific claims, analyses, and conclusions, regardless of whether generative AI tools were used during the preparation of this work.

REPRODUCIBILITY STATEMENT

The DES-JEPA/DES-PFN architecture, training objectives (Eqs. 2–9), and the synthetic pretraining meta-distribution together with its sampling parameters are fully specified in Section 4 and Appendices A and B, and the real-world datasets, splits, metrics and zero-shot evaluation protocol underlying Table 3 are properly described.

Released artifact. Anonymized source code, configuration files, and model code used to produce every number in this paper are appended to this submission as supplementary material. Model weights will be released publicly upon acceptance. The artifact contains the exact commands, seeds and hyperparameters needed to (i) regenerate the synthetic pretraining corpus, (ii) retrain both stages, and (iii) reproduce every table, figure and reported number, including the 4-family codebook analyses of Fig. 3. Because several datasets cannot be redistributed, the code does not bundle them: it documents how to obtain each from its original public source (easy-tpp, loghub HDFS and the Vehicle test split[1], and the EasyTPP-redistributed Retweet/StackOverflow/Taobao/Amazon logs) together with the exact preprocessing applied, and access-controlled releases require users to obtain their own credentialed access. The supplementary material is anonymized for double-blind review.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Türkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, et al. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2403.07815.

Michael Arbel, David Salinas, and Frank Hutter. Equitabpfn: A target-permutation equivariant prior fitted networks. 2026. URL https://arxiv.org/abs/2502.06684.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael G. Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, pp. 15619–15629, 2023. URL https://doi.org/10.1109/CVPR52729.2023.01499.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

[1] available here

Dmitrii Babaev, Nikita Ovsov, Ivan Kireev, Maria Ivanova, Gleb Gusev, Ivan Nazarov, and Alexander Tuzhilin. Coles: Contrastive learning for event sequences with self-supervision. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD/PODS '22, pp. 1190–1199. ACM, 2022. doi: 10.1145/3514221.3526129. URL http://dx.doi.org/10.1145/3514221.3526129.

Emmanuel Bacry, Martin Bompaire, Stéphane Gaïffas, and Jean-François Muzy. Sparse and low-rank multivariate hawkes processes. *Journal of Machine Learning Research*, 21(50):1–32, 2020. URL http://jmlr.org/papers/v21/15-114.html.

Wonho Bae, Mohamed Osama Ahmed, Frederick Tung, and Gabriel L. Oliveira. Meta temporal point processes. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2301.12023.

Junyeob Baek, Hosung Lee, Christopher Hoang, Mengye Ren, and Sungjin Ahn. Discrete jepa: Learning discrete token representations without reconstruction, 2025. URL https://arxiv.org/abs/2506.14373.

Albert-László Barabási. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039): 207–211, 2005.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. URL https://arxiv.org/abs/2004.05150.

David Berghaus, Patrick Seifner, Kostadin Cvejoski, Cesar Ojeda, and Ramses J Sanchez. In-context learning of temporal point processes with foundation inference models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=h9HwUAODFP.

Xiangyi Chen, Kousik Rajesh, Matthew Lawhon, Zelun Wang, Hanyu Li, Haomiao Li, Saurabh Vishwas Joshi, Pong Eksombatchai, Jaewon Yang, Yi-Ping Hsu, Jiajing Xu, and Charles Rosenberg. PinFM: Foundation model for user activity sequences at a billion-scale visual discovery platform. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*, pp. 381–390, 2025. doi: 10.1145/3705328.3748050. URL https://arxiv.org/abs/2507.12704.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, July 2006. ISBN 0471241954.

Felix Creutzig, Amir Globerson, and Naftali Tishby. Past-future information bottleneck in dynamical systems. *Physical Review E*, 79(4):041925, 2009. doi: 10.1103/PhysRevE.79.041925.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167, 2024. URL https://proceedings.mlr.press/v235/das24c.html.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 1555–1564, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939875. URL https://doi.org/10.1145/2939672.2939875.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021. URL https://arxiv.org/abs/2106.07447.

Qian Huang, Eric Zelikman, Sarah Li Chen, Yuhuai Wu, Gregory Valiant, and Percy Liang. Lexinvariant language models. *arXiv preprint arXiv:2305.16349*, 2023.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. Mimic-iv, a freely accessible electronic health record dataset. *Scientific Data*, 10 (1):1, 2023.

Yan Kagan and Leon Knopoff. Statistical study of the occurrence of shallow earthquakes. *Geophysical Journal International*, 55(1):67–86, 10 1978. ISSN 0956-540X. doi: 10.1111/j.1365-246X.1978.tb04748.x. URL https://doi.org/10.1111/j.1365-246X.1978.tb04748.x.

