EVENTFM: ZERO-SHOT FORECASTING OF WHAT HAPPENS WHEN

Anonymous authors
Paper under double-blind review

ABSTRACT

Foundation models promise train-once zero-shot forecasting, yet *event forecasting* remains far less developed than regular time-series forecasting. The obstacle is structural: event streams unfold as irregular occurrences in continuous time, lacking a shared grid, fixed output space, and large interoperable corpora that make regular time-series foundation models possible. Moreover, an event forecaster must predict both *what* happens next and *when* it happens, by inferring sequence-local relations among event types and arrival times. These challenges motivate EventFM, a foundation model for sequence-level zero-shot forecasting of marked temporal point processes. EventFM's event-native architecture normalizes time within each history, constructs one candidate query for each observed mark, and jointly predicts the next mark and its conditional occurrence time. To resolve the pretraining problem, we introduce the *Event Structural Simulator* (EventSS), an on-the-fly synthetic prior over event relations. EventSS captures trigger-response relations, overlapping workflows, delayed and cancellable consequences, recurrence, and non-additive event interactions. We pretrain EventFM on an EventSS prior and evaluate it in a strict sequence-level zero-shot setting, where each prediction uses only the observed history of the target sequence. Across five real-world benchmarks and structured synthetic scenarios, a single frozen EventFM shows strong zero-shot transfer, while targeted ablations show that both the event-native architecture and EventSS prior are important for transfer.

1 INTRODUCTION

Event sequences are the natural representation of many systems that evolve through discrete, typed occurrences in continuous time. Each event is a pair consisting of a real-valued timestamp, *when* something happens, and a categorical mark, *what* kind of event it is. Neural spikes (Truccolo et al., 2005), clinical events (Lee & Hauskrecht, 2021; Wei et al., 2023), financial events (Aït-Sahalia et al., 2015), customer purchases (Lysenko et al., 2019), and supply-chain workflows (Liu et al., 2007) all occur irregularly and asynchronously, while also carrying categorical identities such as spike source, intervention type, transaction type, purchase category, or workflow stage. Marked temporal point processes (MTPPs) have become a canonical framework for modeling such data (Shchur et al., 2021). In parallel, time series foundation models (TSFMs) have transformed regular forecasting via a zero-shot single forward pass on an unseen context, without target-side data collection, training, or tuning (Ansari et al., 2025; Jain et al., 2026). The success of TSFMs rests on shared structure across tasks: ordered grids, numerical targets, and reusable patterns such as trend, seasonality, and autocorrelation. Their pretraining benefits from broad public corpora and synthetic data, including scalable, privacy-free fully synthetic priors in ForecastPFN, CauKer, TempoPFN, and SarSim0 (Dooley et al., 2023; Xie et al., 2026; Moroshan et al., 2025; Oreshkin et al., 2026). Event forecasting is a natural candidate for the same paradigm, but lacks these conveniences.

Indeed, event forecasting violates each of these regularities. Events occur at irregular continuous times, and relevant time scales vary by orders of magnitude across processes. The prediction space is also process-specific: event-type vocabularies differ across sequences, labels need not align semantically, and the model must predict both the next time and mark. Finally, event dynamics are often relational rather than purely temporal, depending on which event types occurred, in what order, and through mechanisms such as excitation, inhibition, delayed effects, or multi-event interactions. This creates a three-fold gap. First, conventional TSFM architectures do not directly support this input-

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.

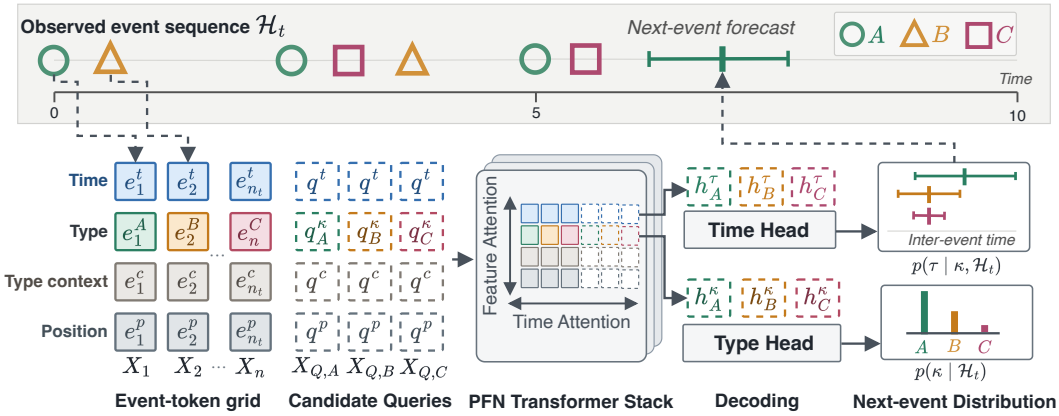


Figure 1: **EventFM zero-shot event forecasting.** From an observed marked event history, EventFM encodes each event's time, anonymous type, type context, and position. It then constructs one query per observed type, and uses a PFN transformer to predict the next-type distribution and type-conditional inter-event-time distributions, shown as capped bars for P10-90 quantiles.

output structure. Second, event forecasting lacks a standardized cross-domain pretraining corpus: real event logs are fragmented, proprietary, or privacy-sensitive. Third, synthetic priors for regularly sampled data do not model the event-level dependencies needed to generate event processes.

In this paper, we bridge this gap with a foundation model that infers a previously unseen event process from a single observed sequence, without assuming a shared time scale, event vocabulary, or target-side training data. We make two complementary contributions. (1) We introduce **EventFM**, an event-native PFN architecture for transfer across heterogeneous event sequences. EventFM treats time and event type as sequence-local coordinates, normalizing timestamps and constructing one candidate query for each observed event type so it can infer event dynamics in context and predict both *what* happens next and *when* (Figure 1). Thus, one backbone transfers across time scales, history lengths, and event-type dictionaries, with predictions driven by the sequence context rather than fixed labels or target-specific parameters. (2) We introduce **Event Structural Simulator (EventSS)**, a fast on-the-fly synthetic prior for learning this inference procedure without a large cross-domain event corpus. For each training sequence, EventSS samples a complete event process: event-type graph, source processes, transition and scheduling mechanisms, as well as delay laws. It simulates the process through a queue of pending future opportunities that events create, modify, or cancel. This queue-based instance-level semantics lets EventSS express trigger-response relations, overlapping workflows, gating, recurrence, and non-additive interactions, complementing Hawkes-style generators that aggregate past-event influence only through aggregate type intensities. Across five real benchmarks and three synthetic groups, EventFM ranks in the top two for both mark accuracy and time CRPS throughout, leading CRPS in seven of eight settings.

2 Background and Related Work

Neural Temporal Point Processes. Classical MTPPs specify mark-specific conditional intensities, ranging from classical parametric choices like Poisson (Kingman, 1993) to Hawkes (Hawkes, 1971) and ETAS excitation kernels (Ogata, 1988). Some parameterize the conditional intensity with recurrent or attention-based architectures (Du et al., 2016; Chang et al., 2025); others model the next-event time distribution directly (Omi et al., 2019; Kerrigan et al., 2026); and some extend point-process models to spatial or relational event settings (Chen et al., 2021; Trivedi et al., 2019). Most neural TPPs remain target-fitted: new dataset typically requires its own training and tuning. MetaTPP amortizes adaptation across target-domain sequences (Bae et al., 2023), while HyperHawkes performs zero-shot prediction using textual side information (Dubey et al., 2022). EventFM targets a stricter foundation-model setting: pretrain once on a broad synthetic prior, then apply the frozen model to target sequences with no target-dataset training or side information.

In-Context Event Models. Several event foundation models are pretrained within a single domain, such as electronic health records (Pang et al., 2021; Renc et al., 2025) or financial transaction histories (Hu et al., 2023; Ostroukhov et al., 2026). These models transfer within a fixed domain and event vocabulary, whereas another line of work infers new event processes from test-time context. FIM-PP (Berghaus et al., 2026) estimates a target conditional intensity from multiple context sequences, while EVIL searches for a short parameter-free Python program that computes predictive statistics from the same kind of context (Berghaus, 2026). FIM-PP and EVIL infer from multiple target-process sequences supplied at test time. We target the single-sequence regime: the frozen model forecasts from only the observed prefix of the sequence to be continued.

Synthetic Pretraining Data. TSFMs increasingly rely on synthetic pretraining, with priors based on Gaussian processes, causal generators, and classical stochastic processes (Ansari et al., 2024; Dooley et al., 2023; Xie et al., 2026; Oreshkin et al., 2026). These priors generate fixed-grid numerical sequences rather than marked events. The closest event analogue is Hawkes-style simulation, which aggregates pairwise mark interactions into intensities, samples by thinning (Ogata, 1981), and underlies FIM-PP pretraining (Berghaus et al., 2026) and the generated datasets used by EVIL (Berghaus, 2026). Hawkes-style priors aggregate history into mark-level rates; EventSS operates over scheduled event queues, making one-to-one triggers, overlapping workflows, and history-dependent gating natural to express. Remarkably, this queue semantics is already standard in discrete-event and business-process simulation, where pending events are maintained explicitly and executed in scheduled order for a specified system (Law, 2015; Palmer et al., 2019; QBP Simulator Developers, 2026; Automated Process Improvement, 2026). EventSS samples many random systems during pretraining so EventFM learns to generalize across graphs, sources, schedules, and delay laws.

Next-event forecasting. We focus on categorical marked event sequences, where each event (τ_i, κ_i) consists of a continuous occurrence time τ_i, $0 < \tau_1 < \tau_2 < \cdots$, and a categorical label, or *mark* $\kappa_i \in \mathcal{K}$. The mark space $\mathcal{K}$ contains $K = |\mathcal{K}|$ event types and may differ across sequences. Given the observed history, $\mathcal{H}_n = \big((\tau_i, \kappa_i)\big)_{i=1}^{n}$, the next-event forecasting task is to predict both the time τ_{n+1} and mark κ_{n+1} of the next event. Formally, we model the joint conditional law, representing it through the chain-rule factorization for measurable $A \subset (\tau_n, \infty)$ and $k \in \mathcal{K}$:

$$P_n(A, k) = \Pr(\kappa_{n+1} = k \mid \mathcal{H}_n)\, F(\tau_{n+1} \in A \mid k, \mathcal{H}_n). \tag{1}$$

EventFM parameterizes $\Pr(\kappa_{n+1} = k \mid \mathcal{H}_n)$ as a categorical distribution over candidate marks and represents $F(\cdot \mid k, \mathcal{H}_n)$ through conditional quantiles, $Q_\alpha(k, \mathcal{H}_n) = F^{-1}_{\tau|k,\mathcal{H}_n}(\alpha), \alpha \in \mathcal{A}$. The mark distribution is trained by cross-entropy, and the mark-conditional time distribution is trained by multi-quantile loss using the realized next mark.

Sequence-level zero-shot forecasting. We call a forecasting model *zero-shot* when a frozen model is applied to a previously-unseen target sequence, using only that sequence's observed history $\mathcal{H}_n$: no additional sequences from the target process, target-side parameter updates, model selection, or hyperparameter tuning are permitted. Thus, all information about the target event process available to EventFM at prediction time must be inferred in context from the single history being forecast.

3 EventFM

EventFM is built around the joint nature of events: each event couples a categorical mark, *what* occurs, with a continuous timestamp, *when* it occurs. Both acquire meaning through their relation to the preceding event history. This distinguishes event forecasting from conventional time-series models, which operate on numerical values over a shared temporal grid; language models, which operate on discrete tokens from a global vocabulary; and tabular models whose feature columns have fixed semantics for a single prediction target. Consequently, both the architecture and the pretraining prior of EventFM are designed around the joint structure of time, mark, and sequence history.

3.1 EventFM Architecture

Architecturally, EventFM treats time and marks as process-local: time has no fixed origin or scale, mark identifiers carry no global semantics, and prediction conditions time on type. Figure 1 summarizes the event-token representation, candidate-query construction, dual-axis transformer, and joint decoding procedure.

Event encoding. Given an observed history, $\mathcal{H}_n = \big((\tau_i, \kappa_i)\big)_{i=1}^{n}$, let $s_n = \tau_n - \tau_1$ denote the observed time span and $\bar{\Delta}_n = s_n/(n-1)$ the mean context inter-event time. Each event (τ_i, κ_i) is represented by four embedding blocks: a time embedding, a mark-identity embedding, a mark-context embedding, and an ordinal-position embedding:

$$e_i^{\text{time}} = E_{\text{time}}(\tau_i \mid \mathcal{H}_n) = \text{Proj}_{\text{time}}\left(\frac{\tau_i - \tau_1}{s_n}, \frac{\Delta\tau_i}{\bar{\Delta}_n}\right), \quad e_i^{\text{type}} = E_{\text{type}}(\kappa_i) = \omega_{\pi_n(\kappa_i)},$$

$$e_i^{\text{context}} = E_{\text{context}}(\kappa_i \mid \mathcal{H}_i), \quad e_i^{\text{position}} = E_{\text{position}}(i-n).$$

Here, $\Delta\tau_1 = 0$ and $\Delta\tau_i = \tau_i - \tau_{i-1}$ for $i > 1$. The time embedding records the event's location within the observed span and its inter-event time relative to the mean context gap. Both quantities are invariant to the absolute origin and units of the timeline. For the mark-identity embedding, let $\mathcal{K}_n = \{\kappa_1, \ldots, \kappa_n\}$ denote the marks observed in $\mathcal{H}_n$ and $K_n = |\mathcal{K}_n|$. At each forward pass, a fresh permutation π maps each type ID to a code vector $\omega_{\pi(\kappa_i)}$ from dictionary $\{\omega_j\}_{j=1}^{K_{\max}}$. This preserves type equality within the pass while preventing persistent type-specific embeddings across training. The mark-context embedding e_i^{context} summarizes sequence-local usage of κ_i through its prefix frequency and distance to the previous occurrence. Previous occurrence d events back is mapped to $d/(d+1)$, with first occurrences assigned one. The position embedding e_i^{position} encodes the ordinal offset from the forecast origin using sinusoidal features at multiple periods. The four embedding blocks are stacked as separate feature streams: $X_i = [e_i^{\text{time}}, e_i^{\text{type}}, e_i^{\text{context}}, e_i^{\text{position}}] \in \mathbb{R}^{4\times d}$. A deterministic feature-position encoding is added so that the shared feature-attention operation can distinguish the four streams.

Candidate-query construction. The model predicts only among the marks in $\mathcal{K}_n$. For every candidate $k \in \mathcal{K}_n$, it creates a query with the same unknown-time, mark-context, and next-position streams, but with the candidate's anonymous type code in the mark stream: $X_{\text{Q},k} = [q^{\text{time}}, \omega_{\pi_n(k)}, q^{\text{context}}, E_{\text{position}}(1)]$, where q^{time} and q^{context} are shared learned query embeddings. The backbone input concatenates the observed events and all candidate queries along the event axis: $Z^{(0)} = [X_1, \ldots, X_n, X_{\text{Q},k_1}, \ldots, X_{\text{Q},k_{K_n}}] \in \mathbb{R}^{(n+K_n)\times 4\times d}$. Across a batch, candidate rows are padded to the largest visible candidate count and invalid rows are masked. Types absent from a sequence's context are not candidate queries and receive zero probability.

Dual-Axis In-Context Backbone. Following the PFN principle of in-context prediction by attention (Hollmann et al., 2023), each backbone layer alternates feature and event attention. Feature attention combines the time, mark, context, and position streams within each event or query; event attention then propagates information across the observed history and candidate queries:

$$\widetilde{Z}^{(\ell)} = \text{Attn}_{\text{feature},\ell}\big(Z^{(\ell)}\big), \quad \overline{Z}^{(\ell)} = \text{Attn}_{\text{event},\ell}\big(\widetilde{Z}^{(\ell)}; \mathcal{M}_n\big), \quad Z^{(\ell+1)} = \text{Update}_\ell\big(\overline{Z}^{(\ell)}\big).$$

Here, Update_ℓ denotes the residual, normalization, and pointwise transformations of layer ℓ. The mask $\mathcal{M}_n$ restricts context events to context, while candidate queries attend to both context and one another, letting alternative next-event types compete within each layer. With no candidate-order encoding and a shared next-event position, candidate attention is permutation equivariant and naturally handles varying numbers of candidate event types with the same learned backbone.

Direct candidate type head. Let $h^{\kappa}_{\text{Q},k} = Z^{(L)}_{\text{Q},k,1}$ denote the final type-stream state of candidate query k. A shared MLP assigns each candidate a logit,

$$\ell_k = \text{TypeHead}\big(h^{\kappa}_{\text{Q},k}\big), \qquad p(\kappa_{n+1} = k \mid \mathcal{H}_n) = \frac{\exp(\ell_k)}{\sum_{k'\in\mathcal{K}_n}\exp(\ell_{k'})}. \tag{2}$$

Sharing the head across candidate marks makes prediction permutation equivariant in mark order and agnostic to the number of observed event types.

Candidate-conditional time head. Let $h^{\tau}_{\text{Q},k} = Z^{(L)}_{\text{Q},k,0}$ denote the final time-stream state of candidate k. A shared MLP maps it to strictly increasing, median-anchored normalized-delay quantiles $(q_{k,\alpha})_{\alpha\in\mathcal{A}}$. The corresponding absolute-time quantiles are

$$\widehat{Q}_\alpha\left(\tau_{n+1} \mid \kappa_{n+1} = k, \mathcal{H}_n\right) = \tau_n + \bar{\Delta}_n q_{k,\alpha}, \qquad k \in \mathcal{K}_n. \tag{3}$$

Interpolating these quantiles gives the mark-conditional CDF $F(\tau \mid k, \mathcal{H}_n)$.

Together, these outputs instantiate the joint law in equation 1. Marks outside $\mathcal{K}_n$ receive zero probability.

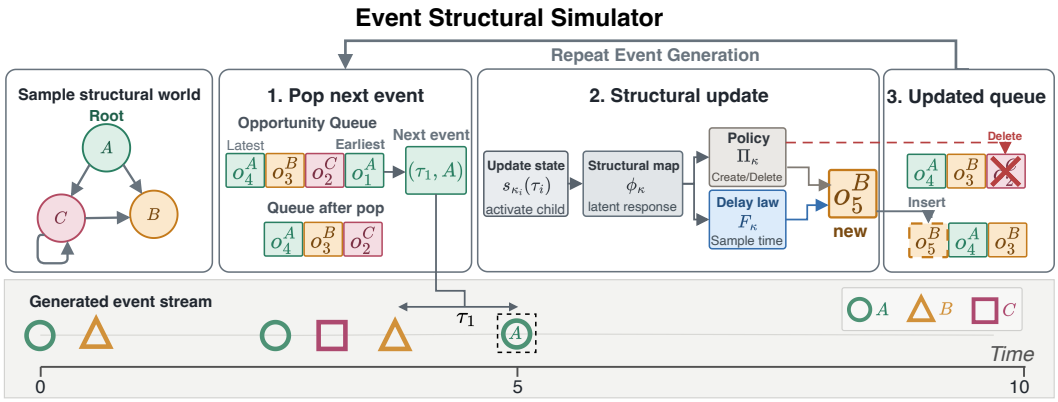


Figure 2: **Event Structural Simulator (EventSS).** EventSS samples an event-dependency graph and repeatedly emits the earliest queued opportunity. Each event updates structural state, creates, deletes, or schedules future opportunities, and is appended to the generated event stream.

3.2 EventFM Pretraining Prior: Event Structural Simulator

We design a prior over *event-instance relations*, rather than aggregate event rates. EventSS (*Event Structural Simulator*) is an on-the-fly synthetic prior that samples a new event process for each training sequence, varying its graph, time scale, mark vocabulary, delay laws, and history length. Unlike additive Hawkes-style priors that superpose past-event effects into aggregate intensities (Appendix A), EventSS maintains distinct pending event instances that can create, modify, or resolve future opportunities. This naturally expresses one-to-one triggers, overlapping workflows, delayed or cancellable consequences, recurrence, and non-additive interactions, while enabling direct simulation without thinning. Training on these random systems teaches EventFM to infer event dynamics from context. Figure 2 summarizes the cycle. Appendix E gives full details.

Sampled worlds. EventSS samples a *world*, an event-generating process fixed within a sequence and resampled across sequences. Shared hyperparameters $\xi \sim P(\xi)$ coordinate temporal scales, graph structure, and mechanism complexity, encouraging coherent combinations rather than independently sampled components. The dependency structure is a directed graph $G = (\mathcal{V}, \mathcal{E})$ over event types $\mathcal{V} = \{1, \dots, K\}$. For structural worlds, EventSS samples G from DAG archetypes including pipelines, fan-in/out graphs, diamonds, layered DAGs, cascade trees, and grown DAGs; see Appendix E. For each type κ, let $\mathcal{P}_\kappa = \{j : (j, \kappa) \in \mathcal{E}\}$ denote its parents, inducing roots $\mathcal{R} = \{\kappa : \mathcal{P}_\kappa = \emptyset\}$ and non-roots $\mathcal{N} = \mathcal{V} \setminus \mathcal{R}$. An edge $j \to \kappa$ means that an event of type j may activate type κ's generative mechanism. Self-activation is sampled separately through the recurrence policy. Root types $r \in \mathcal{R}$ initiate activity exogenously through sampled source processes B_r.

For each structural world, EventSS samples a dependency graph G, a state encoder S_κ for each event type, a source process B_r for each root type $r \in \mathcal{R}$, and for each non-root type $\kappa \in \mathcal{N}$ a structural map ϕ_κ, opportunity policy Π_κ, and delay law F_κ. These components are coupled through shared world-level hyperparameters ξ:

$$\big(G, \{S_\kappa\}_{\kappa=1}^{K}, \{B_r\}_{r\in\mathcal{R}}, \{(\phi_\kappa, \Pi_\kappa, F_\kappa)\}_{\kappa\in\mathcal{N}}\big) \sim P(\cdot \mid \xi), \qquad \xi \sim P(\xi). \tag{4}$$

Markov worlds use a restricted parameterization consisting of a transition matrix and per-mark delay laws. Each event creates exactly one successor opportunity, whose type is sampled from the transition matrix and whose delay is drawn from the corresponding law. Both modes use the same event queue and produce the same marked-sequence representation.

Opportunity scheduling. EventSS executes each sampled world as a discrete-event simulation over a set $\mathcal{O}$ of pending *opportunities*. Each opportunity represents a scheduled candidate event with a target mark, proposed occurrence time, and state determining whether it occurs, expires, or is modified before execution. Maintaining these opportunities separately preserves instance-level event relations rather than collapsing historical influence into aggregate rates.

State encoder. Each mark κ maintains a latent state $s_\kappa(t)$ summarizing its recent activity. When κ is triggered, its state together with those of its parents parameterizes the structural map, opportunity policy, and delay law used to update the event queue. The state encoder is parameterized through a perceptron operating on a vector of summary statistics of type specific event history - see Appendix E.4 for details.

Root types $r \in \mathcal{R}$ receive exogenous sources B_r (Appendix E.3) that seed $\mathcal{O}$ with parentless opportunities. Each non-root type $\kappa \in \mathcal{N}$ receives a generative mechanism $(\phi_\kappa, \Pi_\kappa, F_\kappa)$ comprising a structural map, opportunity policy, and conditional delay law, introduced next.

Structural map ϕ_κ. When an event of parent type $j \in \mathcal{P}_\kappa$ occurs, ϕ_κ maps the current target, parent-set, and triggering-parent states to a latent response

$$z_{\kappa \leftarrow j}(t) = \phi_\kappa(s_\kappa(t), \mathbf{s}_{\mathcal{P}_\kappa}(t), s_j(t)) \,. \tag{5}$$

This response conditions both the opportunity policy and delay law, determining how the trigger affects future events of type κ. In practice, ϕ_κ also receives persistent mark attributes and is sampled with the world as a random projection with complexity-dependent nonlinearity; see Appendix E.5.

Opportunity policy Π_κ. Let $\mathcal{O}^-$ and $\mathcal{O}^+$ denote the opportunity set immediately before and after applying the mechanism for type κ. The policy updates the queue according to

$$\mathcal{O}^+ = \Pi_\kappa\big(\mathcal{O}^-, z_{\kappa \leftarrow j}(t)\big) \,. \tag{6}$$

It may create, remove, or modify pending opportunities, enabling delayed effects, cancellation, gating, and interactions among overlapping event relations; see Appendix E.6.