Ivan Karpukhin and Andrey V. Savchenko. Detecting the future: All-at-once event sequence forecasting with horizon matching. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence (AAAI-26)*, volume 40, pp. 22536–22544, 2026. doi: 10.1609/aaai.v40i27.39413. URL https://ojs.aaai.org/index.php/AAAI/article/view/39413.

Gavin Kerrigan, Kai Nelson, and Padhraic Smyth. Eventflow: Forecasting temporal point processes with flow matching. 2026. URL https://arxiv.org/abs/2410.07430.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive computation and machine learning. MIT Press, 2009. ISBN 9780262013192. URL https://books.google.co.in/books?id=7dzpHCHzNQ4C.

Alex Labach, Aslesha Pokhrel, Xiao Shi Huang, Saba Zuberi, Seung Eun Yi, Maksims Volkovs, Tomi Poutanen, and Rahul G. Krishnan. Duett: Dual event time transformer for electronic health records. In Kaivalya Deshpande, Madalina Fiterau, Shalmali Joshi, Zachary Lipton, Rajesh Ranganath, Iñigo Urteaga, and Serene Yeung (eds.), *Proceedings of the 8th Machine Learning for Healthcare Conference*, volume 219 of *Proceedings of Machine Learning Research*, pp. 403–422. PMLR, 8 2023. URL https://proceedings.mlr.press/v219/labach23a.html.

Patrick J. Laub, Young Lee, Philip K. Pollett, and Thomas Taimre. Hawkes models and their applications. *Annual Review of Statistics and Its Application*, 12 (Volume 12, 2025):233–258, 2025. ISSN 2326-831X. doi: https://doi.org/10.1146/annurev-statistics-112723-034304. URL https://www.annualreviews.org/content/journals/10.1146/annurev-statistics-112723-034304.

Yann LeCun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. 2022. URL https://api.semanticscholar.org/CorpusID:251881108.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3744–3753, 2019.

Zhuoqun Li and Mingxuan Sun. Sparse transformer hawkes process for long event sequences. In Danai Koutra, Claudia Plant, Manuel Gomez Rodriguez, Elena Baralis, and Francesco Bonchi (eds.), *Machine Learning and Knowledge Discovery in Databases: Research Track*, pp. 172–188, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-43424-2.

Hugo Math, Rainer Lienhart, and Robin Schön. Harnessing event sensory data for error pattern prediction in vehicles: A language model approach. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(18):19423–19431, 4 2025. doi: 10.1609/aaai.v39i18.34138.

Hongyuan Mei and Jason Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/6463c88460bd63bbe256e495c63aa40b-Abstract.html.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. *International Conference on Learning Representations*, 2022.

Jingang QU, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=SxsyLjIfWB.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *npj Digital Medicine*, 4:86, 2021. URL https://doi.org/10.1038/s41746-021-00455-y.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Cosma Rohilla Shalizi and James P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of Statistical Physics*, 104:817–879, 2001. doi: 10.1023/A:1010388907793.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes, 2020. URL https://arxiv.org/abs/1909.12127.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Ethan Steinberg, Jason Alan Fries, Yizhe Xu, and Nigam Shah. MOTOR: A time-to-event foundation model for structured medical records. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2301.03150.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C), 3 2024. ISSN 0925-2312. doi: 10.1016/j.neucom.2023.127063. URL https://doi.org/10.1016/j.neucom.2023.127063.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

Stijn Verdenius, Andrea Zerio, and Roy L. M. Wang. LaT-PFN: A joint embedding predictive architecture for in-context time-series forecasting. *arXiv preprint arXiv:2405.10093*, 2024. URL https://arxiv.org/abs/2405.10093.

Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I. Jordan. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles*, pp. 117–132, 2009.

Siqiao Xue, Xiaoming Shi, James Zhang, and Hongyuan Mei. HYPRO: A hybridly normalized probabilistic model for long-horizon prediction of event sequences. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Siqiao Xue, Xiaoming Shi, Zhixuan Chu, Yan Wang, Hongyan Hao, Fan Zhou, Caigao Jiang, Chen Pan, James Y. Zhang, Qingsong Wen, Jun Zhou, and Hongyuan Mei. Easytpp: Towards open benchmarking temporal point processes. In *International Conference on Learning Representations*, 2024.

Mai Zeng, Florence Regol, and Mark Coates. Interacting diffusion processes for event sequence forecasting, 2024. URL https://arxiv.org/abs/2310.17800.

Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive Hawkes process. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11183–11193. PMLR, 7 2020. URL https://proceedings.mlr.press/v119/zhang20q.html.

Yunhao Zhang, Ruiying Qi, Jiale Zheng, Jianfeng Zhang, Lujia Pan, and Junchi Yan. Time series as language: A universal tokenizer for general-purpose time series foundation models. *arXiv preprint arXiv:2606.09861*, 2026. URL https://arxiv.org/abs/2606.09861.

Jieming Zhu, Shilin He, Pinjia He, Jinyang Liu, and Michael R. Lyu. Loghub: A large collection of system log datasets for ai-driven log analytics. 2023. URL https://arxiv.org/abs/2008.06448.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer Hawkes process. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11692–11702. PMLR, 7 2020. URL https://proceedings.mlr.press/v119/zuo20a.html.

APPENDIX

A PRETRAINING AND THE SYNTHETIC META-DISTRIBUTION

Both models, DES-JEPA and DES-PFN, are pretrained exclusively on synthetic tasks. Thus, no event identifier, sequence, or timestamp from a downstream dataset is observed while fitting either DES-JEPA or DES-PFN. This is important in our setting: the integers naming event types are sampled afresh for every task and have no cross-task semantics. The prior is consequently a distribution over event *dynamics*, not a global vocabulary. We describe below the exact meta-distribution used by DES-JEPA and the DES-PFN.

A.1 TASK CONSTRUCTION AND LABEL RANDOMIZATION

Let g denote a DGP family, let θ denote its task-specific parameters, and let K_{act} be the number of active raw event types. A synthetic task is drawn as

$$g \sim p(g), \qquad K_{\mathrm{act}} \sim \mathrm{Unif}\{3, \dots, K_{\max}\}, \qquad \theta \sim p_g(\theta), \qquad \{(x_{1:L}^{(s)}, \Delta t_{1:L}^{(s)})\}_{s=1}^{C+1} \sim P_{g,\theta}. \tag{11}$$

Here C streams form the context and the remaining stream is the query. Crucially, after sampling a task we draw a fresh injective map from its local alphabet $\{1, \dots, K_{\mathrm{act}}\}$ into the available raw-id slots and apply it to all $C+1$ streams. The same permutation is used within a task, so its transition structure is intact. We prepend a BOS marker Devlin et al., append EOS, and form absolute timestamps by cumulative summation of the generated gaps.

Fig. 4 summarizes this sampling procedure. Family identity is sampled once per pretraining of a DES-PFN batch for efficient vectorized generation, while θ, K_{act}, and the raw-id permutation are redrawn independently for every task in that batch. Therefore a model cannot memorize a finite catalogue of generators.

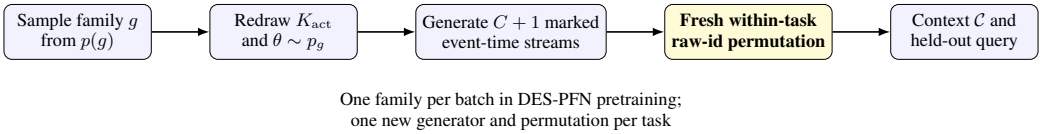

Figure 4: **Synthetic prior-fitting task.** Every task is a newly sampled stochastic process with an anonymous local alphabet. *Context and query streams share a process and permutation*, whereas different tasks do not.

A.2 MIXTURE OF STOCHASTIC PROCESS FAMILIES

The meta-distribution contains the following six families:

- **Markov chains** (Koller & Friedman, 2009): an order $r \in \{1, \dots, 5\}$ is redrawn each task; the next type is a softmax over a sparse, recency-decayed linear interaction with the last r types, sharpened by a random nonlinear MLP term.
- **Hawkes processes** (Kagan & Knopoff, 1978): a multivariate self-exciting point process simulated by exact Ogata thinning, with a sparse excitation matrix with either a short-range exponential or a long-range, non-Markovian power-law (Omori–Utsu) kernel.
- **Untrained neural sequence models** (Hochreiter & Schmidhuber, 1997): a randomly initialized LSTM, never trained, whose hidden state jointly drives both the next event type and its inter-arrival time. This forms a generic, fully-coupled nonlinear dynamic process.
- **Hidden Markov models** (Koller & Friedman, 2009): a sticky latent-state chain (3–7 states) whose state simultaneously governs a Dirichlet emission over event types and a Weibull holding time, the same hidden cause drives both what happens and when.
- **Bursty renewal processes** (Barabási, 2005): i.i.d. heavy-tailed inter-arrival times paired with a sticky-categorical type chain, modeling verbatim repeat-heavy streams such as keep-alives or retries in event logs.

- **Template chains**: a near-deterministic finite-state chain over event types. A random directed graph with at most one outgoing edge per node, walked deterministically and emitted through a small uniform-noise floor. Giving event-type *pairs* a near-deterministic causal coupling that self-exciting rate models cannot express.

Template chains receive probability 0.30 and Hawkes processes probability 0.15; Markov, neural, HMM, and burst families each receive 0.1375. The elevated template mass ensures frequent exposure to high-strength, type-specific causal relations, whereas the Hawkes mass supplies event-time coupling without making its more expensive exact simulation dominate training.