Delay law F_κ. Each newly created opportunity o of type κ is scheduled by drawing $u_o \sim \mathrm{Uniform}(0,1)$ and adding a positive, state-dependent delay to the triggering time:

$$\widetilde{\tau}_o = t + F_\kappa^{-1}(u_o \,|\, z_{\kappa \leftarrow j}(t)) \,. \tag{7}$$

Thus, event timing depends on the target mark, triggering parent, and current event-history state, with delay laws drawn from exponential, gamma, Weibull and other families; see Appendix E.7.

Event selection. EventSS executes the earliest scheduled opportunity,

$$o^\star = \arg\min_{o \in \mathcal{O}} \widetilde{\tau}_o, \qquad (\tau_i, \kappa_i) = (\widetilde{\tau}_{o^\star}, \kappa_{o^\star}). \tag{8}$$

EventSS consumes $o^\star$, emits the event, and updates type states. In structural worlds, each outgoing edge $\kappa_i \to \kappa$ activates the child mechanism: ϕ_κ produces the response, Π_κ updates the queue, and F_κ schedules new or rescheduled opportunities. In Markov worlds, κ_i selects one successor mark and schedules one opportunity from its delay law. The cycle repeats to the target sequence length.

3.3 Pretraining and Zero-Shot Inference

Training objective. EventFM is trained on next-event triples $(\mathcal{H}_n, \tau_{n+1}, \kappa_{n+1})$. Candidate-type logits use cross-entropy, while the time head uses multi-quantile loss for the realized next mark. Let $\widehat{Q}_\alpha(\kappa_{n+1}, \mathcal{H}_n)$ denote its predicted type-conditioned time α-quantile, $\rho_\alpha(u) = u(\alpha - \mathbf{1}\{u < 0\})$, the pinball loss, and λ_τ and λ_κ time and type weights, then:

$$\mathcal{L} = \frac{\lambda_\tau}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} \rho_\alpha\Big(\tau_{n+1} - \widehat{Q}_\alpha(\kappa_{n+1}, \mathcal{H}_n)\Big) - \lambda_\kappa \log p(\kappa_{n+1} \mid \mathcal{H}_n). \tag{9}$$

Both arguments of ρ_α are divided by the context mean gap $\bar{\Delta}_n$ before the loss is evaluated. Training data come entirely from EventSS and Appendix C gives full training details.

Zero-shot inference. At inference, EventFM is fully frozen: a target sequence only determines its normalized inputs, randomized mark assignment, and candidate queries. No parameter updates, model selection, or tuning are performed. For mark accuracy, prediction uses $\widehat{\kappa}_{n+1} = \arg\max_{k \in \mathcal{K}_n} p(\kappa_{n+1} = k \mid \mathcal{H}_n)$. For point forecast, we use the predictive median, including for RMSE, which is directly available from the predicted quantiles. For autoregressive decoding, we sample the mark from $p(\kappa \mid \mathcal{H}_n)$, optionally applying temperature T_κ to the mark logits, and use the corresponding mark-conditional predictive median for time. Autoregressive forecasts append each resulting event and repeat with the frozen model.

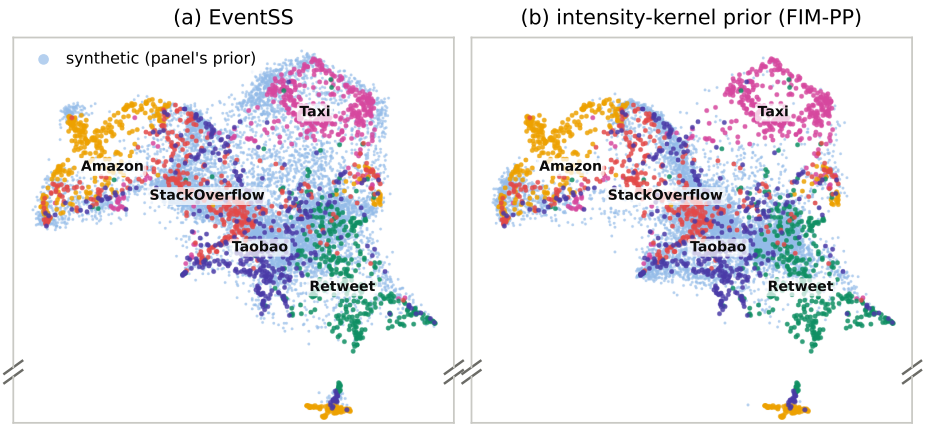


Figure 3: **Synthetic-prior coverage.** In a shared 32-feature UMAP, EventSS covers more of the real-data manifold than the intensity-kernel prior. UMAP is fit on 2000 real sequences, with 9926 prior samples projected afterward.

Theoretical Analysis Appendices G–I analyze an idealized abstraction of EventSS and the objective in equation 9. Because EventFM normalizes time and relabels marks, the population minimizer is a Bayesian posterior predictive for the *normalized* next event. The world, time scale, and mark identities are latent, and the outputs are mark probabilities and conditional quantiles. Relative to an oracle that knows the world, its cumulative excess log loss on EventSS data is at most $\log(1/P(S^\star)) + \frac{p}{2}\log N + O(1)$, where $S^\star$ is the world's discrete structure and p its effective number of continuous parameters (Thm. 1). Thus, structure has bounded cost and parameters are learned at average rate $O(p \log N/N)$. On real data, our oracle inequality (Thm. 2) adds a prior-determined misspecification gap that longer contexts cannot remove. These results characterize the form of in-context learning and the role of the prior rather than provide numerical guarantees for short contexts. We adapt classical techniques (Clarke & Barron, 1990; Barron, 1999) to EventFM's normalized, quantile-trained PFN setting (Müller et al., 2022; Nagler, 2023).

4 Empirical Results

We evaluate EventFM on five MTPP benchmarks: Retweet, Taxi, Taobao, StackOverflow, and Amazon, using EasyTPP splits (Xue et al., 2024). They cover social, mobility, e-commerce, and online-community processes, with three to 22 event types. The synthetic suite isolates deterministic cycles and counting patterns, stochastic excitation and seasonality, and mixed request–response and shipment-delivery workflows. All targeted scenarios use purpose-built generators outside EventSS and are intentionally out of distribution, with the strongest shift in the fully deterministic cases. Appendix B.1 provides detailed dataset descriptions and statistics.

Tasks and metrics. For next-event prediction, each $i \in \{2, \ldots, N-1\}$ defines a forecast origin: the model observes events through i and predicts $i+1$. We report mark accuracy, time MAE, time RMSE, time CRPS, and mark Brier score. For long-horizon forecasting, the model observes through $N-8$ and predicts the final eight events, appending each before the next. We report optimal transport distance (OTD) and per-type count RMSE. Appendix B.2 gives scoring details.

Baselines. All methods follow the same sequence-level zero-shot protocol, conditioning only on the observed target-sequence history. We compare EventFM with mean/mode and last-event baselines, Qwen3.5-9B, and zero-shot adaptations of FIM-PP and EVIL; see Appendix B.4 for details.

4.1 EventSS Coverage of Real Event Data

Figure 3 compares EventSS with an additive intensity-kernel prior across five real corpora in a scale-invariant 32-feature space. EventSS covers substantially more of the real-data manifold. Its nearest synthetic neighbour has mean rank 13.2 versus 19.0, is closer in raw distance for 69.4%

Table 1: Next-event prediction on real and synthetic datasets. Accuracy scores the predicted type (↑); CRPS scores the time distribution (↓). Synthetic results average scenarios by group, with CRPS normalized by mean inter-arrival time. Ablations replace EventSS with an intensity-kernel prior or remove context/position features. Best and second-best zero-shot results are bold and underlined.

	Real					**Synthetic**		
Model	Retweet	Taxi	Taobao	StackOv.	Amazon	Det.	Stoch.	Mixed
				Type accuracy ↑				
Mean/mode	<u>0.589</u>	0.191	0.515	**0.430**	**0.338**	0.277	0.384	0.364
Last	0.534	0.000	**0.602**	0.303	0.218	0.233	**0.553**	0.055
Qwen3.5 9B	0.582	0.749	0.567	0.401	0.318	0.647	0.466	0.548
FIM-PP (zs)	0.521	0.280	0.373	0.330	0.255	0.377	0.342	0.342
EVIL (zs)	0.571	**0.858**	0.564	0.376	0.292	<u>0.789</u>	0.487	**0.714**
EventFM	**0.592**	<u>0.855</u>	<u>0.596</u>	<u>0.414</u>	<u>0.319</u>	**0.801**	<u>0.515</u>	<u>0.687</u>
Ablation w/o EventSS	0.594	0.182	0.599	0.421	0.329	0.286	0.510	0.283
Ablation w/o ctx./pos.	0.584	0.814	0.553	0.415	0.320	0.521	0.443	0.530
				CRPS ↓				
Mean/mode	2.5k	0.174	0.061	0.757	0.282	0.346	0.975	0.924
Last	2.8k	0.216	0.067	0.896	0.301	0.542	1.136	1.313
Qwen3.5 9B	2.7k	0.163	0.044	0.717	0.320	<u>0.234</u>	0.869	0.795
FIM-PP (zs)	2.5k	0.166	**0.042**	<u>0.642</u>	0.440	0.488	<u>0.736</u>	<u>0.717</u>
EVIL (zs)	<u>2.4k</u>	<u>0.158</u>	0.049	0.714	<u>0.270</u>	0.292	0.881	0.732
EventFM	**2.2k**	**0.137**	<u>0.042</u>	**0.556**	**0.225**	**0.145**	**0.701**	**0.525**
Ablation w/o EventSS	2.3k	0.127	0.041	0.516	0.248	0.361	0.679	0.672
Ablation w/o ctx./pos.	2.2k	0.135	0.042	0.554	0.224	0.246	0.698	0.621

of sequences, and yields lower normalized energy distance on every corpus (0.14 versus 0.21 on average). Appendix J gives the protocol and corpus-level results.

4.2 Quantitative Forecasting Performance

Next-event prediction. Table 1 reports type accuracy from the point prediction and CRPS from the time distribution. Across the five real datasets and three synthetic groups, EventFM ranks in the top two in all 16 comparisons for accuracy and CRPS. The next-highest top-two count is EVIL with six of 16. EventFM has the highest accuracy on Retweet and the deterministic synthetic group, and ranks second in the other six accuracy columns. For CRPS, EventFM is best in seven of eight columns and second only on Taobao, where it is within 0.001 of FIM-PP. Thus, although the leading accuracy method varies among EventFM, mean/mode, the last-event heuristic, and EVIL, EventFM is the only method that is almost always competitive. Appendix Tables 10, 11, and 12 report all seven metrics with sequence-bootstrap 95% confidence intervals. Across all seven metrics and eight columns, EventFM ranks in the top two in 50 of 56 comparisons (89.3%); EVIL is next with 25 of 56 (44.6%). Their leaders vary more by dataset and metric, particularly for long-horizon forecasting.

4.3 EventSS Reconstruction and Distribution Shifts

On held-out EventSS sequences, EventFM outperforms mean/mode and last on mark accuracy, time RMSE, CRPS, and Brier score. It retains this next-event lead under shifts in event-type cardinality, temporal scale, delay-law family, and mechanism composition. For eight-event rollouts, it matches mean/mode on OTD and leads on count RMSE both in and out of distribution. Appendices B and K give protocols and results.