Table 2: Task-level randomization in the synthetic prior of DES-PFN pretraining. Intervals are sampled independently unless stated otherwise. All tasks also sample an activation from {ReLU, GELU, tanh, sigmoid, SiLU} and additive generator noise in $[0, 0.1]$ where applicable.

Family	Randomized structure and parameters
Markov	Order $r \in \{1, \ldots, 5\}$; sparse recency-decayed transition mechanism with density in $[0.3, 0.8]$; a nonlinear interaction term with probability 0.5; Weibull holding times.
Hawkes	Sparse excitation density in $[0.2, 0.5]$, background rates in $[0.05, 0.3]$, and spectral radius below 0.9. With probability 0.5, use an exponential kernel with decay in $[1, 5]$; otherwise use a normalized power-law kernel with $\alpha \in [0.3, 1.2]$ and $c \in [0.05, 0.5]$. Both are simulated by exact Ogata thinning Kagan & Knopoff (1978).
Neural	An untrained LSTM Hochreiter & Schmidhuber (1997) with hidden width 8–32, one or two layers, event embedding width 8–16, and initialization standard deviation in $[0.5, 2.0]$; its hidden state drives the next mark and Weibull gap.
HMM	3–7 latent states, Dirichlet emission concentration in $[0.2, 0.5]$, and stickiness in $[0.7, 0.99]$ (sampled log-uniformly in escape probability). The latent state jointly controls mark emissions and holding times.
Burst	A renewal process with Weibull shape $[0.15, 0.7]$ and scale $[0.2, 1.0]$, paired with a categorical mark chain with stickiness $[0, 0.9]$ and Dirichlet concentration $[0.3, 1.5]$.
Template	A library of at most 10 short cascades with length 3–12, overlap $[1.2, 2.5]$, and near-deterministic follow-through (deviation noise in $[0, 0.1]$). Fast and slow gap scales are $[0.05, 0.3]$ and $[0.5, 3]$. With probability 0.5, a template branches to 2–3 successors.

These families deliberately cover complementary mechanisms: finite-order dependence (Markov), self-excitation and long memory (Hawkes), generic nonlinear recurrence (neural), latent regime switching (HMM), heavy-tailed bursts (renewal), and almost deterministic multi-step causal cascades (template). The mixture is not intended as a literal model of any one domain. Rather, it supplies varied but controlled predictive structures from which a process-invariant representation can be learned.

A.3 DES-JEPA PRETRAINING

DES-JEPA pretraining uses the same six-family mixture with fixed total length 128, 25 context streams per task, and $K_{\text{act}} \sim \text{Unif}\{3, \ldots, 1024\}$. It was optimized for $40{,}000$ steps with batch size 10, AdamW learning rate 4×10^{-4}, weight decay 10^{-2}, 500 linear warm-up steps, and gradient clipping at 1.0. The student has six layers of width 768 with 12 heads and a causal window of 32; the three-layer bidirectional teacher is updated by EMA with coefficient 0.999. The codebook has 768 vectors of dimension 768 and predicts a one-event target window.

A.4 DES-PFN PRETRAINING

The DES-PFN pretraining uses a log-uniformly sampled context size $C \in [32, 2048]$ and total sequence length $L \in [16, 256]$. The sampler draws a token budget B in log space around the geometric centre of the admissible range, with a 1.5-octave width, samples C log-uniformly, and sets $L \approx B/C$ with multiplicative jitter of at most $2^{\pm 0.5}$. Length is rounded to a multiple of four and the hard safety constraint $C, L \leq 60{,}000$ is then enforced. This produces both many-short-context

and few-long-context tasks without disproportionately favouring the expensive upper-right corner of the (C, L) grid. Active cardinality is sampled uniformly from $[3, 1024]$, except for Hawkes tasks, which are capped at 256 active types because exact thinning scales poorly with cardinality while its temporal mechanism remains represented.

For the HMM, burst, and template families only, each generated inter-event gap is independently replaced by a zero gap with probability drawn from $[0, 0.10]$, $[0, 0.05]$, and $[0, 0.05]$, respectively (never at the first event). This deliberately includes tied timestamps, mimicking coarse or batch-recorded event logs. Other families have no such imputation.

We use 16 tasks per optimization step for $100{,}000$ steps, 8 data workers, bfloat16 mixed precision, AdamW with learning rate 3×10^{-4} and weight decay 10^{-3}, $1{,}000$ warm-up steps, cosine decay, and gradient clipping at 1.0. The model has ten transformer layers of width 512, eight heads, 64 inducing vectors, temporal window 64, and the frozen 768-entry DES-JEPA codebook. It jointly optimizes next code, discretized next-time, context auxiliary, and prototype-decoding losses, with time-loss weight 0.3, context-loss weight 0.3, and decode loss weight 0.3; label smoothing is 0.1 and the logit regularizer weight is 10^{-3}. The prototype decoder has dimension 192 and predicts among $1{,}032$ raw-id slots using only the task context.