4.4 Forecast Trajectories

Figure 4 exposes complementary failures. FIM-PP's training prior lets each past event change a target intensity only through a fixed source–target type-pair kernel. It has no state that counts position within a run, so it cannot represent the 3A/2B rule and falls back to a mean/mode-like forecast. Al-

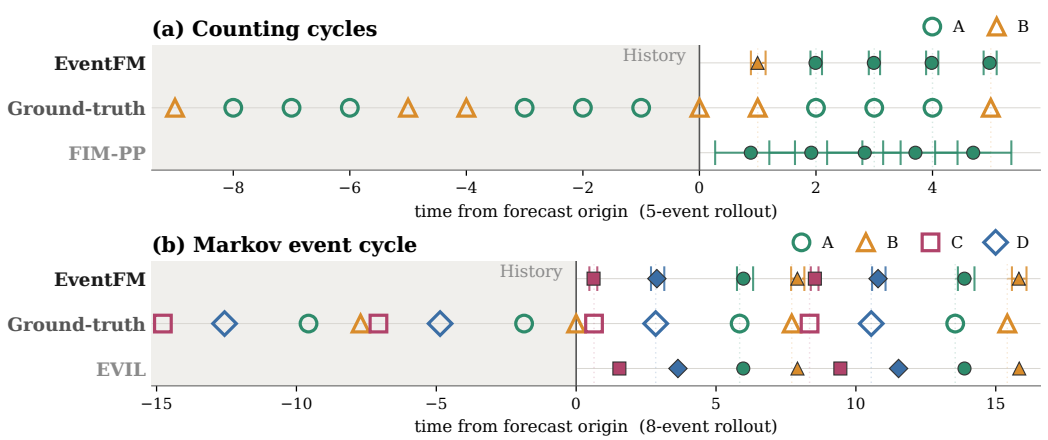


Figure 4: **Zero-shot trajectory comparisons.** Filled markers show model medians, hollow markers show ground truth, and capped bars show P10–P90 intervals. Dashed bars are widened only enough to remain visible without moving their medians. (a) EventFM continues a 3A/2B counting cycle, whereas FIM-PP falls back to modal/mean behavior. (b) EventFM recovers a type-dependent Markov cycle. EVIL predicts the event types but not their narrow timing.

though every event is exactly one time unit apart, FIM-PP's mean P10–P90 distance is 1.363 versus 0.034 for EventFM. EVIL has the opposite problem. It classifies all eight Markov-cycle events correctly but misses their narrow type-dependent timing, consistent with its worse deterministic-dataset CRPS (0.292 versus 0.145 for EventFM). The shown errors were selected to represent each model's average metrics on the corresponding dataset. Appendix F provides further examples.

4.5 EventFM and EventSS Ablations

EventFM. The w/o context/position ablation in Table 1 removes e_i^{context} and e_i^{position}, retaining the time and type streams. It preserves most real-data results but lowers Taxi, Taobao, and all three synthetic-group accuracies. Timing degrades mainly on deterministic and mixed groups, showing that these sequence-relative embeddings primarily aid event-type structure. Further ablations show that isolating feature streams sharply harms mark prediction, while concatenating them restores competitive accuracy. Appendix B.6.1 gives full results.

EventSS. The w/o EventSS ablation holds the architecture fixed and replaces EventSS with the additive intensity-kernel prior used by FIM-PP (Berghaus et al., 2026). It preserves several real-data results but sharply lowers Taxi and deterministic and mixed synthetic accuracy, indicating narrower transfer. Component ablations support complementary structural and Markov pathways and multi-event memory, while finer prior removals have mixed effects. Appendix B.6.2 gives full results.

5 Conclusion

We introduced EventFM, an event-native foundation model for sequence-level zero-shot forecasting, and EventSS, a synthetic prior over queue-based event relations. Across five real benchmarks and three synthetic groups, one frozen EventFM ranks in the top two in every main accuracy and CRPS comparison and leads CRPS in seven of eight settings. EventSS covers real event processes more broadly than an additive intensity-kernel prior, while controlled shifts and ablations show that the event representation, structural and Markov pathways, and multi-event memory support transfer. These results move event forecasting toward the train-once inference paradigm that has made foundation models useful in language, time series, and tabular data, while opening a practical path for cold-start and privacy-constrained domains where target-specific training is difficult. Improving long-horizon mark forecasts and incorporating event metadata remain important directions.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, programming assistance, results visualizations and paper drafting and editing. We have not used generative AI tools for generating datasets, designing methods, proposing hypotheses, interpreting results, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work. LLM-generated content was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

L Adamopoulos. Some counting and interval properties of the mutually-exciting processes. *Journal of Applied Probability*, 12(1):78–86, 1975.

Yacine Aït-Sahalia, Julio Cacho-Diaz, and Roger J. A. Laeven. Modeling financial contagion using mutually exciting jump processes. *Journal of Financial Economics*, 117(3):585–606, 2015.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=gerNCVqqtR. Expert Certification.

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, Huibin Shen, Lorenzo Stella, Xiyuan Zhang, Mononito Goswami, Shubham Kapoor, Danielle C. Maddix, Pablo Guerron, Tony Hu, Junming Yin, Nick Erickson, Prateek Mutalik Desai, Hao Wang, Huzefa Rangwala, George Karypis, Yuyang Wang, and Michael Bohlke-Schneider. Chronos-2: From univariate to universal forecasting. arXiv preprint arXiv:2510.15821, 2025. URL https://arxiv.org/abs/2510.15821.

Automated Process Improvement. Prosimos: Business process simulation engine. https://github.com/AutomatedProcessImprovement/Prosimos, 2026. Accessed 2026-06-17.

Wonho Bae, Mohamed Osama Ahmed, Frederick Tung, and Gabriel L. Oliveira. Meta temporal point processes. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=QZfdDpTX1uM. Code available at https://github.com/BorealisAI/meta-tpp.

Andrew R. Barron. Information-theoretic characterization of Bayes performance and the choice of priors in parametric and nonparametric problems. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (eds.), *Bayesian Statistics 6: Proc. Valencia International Meeting*, pp. 27–52. Oxford University Press, 1999.

David Berghaus. EVIL: Evolving interpretable algorithms for zero-shot inference on event sequences and time series with LLMs. *arXiv preprint arXiv:2604.15787*, 2026.

David Berghaus, Patrick Seifner, Kostadin Cvejoski, César Ojeda, and Ramsés J. Sánchez. In-context learning of temporal point processes with foundation inference models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2509.24762.

Yuxin Chang, Alex Boyd, Cao Xiao, Taha Kass-Hout, Parminder Bhatia, Padhraic Smyth, and Andrew Warrington. Deep continuous-time state-space models for marked event sequences. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, 2025. URL https://arxiv.org/abs/2412.19634. Known as S2P2 (State-Space Point Process).

Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. Neural spatio-temporal point processes. In *International Conference on Learning Representations (ICLR)*, 2021.

Bertrand S. Clarke and Andrew R. Barron. Information-theoretic asymptotics of Bayes methods. *IEEE Trans. Information Theory*, 36(3):453–471, 1990.

Daryl J. Daley and David Vere-Jones. *An Introduction to the Theory of Point Processes, Volume I: Elementary Theory and Methods*. Springer, 2 edition, 2003. doi: 10.1007/b97277.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha Naidu, and Colin White. ForecastPFN: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2311.01933.

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 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, KDD 2016, pp. 1555–1564, New York, NY, USA, 2016. Association for Computing Machinery. URL https://doi.org/10.1145/2939672.2939875.

Manisha Dubey, P. K. Srijith, and Maunendra Sankar Desarkar. HyperHawkes: Hypernetwork based neural temporal point process. arXiv preprint arXiv:2210.00213, 2022.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Peter D. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

Alan G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Sihao Hu, Zhen Zhang, Bingqiao Luo, Shengliang Lu, Bingsheng He, and Ling Liu. Bert4eth: A pre-trained transformer for ethereum fraud detection. In *Proceedings of the ACM Web Conference (WWW)*, 2023. doi: 10.1145/3543507.3583345. URL https://arxiv.org/abs/2303.18138. Code and datasets available at https://github.com/git-disl/BERT4ETH.

Ayush Jain, Rajat Sen, Yichen Zhou, Petros Mol, Abhimanyu Das, and Samet Oymak. TimesFM-3: A zero-shot foundation model for multivariate forecasting, 2026. URL https://research.google/blog/timesfm-3-a-zero-shot-foundation-model-for-multivariate-forecasting/. Google Research.

Gavin Kerrigan, Kai Nelson, and Padhraic Smyth. EventFlow: Forecasting temporal point processes with flow matching. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2026. URL https://arxiv.org/abs/2410.07430. AISTATS 2026 Best Paper Award; code available at https://github.com/GavinKerrigan/eventflow.

J. F. C. Kingman. *Poisson Processes*. Oxford Studies in Probability. Oxford University Press, 1993.

Averill M. Law. *Simulation Modeling and Analysis*. McGraw-Hill Education, 5 edition, 2015.

Jeong Min Lee and Milos Hauskrecht. Modeling multivariate clinical event time-series with recurrent temporal mechanisms. *Artificial Intelligence in Medicine*, 112:102021, 2021.

Rong Liu, Akhil Kumar, and Wil M. P. van der Aalst. A formal modeling approach for supply chain event management. *Decision Support Systems*, 43(3):761–778, 2007.

Anton Lysenko, Egor Shikov, and Klavdiya Bochenina. Temporal point processes for purchase categories forecasting. *Procedia Computer Science*, 156:255–263, 2019.

Hongyuan Mei, Guanghui Qin, and Jason Eisner. Imputing missing events in continuous-time event streams. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. URL https://arxiv.org/abs/1905.05570.

Vladyslav Moroshan, Julien Siems, Arber Zela, Timur Carstensen, and Frank Hutter. TempoPFN: Synthetic pre-training of linear rnns for zero-shot time series forecasting, 2025. URL https://arxiv.org/abs/2510.25502.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *Proc. Int. Conf. Learning Representations (ICLR)*, 2022.

Thomas Nagler. Statistical foundations of prior-data fitted networks. In *Proc. Int. Conf. Machine Learning (ICML)*, pp. 25660–25676, 2023.

Yosihiko Ogata. On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981.

Yosihiko Ogata. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical Association*, 83(401):9–27, 1988.

Takahiro Omi, Naonori Ueda, and Kazuyuki Aihara. Fully neural network based model for general temporal point processes. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/39e4973ba3321b80f37d9b55f63ed8b8-Abstract.html.

Boris N. Oreshkin, Mayank Jauhari, Ravi Kiran Selvam, Malcolm Wolff, Wenhao Pan, Shankar Ramasubramanian, Kin G. Olivares, Tatiana Konstantinova, Andres Potapczynski, Mengfei Cao, Dmitry Efimov, Michael W. Mahoney, and Andrew G. Wilson. Zero-shot forecasting by simulation alone. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2601.00970.

Maxim Ostroukhov, Ruslan Mikhailov, Vladimir Iashin, Artem Sokolov, Andrei Akshonov, Vitaly Protasov, Andrey Goncharov, Dmitrii Beloborodov, Vince Mullin, Roman Yokunda Enzmann, Georgios Kolovos, Jason Renders, Pavel Nesterov, and Anton Repushko. Pragma: Revolut foundation model, 2026. URL https://arxiv.org/abs/2604.08649.

Geraint I. Palmer, Vincent A. Knight, Paul R. Harper, and Asyl L. Hawa. Ciw: An open source discrete event simulation library. *Journal of Simulation*, 13(1):68–82, 2019.

Chao Pang, Xinzhuo Jiang, Krishna S. Kalluri, Matthew Spotnitz, RuiJun Chen, Adler Perotte, and Karthik Natarajan. CEHR-BERT: Incorporating temporal information from structured EHR data to improve prediction tasks. In *Proceedings of Machine Learning for Health (ML4H)*, volume 158 of *Proceedings of Machine Learning Research*, 2021. URL https://proceedings.mlr.press/v158/pang21a.html. Code available at https://github.com/cumc-dbmi/cehrbert.

QBP Simulator Developers. BIMP and QBP Simulator. https://bimp.cs.ut.ee/, 2026. Accessed 2026-06-17.

Pawel Renc, Michal K. Grzeszczyk, Nassim Oufattole, Deirdre Goode, Yugang Jia, Szymon Bieganski, Matthew B. A. McDermott, Jaroslaw Was, Anthony E. Samir, Jonathan W. Cunningham, David W. Bates, and Arkadiusz Sitek. Foundation model of electronic medical records for adaptive risk estimation. *arXiv preprint arXiv:2502.06124*, 2025. URL https://arxiv.org/abs/2502.06124. Builds on the ETHOS patient-health-timeline tokenization framework.

Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural temporal point processes: A review. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 4585–4593, 2021. doi: 10.24963/ijcai.2021/623.

Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. DyRep: Learning representations over dynamic graphs. In *International Conference on Learning Representations (ICLR)*, 2019.

Wilson Truccolo, Uri T. Eden, Matthew R. Fellows, John P. Donoghue, and Emery N. Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, 93(2):1074–1089, 2005.

Song Wei, Yao Xie, Christopher S. Josef, and Rishikesan Kamaleswaran. Granger causal chain discovery for sepsis-associated derangements via continuous-time Hawkes processes. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2023. doi: 10.1145/3580305.3599369.

Shifeng Xie, Vasilii Feofanov, Marius Alonso, Ambroise Odonnat, Jianfeng Zhang, Themis Palpanas, and Ievgen Redko. CauKer: Classification time series foundation models can be pretrained on synthetic data. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2508.02879.

Siqiao Xue, Xiaoming Shi, James Y. Zhang, and Hongyuan Mei. HYPRO: A hybridly normalized probabilistic model for long-horizon prediction of event sequences. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022. URL https://arxiv.org/abs/2210.01753. Code available at https://github.com/ant-research/hypro_tpp.

Siqiao Xue, Xiaoming Shi, Zhixuan Chu, Yan Wang, Hongyan Hao, Fan Zhou, caigao jiang, Chen Pan, James Zhang, Qingsong Wen, JUN ZHOU, and Hongyuan Mei. Easytpp: Towards open benchmarking temporal point processes. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 52354–52375, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/e66233a208ef32f56df6312263239fa0-Paper-Conference.pdf.

A Relation to Additive Intensity-Kernel Priors

The standard approach to synthetic event-sequence generation samples data from additive intensity-kernel processes (Berghaus et al., 2026; Berghaus, 2026). The conditional intensity of type κ takes the form

$$\lambda(t, \kappa \mid \mathcal{H}_t) = g\Big(\mu_\kappa(t) + \sum_{(\tau_j, \kappa_j) \in \mathcal{H}_t} \gamma_{\kappa\kappa_j}(t - \tau_j)\Big), \tag{10}$$

where $\mu_\kappa(t)$ is a base intensity, $\gamma_{\kappa\kappa'}(\cdot)$ is a signed interaction kernel describing the exciting or inhibiting effect of source type κ' on target type κ, and g is a monotone link ensuring nonnegativity. Diversity comes from sampling the base intensities and kernels from parametric families, such as constant, sinusoidal, or gamma-shaped baselines with exponential or Rayleigh kernels. This family covers classical and signed Hawkes processes and, in particular, the pretraining prior of FIM-PP (Berghaus et al., 2026). Its defining structural property is that the contribution of every past event is a fixed function of its type pair and elapsed time, pooled additively into a single rate per target type.

This appendix relates EventSS to that family from two directions. §A.1 describes what the pooling in equation 10 does not represent, and points to the empirical evidence for the resulting coverage gap. §A.2 then establishes the one formal relation between the two families that we prove: a restricted non-negative linear Hawkes subfamily is contained in the EventSS formalism.

A.1 What additive pooling does not represent

Both limitations below follow from the same property: every past contribution is pooled into one rate per target type.

No link between an event and the event that triggered it. In equation 10 each past event contributes to an aggregate rate, and events are drawn from that pooled rate. The contribution retains the *type* of the contributing event and its elapsed time, so the history is not discarded; what the parameterization does not carry is a link between a generated event and the particular earlier occurrence that triggered it, which a later event could then consume.

Suppose two types, orders and confirmations, where each order is answered by one confirmation. Let $n_{\mathrm{ord}}(t)$ and $n_{\mathrm{con}}(t)$ count the orders and the confirmations that have occurred by time t. A kernel that holds a constant value over the observation window contributes that value once per past event, so its pooled contribution is proportional to a count. Taking one such kernel positive from orders and one negative from confirmations therefore gives a confirmation rate proportional to $n_{\mathrm{ord}}(t) - n_{\mathrm{con}}(t)$, with the link g holding it at zero if the difference goes negative. That rate is high while orders are waiting and drops to zero once every order has been confirmed, so an additive model can indeed make the number of confirmations track the number of orders.

What it cannot do is say *which* order a given confirmation settles. The rate depends on the two totals alone, so no individual order is ever singled out and retired, and nothing prevents a confirmation from being generated long after the order it would nominally answer. EventSS carries the attachment directly: an order event inserts one pending confirmation opportunity, and it is that specific opportunity which the later confirmation consumes.

No history-dependent change of direction. Adding one event (τ', κ') to the history changes the intensity in a fixed way,

$$\lambda\big(t, \kappa \mid \mathcal{H}_t \cup \{(\tau', \kappa')\}\big) = g\Big(\eta_\kappa(t \mid \mathcal{H}_t) + \gamma_{\kappa\kappa'}(t - \tau')\Big), \tag{11}$$

where $\eta_\kappa(t \mid \mathcal{H}_t)$ denotes the pooled sum inside g in equation 10. The increment $\gamma_{\kappa\kappa'}(t - \tau')$ inside g does not depend on $\mathcal{H}_t$ and g is monotone, so the added event cannot move the intensity in opposite directions according to which other events are present. The claim is confined to that increment. A nonlinear g does introduce non-additive behaviour of its own through thresholding and saturation, and it makes the realized effect of an event depend on the rest of the history; what monotonicity rules out is a *reversal of sign*. Gating, in which a trigger matters only once an enabling event has occurred, can therefore arise through clipping, but only as a function of the aggregate rate rather than as a relation between two event instances.

Evidence. These are statements about what the two priors parameterize, not about what a model trained on either can ultimately fit, and the measurements below are consistent with the distinction rather than isolating it: the two priors differ in several respects at once, so neither attributes an observed gap to the pooled rate alone. Appendix J compares the event-sequence feature distributions of the sampled EventSS prior and of an additive intensity-kernel prior against real benchmarks. §B.6 specifies the ablation that replaces EventSS with such a prior while holding the architecture fixed, and Table 1 reports its downstream effect.

A.2 Containment of a non-negative linear Hawkes subfamily

The containment result uses the opportunity-scheduling formulation of EventSS from Section 3.2, specified formally in Appendix G. We show that for every non-negative linear Hawkes process with root-confined immigration and an excitation graph that is acyclic up to self-loops, there exists a structural world W inducing the same observable sequence law. The simulator prior of Berghaus et al. (2026) lies in the broader signed family and is *not* covered by this result.

Let $\mathcal{A}_{\pm}$ denote the family of Equation (10), and let

$$\mathcal{A}_{+} = \left\{ \lambda(t, \kappa \mid \mathcal{H}_t) = \mu_\kappa + \textstyle\sum_{(\tau_j, \kappa_j) \in \mathcal{H}_t} \gamma_{\kappa\kappa_j}(t - \tau_j) \;:\; \mu_\kappa \geq 0,\ \gamma_{\kappa\kappa'} \geq 0 \right\} \subset \mathcal{A}_{\pm} \tag{12}$$

be the non-negative sub-family of classical linear multivariate Hawkes processes, obtained with the identity link and constant baselines. Write $\Gamma_{\kappa\kappa'} = \int_0^\infty \gamma_{\kappa\kappa'}(u)\, du$ for the total branching mass of a kernel and $\Gamma = (\Gamma_{\kappa\kappa'})$ for the branching matrix.

The claim compares observable sequence distributions. For a generator G, let $\mathbb{P}_G$ denote its induced law over finite marked sequences on $[0, T]$ from empty prehistory (Daley & Vere-Jones, 2003). Thus $\mathbb{P}_W = \mathbb{P}_\lambda$ means that the two generators produce the same observable law; it does not require their latent mechanisms to agree.

One point of bookkeeping must be settled first, because it bounds what the proposition can assert. Definition 1 makes a world a pair (S, ϑ) with S ranging over a *finite* set of structures and ϑ over a finite-dimensional parameter space, so its delay laws are drawn from a fixed parametric vocabulary and its pending capacities are bounded. The construction below needs neither of those restrictions: it installs the normalized kernel $\gamma_{\kappa\kappa'}/\Gamma_{\kappa\kappa'}$ as a delay density, which for an arbitrary $\lambda \in \mathcal{A}_+$ need not belong to any fixed family, and it inserts $\mathrm{Poisson}(\Gamma_{\kappa\kappa'})$ opportunities at once, which is unbounded. The same applies to the creation rule: the policy vocabulary of Appendix E.6 exposes a mean per *type*, whereas the construction needs one Poisson mean per *edge*, $\Gamma_{\kappa\kappa'}$, selected by the triggering parent.

We therefore state the result against $\widetilde{\mathcal{W}}$, the class obtained from Definition 1 by three relaxations, with the execution semantics of Appendix G left unchanged: an arbitrary delay density per edge, unbounded pending capacity, and parent-conditioned Poisson creation with one mean per edge. Throughout, the mark vocabulary $\mathcal{V}$ and the dependency graph are fixed and admitted by the finite structure set of Definition 1; the relaxations concern only the three components just named. The containment is thus a statement about the scheduling semantics, not about the finitely parameterized worlds the pretraining prior samples. Restricting each $\gamma_{\kappa\kappa'}$ so that its normalization lies in the delay vocabulary of Appendix E.7 removes the first mismatch, but not the other two: a finite pending capacity assigns zero probability to offspring counts above the cap, so except in degenerate cases such as all-zero branching, capping generally yields an approximation rather than the equality $\mathbb{P}_W = \mathbb{P}_\lambda$.

Proposition 1 (Linear Hawkes is a structural world)**.** *Let $\lambda \in \mathcal{A}_+$ have finite branching masses $\Gamma_{\kappa\kappa'} < \infty$ and a subcritical branching matrix $\rho(\Gamma) < 1$. Let $\mathcal{R} = \{\kappa : \Gamma_{\kappa\kappa'} = 0 \ \forall \kappa' \in \mathcal{V}\}$ and $\mathcal{N} = \mathcal{V} \setminus \mathcal{R}$. Suppose that*

(i) $\mathcal{R} \neq \varnothing$, and immigration is confined to root marks and active there, i.e. $\mu_\kappa > 0$ exactly for $\kappa \in \mathcal{R}$. Positivity on every root is needed because a source in Appendix G always holds one pending opportunity, so a root with $\mu_r = 0$ has no admissible source; a nonempty root set is needed because otherwise the construction has no source at all and the queue never fills, which the infinite-event convention of Appendix G excludes. This rules out, for instance, a purely self-exciting single mark with zero background; and

(ii) the excitation graph $G_\Gamma = (\mathcal{V}, \mathcal{E}_\Gamma)$ with $\mathcal{E}_\Gamma = \{(\kappa', \kappa) \in \mathcal{V} \times \mathcal{N} : \Gamma_{\kappa\kappa'} > 0\}$ is acyclic once its self-loops are removed.

Then there exists $W \in \widetilde{\mathcal{W}}$, with randomized opportunity policies, such that $\mathbb{P}_W = \mathbb{P}_\lambda$.

Proof. We construct W componentwise and then apply the cluster representation.

Graph. Set $(\kappa', \kappa) \in \mathcal{E}$ if and only if $\Gamma_{\kappa\kappa'} > 0$. Edges are defined by positive branching mass rather than by $\gamma_{\kappa\kappa'} \not\equiv 0$, excluding kernels nonzero only on a Lebesgue-null set. Such a kernel has $\Gamma_{\kappa\kappa'} = 0$, contributes nothing to λ, and must not induce an edge because the normalization below would otherwise divide by zero. With this convention $\mathcal{P}_\kappa = \emptyset$ exactly when κ receives no excitation, so the graph-theoretic root set in Appendix G coincides with the $\mathcal{R}$ in the proposition.

Sources. For each $r \in \mathcal{R}$, let B_r be a homogeneous Poisson process of rate $\mu_r > 0$, which is a stationary renewal source in the sense of Appendix G and so always holds one pending opportunity. Pending capacities are unbounded, which is the unbounded-capacity relaxation defining $\widetilde{\mathcal{W}}$, needed because the offspring counts below are unbounded. The per-edge Poisson means used below likewise invoke the parent-conditioned-creation relaxation.

State encoders. For each κ, let $S_\kappa \equiv e_\kappa$ be a constant map to a fixed vector distinct for each mark. These encoders carry no history but remain mutually distinguishable, allowing the delay law below to depend on the identity of the triggering parent.