B METHODOLOGY

B.1 DES-JEPA

B.1.1 EMBEDDINGS

Every event is embedded from purely statistics of its own occurrence history: self-information/surprisal, running (global and local-window) frequency, whether the type has been seen before, and recency and dispersion of its own inter-arrival gaps, together with event-count ("step") analogues of the timing features that need no real timestamp. None of these is computed from a learned lookup table keyed on the raw event id, since a raw id carries no stable meaning across draws of $m \sim \mathcal{M}$ (Section 4): therefore event identity is represented purely through these statistics. We attach no additional per-event covariate beyond type and time (no "marks" in the temporal-point-process sense) to any data-generating process.

Fig. 5 shows the pairwise correlation of these features on a representative synthetic batch: highly correlated pairs (e.g. `causal_freq`/`local_freq`, `mean_dt_type`/`mean_step_gap_type`) are deliberately kept rather than pruned, since their real-valued/step-valued counterparts remain the only signal available whenever timestamps are missing or unreliable.

B.2 DES-PFN

B.2.1 EMBEDDINGS

DES-PFN's event-code embedding table is the frozen DES-JEPA codebook itself (Section 4.3), followed by a learned linear projection whenever the two models' hidden sizes differ. Inter-arrival time uses a Chronos-style (Ansari et al., 2024) discretized embedding: $\log(1 + \Delta t/\text{scale})$ is binned into a fixed vocabulary and looked up, turning next-time prediction into classification rather than continuous regression. An optional duplicate-count embedding additionally marks immediate event repeats.

B.2.2 TRANSFORMER BLOCK

DES-PFN's trunk stacks $L{=}5$ layers ($d{=}384$, 12 heads, checkpoint used throughout this paper), each alternating one causal block and one cross-sequence block over the (sequence, time, hidden) grid formed by the reference context and the query.

Causal Attention. Every sequence (each context row and the query) is attended to independently along its own temporal axis: multi-head self-attention with RoPE (Su et al., 2024) position encoding and a sliding-window of 32, so a token at position t attends only to positions $[t{-}32, t]$; a causal mask additionally prevents any position from attending to its own future. Each causal block is followed by a

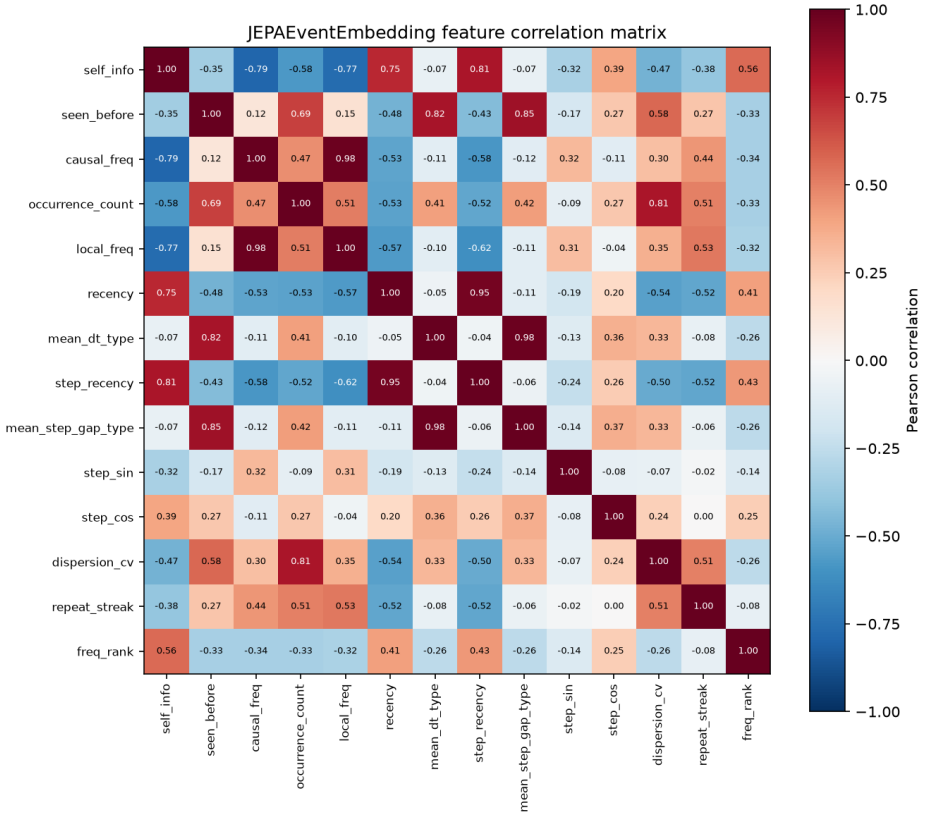

Figure 5: Pairwise Pearson correlation across the 14 causal statistical features feeding the DES-JEPA event embedding, computed on a mixed-DGP synthetic batch.

two-layer GELU feed-forward network of hidden width $4d$, with pre-layer-norm residual connections around both the attention and feed-forward sub-layers.