Policies and delays. For $\kappa \in \mathcal{N}$ and each parent $\kappa' \in \mathcal{P}_\kappa$, let Π_κ respond to a κ'-event at time t by inserting $N_{\kappa\kappa'} \sim \mathrm{Poisson}(\Gamma_{\kappa\kappa'})$ new opportunities of mark κ and deleting none. This policy is a *randomized* queue-update operator: the relation $\mathcal{O}^+ = \Pi_\kappa(\mathcal{O}^-, z_{\kappa\leftarrow\kappa'}(t))$ in Appendix G permits Π_κ to be read as a Markov kernel from $(\mathcal{O}^-, z)$ to queue states, with randomization independent of everything else conditional on z. Each inserted opportunity independently draws its delay from the normalized kernel $f_{\kappa\kappa'}(u) = \gamma_{\kappa\kappa'}(u)/\Gamma_{\kappa\kappa'}$, which is well defined because $\Gamma_{\kappa\kappa'} > 0$ on every edge. This invokes the arbitrary-delay-density relaxation defining $\widetilde{\mathcal{W}}$: for arbitrary $\gamma_{\kappa\kappa'}$ the density $f_{\kappa\kappa'}$ need not lie in the delay vocabulary of Appendix E.7. The latent response $z_{\kappa\leftarrow\kappa'}(t) = \phi_\kappa(s_\kappa(t), \mathbf{s}_{\mathcal{P}_\kappa}(t), s_{\kappa'}(t))$ identifies κ' through its third argument because the constant encoders are distinct. Consequently Π_κ and F_κ may depend on κ' as required.

Conclusion. The resulting events are the superposition of independent immigrant clusters. An event of mark κ' at time t produces $\mathrm{Poisson}(\Gamma_{\kappa\kappa'})$ offspring of mark κ at displacements drawn i.i.d. from $f_{\kappa\kappa'}$, recursively and independently across parents. By the cluster (immigration–birth) representation of multivariate linear Hawkes processes (Adamopoulos, 1975), this superposition has law $\mathbb{P}_\lambda$. The condition $\rho(\Gamma) < 1$ makes every cluster a.s. finite with finite mean size, so the superposition is well defined. Hence $\mathbb{P}_W = \mathbb{P}_\lambda$. □

Corollary 1 (Containment of laws)**.** *Let $\mathcal{A}_+^{\mathrm{root}}$ denote the subfamily of $\mathcal{A}_+$ satisfying the hypotheses of Proposition 1. Then*

$$\{\mathbb{P}_\lambda : \lambda \in \mathcal{A}_+^{\mathrm{root}}\} \subseteq \{\mathbb{P}_W : W \in \widetilde{\mathcal{W}}\}. \tag{13}$$

We do not determine whether the inclusion in equation 13 is strict, and we do not pursue a separation from the full family $\mathcal{A}_+$, nor from $\mathcal{A}_\pm$.

Remark 1 (On the immigration and acyclicity restrictions). Hypothesis (i) follows from the rule for structural worlds in Appendix G that source processes attach only to parentless types: the tuple contains $\{B_r\}_{r\in\mathcal{R}}$, so a mark with parents cannot also carry a background, whereas a general member of $\mathcal{A}_+$ may have $\mu_\kappa > 0$ for every mark. Hypothesis (ii) matches the same specification, which requires dependency graphs to be acyclic once self-loops are removed; the cluster representation itself needs only $\rho(\Gamma) < 1$, which gives almost surely finite clusters of finite mean size whether or not the excitation graph has cycles. Removing either restriction is therefore not a reparameterization but an extension of Definition 1: one would have to admit source processes on non-root marks and cyclic dependency graphs. We leave that to future work. The Poisson cluster construction relies on non-negative kernels throughout and does not extend directly to inhibitory kernels.

B EXPERIMENTAL DETAILS

This appendix provides the complete task definitions, metrics, dataset descriptions, baseline adaptations, experimental limitations, and ablation specifications for the experiments summarized in Section 4.

B.1 DATASETS

We evaluate EventFM on two groups of datasets: standard real-world temporal point process benchmarks; and controlled synthetic processes.

Real-world benchmarks. We use five event-sequence datasets in the standardized form provided by the EasyTPP benchmark (Xue et al., 2024), which fixes preprocessing and train/validation/test splits for cross-model comparability:

- **Retweet**: retweet cascades, with marks indicating small, medium, or large-influence users ($K = 3$).
- **Taxi**: New York City taxi trips, with marks given by the Cartesian product of pickup/drop-off borough and action ($K = 10$).
- **Taobao**: user click behavior on an e-commerce platform, with marks given by item category ($K = 17$).
- **StackOverflow**: user badge awards over a two-year period, with marks given by badge type ($K = 22$).
- **Amazon**: user product-review sequences, with marks given by product category ($K = 16$).

These span $K = 3$ (Retweet) through $K = 22$ (StackOverflow) and cover social, mobility, e-commerce, and online-community domains.

Synthetic processes. To evaluate EventFM under known ground-truth generative processes, we construct a suite of structured synthetic scenarios. Each scenario isolates a single event dynamic that is easily identifiable to a human, so that success or failure on it can be attributed to a specific in-context capability. The scenarios are grouped by the nature of their dynamics.

Deterministic scenarios. In these processes the target pattern has no intrinsic randomness, so a model that identifies it can predict the next event exactly.

- **Event Cycle (Markov)**: each sequence samples a random cyclic order over four event types, a random initial type, and one fixed outgoing delay per type from $[0, 5)$. After these parameters are sampled, both the next type and its time are deterministic.
- **Overlapping Cycles**: two deterministic three-type cycles with disjoint type sets, giving six event types in total, run concurrently. Each cycle independently samples its order, initial type, and type-specific delays from $[0, 5)$; the second cycle also samples an initial offset from $[0, 5)$. Their events are merged in time, requiring the model to separate the two superimposed cycles.
- **Counting Cycle**: each sequence repeats a two-type cycle containing D consecutive weekday events followed by W consecutive weekend events, where D is sampled uniformly from $\{3, \ldots, 7\}$ and W from $\{1, 2, 3\}$. The starting position is sampled uniformly within the cycle and every inter-arrival time is one. Tests counting over longer horizons.

Stochastic scenarios. These processes are random but governed by a single recognizable mechanism, so the model should recover the correct predictive distribution rather than an exact answer.

- **Event Bursts**: three independent same-type Hawkes components are superimposed. Each type has a Poisson immigrant stream, and every event recursively produces a Poisson number of same-type offspring at exponentially distributed delays. The branching ratio and decay timescale vary by type, with one fast, one medium, and one slow timescale in each sequence. Tests in-context estimation of self-excitation strength and timescale.

Table 2: Real-world datasets used in our experiments. K is the number of event types (marks); sequence and event counts are computed from the EasyTPP test splits hosted on Hugging Face.

Dataset	Domain	K	# Sequences	# Events
Retweet	Social	3	2,000	218,465
Taxi	Mobility	10	400	14,820
Taobao	E-commerce	17	500	28,455
StackOverflow	Q&A badges	22	401	26,518
Amazon	Reviews	16	1,851	84,048

- **Type-Specific Seasonality**: three independent event types follow periodic inhomogeneous Poisson processes with squared raised-cosine intensities. Each type has a small constant floor and a dominant seasonal component, a distinct short, medium, or long period, and a random phase. The streams interleave according to their time-varying intensities, so the model must associate each type with its season and infer which type is likely next.

Mixed scenarios. These combine stochastic triggers with deterministic or structured consequences, mimicking recognizable real-world workflows.

- **Stochastic Request–Response**: requests arrive as a homogeneous Poisson process and independently take one of two equally likely request types. Every request produces exactly one shared response type after a deterministic request-type-specific delay; each sequence randomly assigns one short delay from $[0.5, 1.0)$ and one long delay from $[1.5, 2.5)$ to the two request types. Requests and pending responses may overlap, so the model must infer both the trigger relation and its type-conditioned timing.
- **Shipment Deliveries**: shipment orders have Rayleigh-distributed inter-arrival times and each initiates the ordered chain confirmation, shipped, and delivered. The sequence samples characteristic stage delays relative to its mean order gap, while every stage occurrence receives independent multiplicative jitter from $[0.8, 1.2)$. Multiple shipments can overlap, requiring the model to track several in-flight workflows.

Out-of-distribution design. All seven scenarios use purpose-built generators rather than EventSS samples and are intentionally out of distribution, even when their mechanisms resemble parts of the prior. The distinction is strongest for the deterministic scenarios. EventSS assigns zero probability to fully deterministic worlds, although individual delays can be deterministic. Its Markov worlds sample successors from stochastic transition distributions, whereas Event Cycle follows an exact cycle after its parameters are drawn. EventSS stateful policies condition on at most three recent events per mark, whereas Counting Cycle contains same-type runs of up to seven events. Overlapping Cycles likewise superimposes two exact deterministic cycles, a construction not sampled during training.

Fundamental pattern understanding and composition. The targeted suite is designed to expose whether a frozen model can infer fundamental event patterns directly from context: cyclic order, type-conditioned timing, concurrent periodic streams, run-length counting, self-excitation, periodic intensities, and typed trigger–response delays. The deterministic and stochastic scenarios isolate these mechanisms so that success can be attributed to recognizing a specific local event law. The mixed scenarios then compose stochastic arrivals with structured consequences: Request–Response combines Poisson triggers with type-conditioned delayed responses, while Shipment Deliveries combines Rayleigh arrivals with jittered multi-stage chains and overlapping workflows. Success on both levels tests whether the model can identify unfamiliar event patterns in context and compose multiple patterns into more complex systems that mimic real-world event processes.

B.2 Evaluation Procedure

We evaluate EventFM on two prediction tasks, both conditioned on the observed history $\mathcal{H}_t$ at a forecast origin t: a next-event prediction task, which is scored at every eligible event position; and long-horizon task, which predicts the final eight events of each sequence from a single earlier origin.

B.2.1 NEXT-EVENT PREDICTION

For a sequence of N events, we use every origin $t = \tau_i$ for $i \in \{2, \dots, N-1\}$. Thus, every history contains at least two observed events and every origin has a true next-event target. Given the history $\mathcal{H}_t$, EventFM predicts the next event from the model distribution $h_{\text{EFM}}(\tau, \kappa \mid \mathcal{H}_t)$, consistent with the joint-law factorization in equation 1,

$$\widehat{(\tau, \kappa)}_{n_t+1} \sim h_{\text{EFM}}(\tau, \kappa \mid \mathcal{H}_t). \tag{14}$$

We report point-estimate metrics and distributional metrics, scored against the true next event $(\tau, \kappa)_{n_t+1}$.

Point-estimate metrics. Both time metrics use the marginal predictive median:

- **MAE**: mean absolute error between the true time τ_{n_t+1} and the median of the marginal time distribution.
- **RMSE**: root mean squared error between the true time τ_{n_t+1} and the same median. Although the mean is optimal for squared error, recovering it from finite quantiles requires tail assumptions. We use the median as a simpler approximation.
- **Accuracy**: $\mathbf{1}[\hat{\kappa}_{n_t+1} = \kappa_{n_t+1}]$, the predicted type against κ_{n_t+1}, regardless of time.

Distributional metrics. We evaluate the two marginals of h_{EFM} separately: the *time* marginal density f_{EFM} (ignoring type) and the *type* marginal mass p_{EFM} (ignoring time),

$$f_{\text{EFM}}(\tau \mid \mathcal{H}_t) = \sum_{\kappa} h_{\text{EFM}}(\tau, \kappa \mid \mathcal{H}_t), \qquad p_{\text{EFM}}(\kappa \mid \mathcal{H}_t) = \int_t^{\infty} h_{\text{EFM}}(\tau, \kappa \mid \mathcal{H}_t)\, \mathrm{d}\tau, \tag{15}$$

The corresponding marginal time CDF is

$$F_{\text{EFM}}(z \mid \mathcal{H}_t) = \sum_{k \in \mathcal{K}_{n_t}} p_{\text{EFM}}(k \mid \mathcal{H}_t) F_{\tau \mid k, \mathcal{H}_t}(z). \tag{16}$$

All quantile-based forecasts are evaluated on the common grid $\mathcal{A} = \{0.1, 0.2, \dots, 0.9\}$; point-only baselines are represented by the corresponding point-mass distribution on this grid. We score each marginal against the true event:

- **CRPS**: continuous ranked probability score (Gneiting & Raftery, 2007) of the marginal time CDF F_{EFM} against τ_{n_t+1},

$$\text{CRPS}(F_{\text{EFM}}, \tau_{n_t+1}) = \int_{-\infty}^{\infty} \left(F_{\text{EFM}}(z \mid \mathcal{H}_t) - \mathbf{1}[z \geq \tau_{n_t+1}]\right)^2 \mathrm{d}z. \tag{17}$$

- **Brier**: Brier score of the type marginal p_{EFM} against κ_{n_t+1},

$$\text{Brier} = \sum_{\kappa'=1}^{K} \left(p_{\text{EFM}}(\kappa' \mid \mathcal{H}_t) - \mathbf{1}[\kappa' = \kappa_{n_t+1}]\right)^2. \tag{18}$$

CRPS and the Brier score are proper scoring rules for the time and mark marginals, respectively; MAE, RMSE, and accuracy evaluate metric-specific point summaries.