Col-Wise Attention. Cross-sequence information is exchanged along the sequence axis, independently at every time step, via a two-stage induced-attention design following tabular foundation models such as TabICLv2 (QU et al., 2026) and the inducing-point mechanism of Set Transformer (Lee et al., 2019): a small, fixed set of n_{inducing}=16 learnable vectors first attend to the context rows' hidden states at that time step, pooling them into one compact summary, which is then attended to as keys/values by the query row, replacing what would otherwise be full $O(n^2)$ attention across up to hundreds of reference sequences. In the configuration used throughout this paper, context rows receive *no* cross-sequence attention at all: they are updated purely by the causal block above, with no visibility into each other or into the query. Only the query row cross-attends, reading the single pooled summary built from the context rows' own temporal representations. This asymmetric split guarantees the context can never leak information from the query, while the query is conditioned on the whole context set at every layer.

B.2.3 Training Objective

During training, we sample a process $m \sim \mathcal{M}$ alongside a reference context $\mathcal{C}$ and a query sequence Q generated by that same process. After tokenizing the observations using the frozen DES-JEPA codebook, the model is optimized to approximate the true future law $P_m(F \mid Q_t, \mathcal{C})$ step-by-step. Specifically, at each valid sequence position t, the model must predict the immediate next components of the future sequence F: the latent code z_{t+1}, the time bin τ_{t+1}, and the raw decoded event type X_{t+1}.

For a given sequence position t, the step-wise loss relies on the observed prefix $Q_{t,\mathcal{Z}}$ and is defined as:

$$\ell_t = \mathrm{CE}\left(z_{t+1},\, p_\phi(z_{t+1} \mid Q_{t,\mathcal{Z}}, \mathcal{C}_\mathcal{Z})\right) + \alpha\, \mathrm{CE}\left(\tau_{t+1},\, p_\phi(\tau_{t+1} \mid Q_{t,\mathcal{Z}}, \mathcal{C}_\mathcal{Z})\right),$$

where α balances the event-code and temporal prediction terms.

The overall training objective amortizes this autoregressive future-prediction loss over the meta-distribution $\mathcal{M}$:

$$\mathcal{L}_{\mathrm{PFN}} = \mathbb{E}_{m\sim\mathcal{M}}\mathbb{E}_{(\mathcal{C},Q)\sim m}\left[\lambda_{\mathrm{ctx}}\, \mathbb{E}_{t\in\mathcal{C}_\mathcal{Z}}[\ell_t] + \lambda_{\mathrm{qry}}\, \mathbb{E}_{t\in Q_\mathcal{Z}}[\ell_t] + \lambda_{\mathrm{dec}}\mathcal{L}_{\mathrm{dec}}\right].$$

The query term receives a higher weight ($\lambda_{\mathrm{qry}} > \lambda_{\mathrm{ctx}}$) because autoregressively predicting the unseen future F of the query is the primary inference task, whereas the context and decoding terms serve as auxiliary supervision to strictly align the shared predictive representation.

C Evaluation

C.1 Next-Event Prediction

Table 3: **Real-world next-event-type accuracy** (top-1, mean±sample standard deviation over five seeds), n_{context}=128, n_{query}=20, and L=32. RMTPP/NHP/SAHP are reported *few-shot* (refit from scratch on the same 128 context sequences DES-PFN/FIM-PP see) and *full* (trained on the entire dataset split; mean only). The supplied evaluation artifacts contain specialist results for the four EasyTPP datasets only; unavailable cells are "–". FIM-PP is capped at $|\mathcal{X}| \leq 22$ marks. Bold/underline mark the best/second-best observed score in a row; ⋆ marks the best zero-/few-shot score. Vehicle is reported at two vocabulary caps.

		RMTPP (Du et al., 2016)		NHP (Mei & Eisner, 2017)		SAHP (Zhang et al., 2020)			
Dataset	$\lvert\mathcal{X}\rvert$	few-shot	full	few-shot	full	few-shot	full	FIM-PP (Berghaus et al., 2026)	DES-PFN (ours)
retweet (Xue et al., 2024)	3	0.387±0.050	0.520	0.204±0.051	**0.571**	0.532±0.008	0.393	0.387±0.049	0.563±0.037⋆
stackoverflow (Xue et al., 2024)	22	0.009±0.005	0.433	0.005±0.004	0.449	0.417±0.051	0.449	**0.454±0.039**⋆	0.447±0.042
taobao (Xue et al., 2024)	17	0.294±0.092	**0.578**	0.497±0.066	0.519	0.429±0.071	0.432	0.522±0.041⋆	0.412±0.074
amazon (Xue et al., 2024)	16	0.132±0.020	0.234	0.064±0.014	**0.303**	0.213±0.037	0.283	0.187±0.028	0.239±0.039⋆
HDFS (Zhu et al., 2023; Xu et al., 2009)	14	–	–	–	–	–	–	–	**0.222±0.028**⋆
Vehicle (Math et al., 2025)	500	–	–	–	–	–	–	–	**0.163±0.016**⋆
Vehicle (Math et al., 2025)	700	–	–	–	–	–	–	–	**0.103±0.009**⋆