B.2.2 LONG-HORIZON PREDICTION

For each sequence, the final eight ground-truth events form the forecast target. We set the origin to the event immediately preceding them, $t = \tau_{N-8}$, and condition on $\mathcal{H}_t = \{(\tau_i, \kappa_i)\}_{i=1}^{N-8}$. The true continuation and predicted forecast are

$$S = \{(\tau_i, \kappa_i)\}_{i=N-7}^{N}, \qquad \hat{S} = \{\widehat{(\tau, \kappa)}_j\}_{j=1}^{8}. \tag{19}$$

The forecast is generated *autoregressively*: starting from $\hat{\mathcal{H}} = \mathcal{H}_t$, each model produces its next event, appends that prediction to the running history, and repeats this procedure exactly eight times. Metrics:

- **OTD**: optimal transport distance (Mei et al., 2019) between $\hat{S}$ and the true eight-event continuation S. This edit-style distance aligns same-type events (charging their time difference) and charges fixed costs C_{ins} and C_{del} for each unmatched predicted and true event, respectively, penalizing errors in timing and type jointly.
- **Per-type count RMSE**: RMSE over the per-type event counts in the predicted and true eight-event continuations (Xue et al., 2022; Berghaus et al., 2026), measuring whether the model recovers the correct mark mix.

B.3 Synthetic Evaluation Protocols

B.3.1 In-distribution sequence reconstruction

Sampling protocol. We draw held-out sequences from the same EventSS training mixture used to pretrain the reference model, using structural/Markov modes 0.50/0.50 and the graph, source, state, policy, and eight-family delay priors in Appendix E. Each sequence contains 64 events; we generate 1000 evaluation sequences, re-based so the first event occurs at time zero. Sequences are scored by the identical evaluation pipeline used for the real-data benchmarks. The results are reported in Appendix K.

Reconstruction and scoring. For next-event reconstruction, each valid interior origin yields one held-out target, giving $7{,}750$ next-event predictions across the 1000 sequences; we report type accuracy, time RMSE, CRPS, and mark Brier. For multi-event reconstruction, the model observes the prefix through event $N - 8$ and autoregressively predicts the final eight events, appending each prediction before the next step; we score the rollout by sequence-level OTD and per-type count RMSE. Each entry reports the point estimate and its 95% sequence-bootstrap confidence interval.

B.3.2 Synthetic distribution shifts

In-distribution reference. The reference point is the in-distribution dataset described above, comprising 1000 sequences of 64 events from the EventFM training mixture. Each lever perturbs a single generative factor while all other settings, the frozen model, and the evaluator are held fixed.

Levers and sweep levels.

- *Event-type cardinality.* Increase the number of event types that exist in each world. Because a 64-event sequence realizes only about five distinct types under the training prior, we control cardinality directly by replacing the active-mark-count prior and fixing the world to contain exactly K event types, sweeping $K \in \{7, 8, 10\}$ ($1.3\times$ /$1.6\times$ /$2\times$ the in-distribution realized mark count).
- *Temporal scale.* Multiply all inter-event times by $a \in \{0.01, 0.1, 10, 100\}$.
- *Unseen delay laws.* Force every delay to a family absent from pretraining— Pareto and log-logistic (heavy, polynomial tails) and triangular (bounded)—plus a mixed dataset drawing one family per sequence.
- *Unseen mechanism compositions.* Enrich the creation-policy mixture toward an excitation and a many-to-one regime.

Baselines and metrics. Every dataset comprises 1000 sequences of 64 events. The datasets are held-out from pretraining and generated by an EventSS variant that modifies only the perturbed factor, leaving the trained simulator otherwise unchanged. We evaluate the frozen EventFM against the mean/mode and last zero-shot predictors, measuring type accuracy, time RMSE, CRPS, mark Brier, sequence-level OTD, and per-type count RMSE. The results are reported in Appendix K.

B.3.3 Targeted scenarios

Generators and datasets. Each of the seven targeted scenarios defined above (three deterministic, two stochastic, two mixed) is realized as a test-only benchmark of 512 sequences of 128 events each, exported in the EasyTPP event-sequence format and scored by the identical evaluation pipeline used

for the real-data benchmarks. Scenario parameters—cycle orders, periods, stage delays, and decay timescales—are resampled per sequence, so the governing pattern must be inferred in context rather than memorized.

Forecast origins and rollout. We apply two protocols to the same sequences. For next-event prediction, every interior origin $i \in \{2, \ldots, N-1\}$ yields one teacher-forced target, giving 126 targets per sequence and 64,512 in total per scenario. For long-horizon prediction, the model observes the prefix through event $N-8$ and autoregressively predicts the final eight events, appending each predicted event before the next step; event types are sampled at temperature 1.0, times are taken at the predictive median, and the rollout is scored with an OTD cost of 0.25.

Models and baselines. All methods are evaluated zero-shot with frozen weights. EventFM uses last-5 checkpoint averaging. We compare against the baselines described in §B.4: the mean/mode and last-event heuristics, FIM-PP, EVIL (synthetic variant), and an LLM (Qwen3.5-9B) prompted with the event history.

Metrics and aggregation. We score the targeted scenarios with a subset of the metrics defined for the real datasets in §B.2: type accuracy, time RMSE, Brier score, and CRPS for next-event prediction, and the eight-event autoregressive OTD for long-horizon prediction. Per-scenario results are reported in Table 9, and the main results table (Table 1) summarizes each group (deterministic, stochastic, and mixed) by its type accuracy and CRPS. Type accuracy is averaged unweighted across the scenarios in a group, whereas CRPS is first normalized by each scenario's mean inter-arrival time before averaging, so the group summary is not dominated by scenarios with larger time scales.

B.4 Baselines

We call a method *zero-shot* if, at each forecast origin $t = \tau_i$, it uses only the history through event i to predict event $i+1$. We call a method *few-shot* if, instead, it also receives other sequences from the target dataset as in-context examples, but still does not update its weights. Like EventFM, every baseline is evaluated zero-shot.

To our knowledge, no prior dedicated zero-shot event forecasting model exists. We therefore compare against naive predictors and zero-shot adaptations of foundation models designed for other data formats or inference regimes:

- **Mean/mode**: predicts the mean inter-event time and the most-common event type in $\mathcal{H}_t$.
- **Last**: predicts the last inter-event time and the last event type in $\mathcal{H}_t$.
- **LLM (Qwen3.5-9B)**: prompts a pretrained language model with the observed event history and asks it to emit the next event time and type, with no fine-tuning.
- **FIM-PP (zs)** (Berghaus et al., 2026): adapts a model designed for few-shot foundation inference on temporal point processes. FIM-PP normally receives target-domain training sequences as context. For zero-shot evaluation, we instead provide only the partially observed test sequence $\mathcal{H}_t$ as both query and context.
- **EVIL (zs)** (Berghaus, 2026): adapts an in-context program-synthesis model designed to infer event dynamics from target-domain training sequences. For zero-shot evaluation, we provide only the partially observed test sequence $\mathcal{H}_t$ in place of those training sequences. The variant created from synthetic data is used.

For the long-horizon task, every baseline uses the same recursive procedure: its predicted event is appended to its own history before the next prediction, and this is repeated for eight steps.

B.5 Experimental Limitations

Benchmark coverage. Widely available MTPP datasets are limited, and the five real-world benchmarks used here do not cover the full range of event sequences for which forecasting may be relevant. Their test sequences are also relatively short, averaging approximately 37–109 events per sequence

from the counts in Table 2. Some datasets provide little separation on individual metrics; for example, Taobao CRPS ranges only from 0.042 to 0.067 across the zero-shot methods in Table 1, making small differences difficult to interpret.

Baseline coverage and adaptation. There is also a lack of dedicated zero-shot event-sequence forecasting models in the literature, so we include naive baselines that remain competitive on several datasets and metrics. FIM-PP and EVIL were introduced for few-shot inference, where separate target-domain training sequences provide context. To evaluate them zero-shot, we coarsely adapt each method by using only the observed history of the query sequence as its context. This adaptation performs reasonably well, but differs substantially from their intended use and should not be interpreted as measuring their performance in the native few-shot setting.

Configuration coverage. We do not exhaustively study EventFM and EventSS configurations. The reported analyses vary a limited set of important architecture, training, and prior settings one at a time; they do not cover all settings, interactions, or random initializations. The sensitivity results in Appendix B.6.3 and Table 8 therefore support local robustness around the reported configuration rather than global insensitivity to design choices.

B.6 EventFM and EventSS Ablations

The following sections define and evaluate the EventFM and EventSS ablations.

B.6.1 EventFM Ablations

Table 3 compares the final EventFM reference with three ablations. *No context/position* removes e_i^{context} and e_i^{position} from X_i. *Concatenated feature streams* embeds each feature stream separately and concatenates the resulting embeddings into one event representation before the backbone, replacing feature attention. *No feature attention* preserves the separate stream embeddings but disables their feature-axis interaction.

1. **Context and position provide a useful representation for mark prediction.** Removing them lowers accuracy on Taxi from 0.855 to 0.814 and Taobao from 0.596 to 0.553, while deterministic, stochastic, and mixed synthetic accuracy fall from 0.801, 0.515, and 0.687 to 0.521, 0.443, and 0.530. CRPS changes little on the real datasets and stochastic scenarios, but worsens from 0.145 to 0.246 on deterministic scenarios and from 0.525 to 0.621 on mixed scenarios. Context and position therefore supply sequence-relative cues that are broadly useful for learning type structure and selectively useful for modeling structured timing distributions.
2. **Event time and type must be interpreted jointly.** Disabling feature attention without otherwise combining the streams lowers Taxi accuracy to 0.269 and deterministic and mixed accuracy to 0.291 and 0.293. Taxi Brier score rises from 0.263 to 0.640, and deterministic Brier score from 0.277 to 0.678, even though real-data time RMSE changes much less. Keeping the streams isolated can preserve coarse timing accuracy while losing the dependencies needed to predict and calibrate event types.
3. **Combining the streams is essential, but attention is not the only mechanism.** Concatenation attains mean real accuracy 0.553 versus 0.555 for EventFM and deterministic accuracy 0.802 versus 0.801. This recovery shows that the main requirement is an integrated event representation from which the model can learn time–type dependencies. Feature attention remains preferable because it lowers deterministic Brier score from 0.349 to 0.277 and OTD from 8.289 to 5.902, permits adaptive stream weighting, and keeps the backbone independent of the number and representation size of the input features.

B.6.2 EventSS Component Ablations

Table 4 evaluates replacement and component ablations of the EventSS pretraining prior while holding the architecture, optimization, and evaluation fixed. Each model is trained once from initialization, so we focus on broad changes rather than small differences between individual entries.

Table 3: EventFM ablations on the five EasyTPP datasets and grouped synthetic scenarios, evaluated zero-shot. The EventFM row uses the final reference reported in Table 1; each ablation changes one EventFM component. Synthetic accuracy, Brier score, and per-type count RMSE are averaged within each length-128 scenario group; RMSE, CRPS, and OTD are normalized by scenario mean inter-arrival time before averaging. The last two metric blocks are the eight-event autoregressive rollout. Best results are bold and second-best results are underlined.