Table 4: **Longer-query next-event-type accuracy** (top-1, mean±sample standard deviation over five seeds), using the same n_{context}=128 and L=32 protocol as Table 3, but n_{query}=60. This table is computed from the corresponding query-60 evaluation artifacts. Formatting follows Table 3.

		RMTPP (Du et al., 2016)		NHP (Mei & Eisner, 2017)		SAHP (Zhang et al., 2020)			
Dataset	$\lvert\mathcal{X}\rvert$	few-shot	full	few-shot	full	few-shot	full	FIM-PP (Berghaus et al., 2026)	DES-PFN (ours)
retweet (Xue et al., 2024)	3	0.416±0.027	0.508	0.210±0.057	**0.556**	0.527±0.015	0.418	0.419±0.027	0.540±0.020⋆
stackoverflow (Xue et al., 2024)	22	0.007±0.002	0.434	0.006±0.003	0.453	0.423±0.020	0.454	**0.462±0.017**⋆	0.455±0.015
taobao (Xue et al., 2024)	17	0.300±0.023	**0.577**	0.493±0.021	0.523	0.445±0.020	0.447	0.519±0.036⋆	0.426±0.028
amazon (Xue et al., 2024)	16	0.131±0.017	0.258	0.077±0.010	**0.318**	0.210±0.010	0.278	0.221±0.022	0.257±0.017⋆
HDFS (Zhu et al., 2023; Xu et al., 2009)	14	–	–	–	–	–	–	–	**0.228±0.000**⋆
Vehicle (Math et al., 2025)	500	–	–	–	–	–	–	–	**0.166±0.012**⋆
Vehicle (Math et al., 2025)	700	–	–	–	–	–	–	–	**0.111±0.009**⋆

C.1.1 Long Horizon Forecasting

Following HYPRO (Xue et al., 2022) and FIM-PP (Berghaus et al., 2026), we score a method's autoregressive rollout of N future events by the *Optimal Transport Distance* (OTD) between the generated event sequences $\hat{X} = ((\hat{t}_i, \hat{x}_i))_{i=1}^N$ and the ground-truth continuation $X^\star = ((t_j^\star, x_j^\star))_{j=1}^N$. Both are *raw event sequences* over the process's own alphabet $\mathcal{X}_m$. OTD is the minimum total cost of editing $\hat{X}$ into $X^\star$ under the three operations of *moving* an aligned event's time, *deleting* a spurious predicted event, and *inserting* a missed true event for every aligned pair. Writing $S(\hat{X}, X^\star) = \{(i,j) : \hat{x}_i = x_j^\star\}$ for the admissible same-type pairs and μ for a coupling of the uniform marginals

over the N predicted and N true events, OTD is the linear program

$$\mathrm{OTD}_C(\hat{X}, X^\star) = \min_{\mu} \underbrace{\sum_{(i,j)\in S(\hat{X},X^\star)} \mu_{ij}\,|\hat{t}_i - t^\star_j|}_{\text{move}} + \underbrace{C\sum_i \Big[\tfrac{1}{N} - \sum_j \mu_{ij}\Big]}_{\text{delete}} + \underbrace{C\sum_j \Big[\tfrac{1}{N} - \sum_i \mu_{ij}\Big]}_{\text{insert}}, \tag{12}$$

where the insert and delete costs are symmetrized to a single C, as in (Xue et al., 2022). We report the quantity HYPRO reports: the average of OTD_C over their fixed grid $\mathcal{C} = \{0.05, 0.5, 1, 1.5, 2, 3, 4\}$, so that $\mathrm{OTD} = |\mathcal{C}|^{-1}\sum_{C\in\mathcal{C}} \mathrm{OTD}_C$. Finally, because the move cost is an absolute time distance, we express all predicted and true times in units of a single per-context time scale, so that one grid of C values is meaningful across datasets of very different temporal resolutions. Lower is better throughout.

Table 5: **Long-horizon rollout few-shot quality** (Optimal Transport Distance, lower is better, mean±std over 3 runs), n_{context}=128, n_{query}=60, L=32, DES-PFN decoded via nucleus sampling (p=0.5). RMTPP/NHP/SAHP are the *few-shot* variants (refit from scratch on the same 128 context sequences DES-PFN/FIM-PP see). Bold/underline mark the best/second-best (lowest OTD) score per row.

Dataset	N	DES-PFN (ours)	FIM-PP	RMTPP-few	NHP-few	SAHP-few
Retweet (Xue et al., 2024)	5	**13.120±0.399**	15.484±1.017	14.862±0.294	14.943±0.126	14.024±0.419
Retweet (Xue et al., 2024)	20	**55.899±1.912**	62.632±3.817	60.960±0.895	60.858±0.937	60.036±0.838
StackOverflow (Xue et al., 2024)	5	12.423±0.321	**11.890±0.285**	17.148±0.048	17.172±0.020	13.627±0.305
StackOverflow (Xue et al., 2024)	20	57.647±0.009	**56.104±0.599**	68.739±0.107	68.760±0.031	63.749±0.202
Taobao (Xue et al., 2024)	5	15.437±0.011	**14.164±0.317**	16.086±0.323	15.814±0.204	15.699±0.116
Taobao (Xue et al., 2024)	20	62.994±0.194	**58.749±0.528**	65.387±0.516	63.335±0.320	62.913±0.683
Amazon (Xue et al., 2024)	5	**13.167±0.101**	14.374±0.463	15.146±0.585	15.571±0.322	15.117±0.430
Amazon (Xue et al., 2024)	20	**62.034±1.250**	**61.685±0.706**	66.130±0.418	64.190±0.331	63.753±0.247
Vehicle ($\lvert\mathcal{X}\rvert$=500) (Math et al., 2025)	5	**16.698±0.030**	†	†	†	†
Vehicle ($\lvert\mathcal{X}\rvert$=500) (Math et al., 2025)	20	**68.507±0.047**	†	†	†	†
Vehicle ($\lvert\mathcal{X}\rvert$=700) (Math et al., 2025)	5	**17.098±0.071**	†	†	†	†
Vehicle ($\lvert\mathcal{X}\rvert$=700) (Math et al., 2025)	20	**68.706±0.060**	†	†	†	†

C.2 What the Codebook Looks Like: 3D t-SNE Across Embedding Perplexities

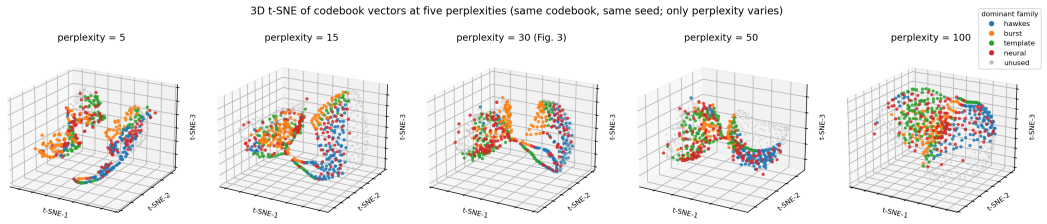

Figure 6: The same learned codebook (K=768) embedded by 3D t-SNE at five perplexities, varying only that parameter ($u \in \{5, 15, 30, 50, 100\}$; fixed seed, identical per-code family labels). Colors give each code's dominant DGP family by usage share; gray codes are never selected. t-SNE perplexity u is the effective number of neighbours used to measure local structure.

t-SNE's perplexity is a free parameter of the *embedding*: it is the effective number of neighbours over which local structure is measured. Low perplexity u resolves fine local detail and fragments the cloud; high u averages over more neighbours and smooths everything toward a single blob. Because it is a free parameter, a single t-SNE panel can make the same codebook look more or less separated than it truly is, and a qualitative claim about the codebook's geometry should not rest on one value of it.

Fig. 6 therefore re-embeds the *same* K=768 learned codebook vectors at $u \in \{5, 15, 30, 50, 100\}$, changing nothing else (identical vectors, identical seed, identical initialization, identical per-code labels) Codes are colored by dominant DGP family, with never selected codes in gray. At u=5 the codebook shatters into many small fragments with the four families strongly interleaved; at u=15–30

the geometry resolves into its clearest structure; by $u{=}100$ the cloud has collapsed toward a single dense ball in which the families are nearly fully intermixed. The qualitative reading that codes reflect *dynamical role* rather than family identity, the claim motivating of the MCPS (Def. 3) holds at every setting, and the $u{=}30$ panel reproduces the single-perplexity view exactly. What does *not* survive the sweep is the apparent *degree* of separation: the elongated, visibly banded structure at $u{=}30$ is substantially a property of that choice of u, and a reader selecting $u{=}100$ would see far less. We therefore treat Fig. 6 as evidence that the codebook's usage is not organized by DGP identity, and explicitly not as a quantitative claim about how many well-separated predictive states it contains. A question we argue is better posed by the rate-distortion and statistical-complexity measurements of the experiments rather than by any single visualization.