	Real					Synthetic		
EventFM variant	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
Type accuracy ↑								
EventFM	0.592	**0.855**	0.596	0.414	0.319	0.801	0.515	**0.687**
No context/position	0.584	0.814	0.553	0.415	0.320	0.521	0.443	0.530
Concatenated feature streams	**0.592**	0.844	**0.597**	0.413	0.320	**0.802**	**0.523**	0.673
No feature attention	0.588	0.269	0.518	**0.426**	**0.329**	0.291	0.384	0.293
Time RMSE ↓								
EventFM	**16.1k**	0.315	0.131	1.136	0.370	**0.284**	1.528	**1.257**
No context/position	16.2k	**0.314**	**0.131**	1.132	0.367	0.453	1.527	1.416
Concatenated feature streams	16.1k	0.316	0.131	**1.127**	**0.365**	0.316	**1.525**	1.308
No feature attention	16.4k	0.317	0.131	1.148	0.370	0.456	1.552	1.440
Brier score ↓								
EventFM	0.510	**0.263**	0.560	0.759	0.840	**0.277**	0.610	**0.438**
No context/position	0.517	0.464	0.606	0.759	0.842	0.656	0.642	0.633
Concatenated feature streams	0.512	0.275	**0.560**	0.760	0.841	0.349	**0.607**	0.478
No feature attention	**0.508**	0.640	0.637	**0.751**	**0.830**	0.678	0.673	0.712
CRPS ↓								
EventFM	2.2k	0.137	0.042	0.556	0.225	**0.145**	0.701	**0.525**
No context/position	2.2k	**0.135**	0.042	0.554	0.224	0.246	0.698	0.621
Concatenated feature streams	**2.2k**	0.136	**0.042**	0.555	0.231	0.166	0.699	0.554
No feature attention	2.3k	0.137	0.042	**0.553**	**0.200**	0.256	**0.695**	0.656
OTD ↓								
EventFM	421.9k	4.259	**1.086**	21.213	11.005	**5.902**	21.602	**16.097**
No context/position	**416.2k**	4.590	1.095	21.095	10.951	13.582	**21.091**	18.370
Concatenated feature streams	427.3k	**4.226**	1.100	21.059	**10.941**	8.289	21.732	16.660
No feature attention	420.8k	4.663	1.109	**21.046**	11.277	14.079	21.340	20.379
Per-type count RMSE ↓								
EventFM	1.742	0.860	1.857	1.544	1.527	**0.488**	2.347	**0.876**
No context/position	**1.620**	1.446	1.854	**1.533**	**1.509**	1.002	2.161	1.284
Concatenated feature streams	1.803	**0.832**	**1.840**	1.535	1.517	0.627	2.386	1.057
No feature attention	1.626	1.493	1.866	1.544	1.562	1.055	**2.138**	1.317

1. **EventSS provides broad structural coverage.** The w/o EventSS control retains the complete EventFM architecture but replaces EventSS with the additive intensity-kernel prior used by FIM-PP (Berghaus et al., 2026). Relative to this control, EventSS raises Taxi type accuracy from 0.182 to 0.855, deterministic synthetic accuracy from 0.286 to 0.801, and mixed synthetic accuracy from 0.283 to 0.687. It also lowers normalized CRPS from 0.361 to 0.145 on deterministic scenarios and from 0.672 to 0.525 on mixed scenarios. The control performs better on several StackOverflow next-event metrics, suggesting that its narrower prior can be better suited to a subset of event processes. Its sharp losses on Taxi and the structured synthetic scenarios nevertheless show that it does not transfer as broadly as EventSS. These results support using a broad prior with instance-level structure rather than relying only on additive type-intensity interactions.
2. **The complete EventSS prior benefits from complementary structural and Markov pathways.** Its mean real type accuracy is 0.555, compared with 0.530 for structural-only pretraining and 0.537 for Markov-only pretraining. The same pattern holds on deterministic synthetic scenarios (0.801 versus 0.763 and 0.730) and mixed scenarios (0.687 versus 0.632 and 0.650). Structural-only pretraining particularly degrades Taxi and Taobao accuracy and timing on Markov-style cycles, whereas Markov-only pretraining produces broader losses in type prediction on overlapping cycles and shipment workflows. These mechanism-aligned failures support retaining both pathways to cover qualitatively different event processes.
3. **Multi-event state memory helps recover structured dependencies.** Restricting each state to one event leaves the real-data averages nearly unchanged but lowers deterministic synthetic accuracy from 0.801 to 0.766 and raises deterministic OTD from 5.902 to 6.784. The largest losses occur on overlapping and counting cycles, where forecasting requires distinguishing multiple active or recurring event relationships rather than reacting only to the latest event.
4. **The remaining ablations have mixed effects, consistent with a robust prior.** Removing heavy-tailed delays or nonstationary root processes improves some entries while worsening others, providing limited evidence for either component on the evaluated tasks. We nevertheless retain both because the goal of EventSS is to define a broad prior over possible event processes, and these mechanisms introduce temporal behaviors that may be beneficial in settings not represented by the current benchmarks. Their mixed effects indicate robustness and redundancy in the prior rather than dependence on either component.

B.6.3 EventSS Hyperparameter Sensitivity Analysis

Tables 5 and 6 perturb the reference EventSS prior described in Appendix E. The three ablated levers are as follows. *Heavy-tail weight* shifts probability mass between the two heavy-tailed delay families (log-normal and inverse-Gaussian) and their light-tailed counterweights (gamma and Rayleigh), moving the heavy share of all sampled delays from its default 25% to either 18% or 32%, a relative change of $\pm 28\%$, with all shape parameters held fixed. *Delay dispersion* instead holds the family weights fixed and moves three shape bounds that set how variable the most dispersed sampled world can be: the gamma shape floor ($\alpha_{\min}$ $1.2 \rightarrow 1.6$ or 0.9, i.e. a maximum coefficient of variation of 0.79 or 1.05 against 0.91 by default), the Weibull shape floor ($\kappa_{\min}$ $0.5 \rightarrow 0.7$ or 0.4, CV 1.46 or 3.14 against 2.24), and the log-normal scale ceiling ($s_{\max}$ $1.3 \rightarrow 1.0$ or 1.6, CV 1.31 or 3.46 against 2.10). *Markov world proportion* moves the fraction of sampled worlds executed through the Markov transition pathway rather than the structural pathway from 50% to 40% or 60%, a ± 10 percentage-point change.

Across both tables, the one-at-a-time changes to heavy-tail weight, delay dispersion, and the Markov-world fraction produce broadly similar performance across real and synthetic datasets, including point-prediction, distributional, and rollout metrics. The changes do not cause a consistent or large degradation across tasks, indicating that the EventSS prior is stable throughout the tested neighborhood rather than depending on finely tuned hyperparameters. The clearest effect is a tangible but small trade-off in the Markov-world fraction: increasing it from 50% to 60% modestly improves several real-data timing metrics while reducing accuracy on the deterministic and mixed synthetic groups, whereas decreasing it to 40% generally shifts this balance in the opposite direction. This localized trade-off does not alter the broader stability conclusion.

Table 4: EventSS component ablations on the five EasyTPP datasets and grouped synthetic scenarios, evaluated zero-shot. The w/o EventSS control replaces EventSS with an additive intensity-kernel prior; the remaining ablations remove one EventSS component. All models are retrained from initialization with non-prior settings fixed. Synthetic accuracy and Brier score are averaged within each length-128 scenario group; RMSE, CRPS, and OTD are normalized by scenario mean inter-arrival time before averaging. Best results are bold and second-best results are underlined.

	Real					Synthetic		
EventSS prior	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				Type accuracy ↑				
EventFM	0.592	0.855	0.596	0.414	0.319	0.801	0.515	0.687
w/o EventSS	**0.594**	0.182	**0.599**	**0.421**	**0.329**	0.286	0.510	0.283
Structural only	0.593	0.759	0.555	0.421	0.323	0.763	0.502	0.632
Markov only	0.583	0.845	0.586	0.381	0.290	0.730	0.511	0.650
No heavy tails	0.591	0.861	0.597	0.413	0.321	0.810	0.518	**0.694**
Stationary roots	0.584	0.859	0.596	0.413	0.318	**0.823**	**0.520**	0.685
1-event history	0.591	**0.862**	0.594	0.411	0.320	0.766	0.514	0.676
				Time RMSE ↓				
EventFM	**16.1k**	0.315	0.131	1.136	0.370	0.284	1.528	1.257
w/o EventSS	16.3k	0.312	**0.127**	**1.046**	0.374	0.566	1.515	1.430
Structural only	16.2k	0.407	0.148	1.773	0.493	0.622	1.668	1.348
Markov only	16.6k	**0.310**	0.129	1.081	**0.359**	0.275	**1.514**	1.269
No heavy tails	16.2k	0.315	0.130	1.126	0.372	0.267	1.531	**1.254**
Stationary roots	16.7k	0.315	0.130	1.124	0.364	**0.266**	1.529	1.261
1-event history	16.2k	0.315	0.131	1.139	0.365	0.294	1.534	1.266
				Brier score ↓				
EventFM	**0.510**	0.263	**0.560**	0.759	0.840	0.277	0.610	0.438
w/o EventSS	0.513	0.644	0.579	**0.755**	**0.831**	0.679	**0.605**	0.720
Structural only	0.510	0.424	0.618	0.762	0.858	0.328	0.622	0.504
Markov only	0.514	**0.241**	0.568	0.772	0.850	0.370	0.613	0.471
No heavy tails	0.511	0.266	0.562	0.759	0.841	0.262	0.607	0.436
Stationary roots	0.516	0.259	0.565	0.760	0.842	**0.248**	0.609	**0.433**
1-event history	0.510	0.259	0.564	0.759	0.842	0.307	0.611	0.449
				CRPS ↓				
EventFM	**2.2k**	0.137	0.042	0.556	0.225	0.145	0.701	**0.525**
w/o EventSS	2.3k	**0.127**	**0.041**	**0.516**	0.248	0.361	**0.679**	0.672
Structural only	2.2k	0.179	0.045	0.655	0.293	0.317	0.727	0.617
Markov only	2.5k	0.127	0.042	0.535	**0.215**	0.142	0.703	0.528
No heavy tails	2.2k	0.138	0.042	0.553	0.228	**0.135**	0.697	0.525
Stationary roots	2.5k	0.135	0.042	0.553	0.228	0.136	0.702	0.528
1-event history	2.2k	0.136	0.042	0.556	0.224	0.151	0.698	0.533
				OTD ↓				
EventFM	421.9k	4.259	**1.086**	21.213	11.005	5.902	21.602	**16.097**
w/o EventSS	433.4k	4.793	1.125	21.888	11.702	15.507	21.490	19.851
Structural only	422.6k	4.320	1.109	**20.879**	**10.768**	6.809	21.607	17.731
Markov only	437.3k	4.323	1.104	21.664	11.462	7.141	**21.325**	16.283
No heavy tails	423.2k	4.228	1.100	21.140	11.053	5.596	21.866	16.162
Stationary roots	443.9k	**4.202**	1.098	21.000	10.988	**5.246**	21.692	16.142
1-event history	**421.0k**	4.291	1.092	21.111	10.873	6.784	21.607	16.195

Table 5: Sensitivity of EventFM to the EventSS prior on five EasyTPP datasets and grouped synthetic scenarios. The variants perturb one prior lever at a time around the EventFM prior. Synthetic accuracy and Brier score are averaged within each length-128 scenario group; MAE, RMSE, and CRPS are normalized by scenario mean inter-arrival time before averaging. Best results are bold and second-best results are underlined.

	Real					Synthetic		
EventSS prior variant	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
Type accuracy ↑								
EventFM	0.592	0.855	**0.596**	0.414	0.319	0.801	0.515	0.687
Fewer heavy-tailed delays	0.592	0.856	0.596	0.412	0.320	0.809	0.512	**0.694**
More heavy-tailed delays	0.592	0.858	0.595	0.413	**0.325**	0.797	0.514	0.691
Tighter delay dispersion	0.590	0.851	0.596	0.411	0.322	0.806	0.513	0.694
Looser delay dispersion	0.592	**0.859**	0.596	0.415	0.317	0.806	**0.515**	0.694
Fewer Markov worlds	**0.592**	0.846	0.593	**0.416**	0.321	**0.815**	0.514	0.692
More Markov worlds	0.592	0.856	0.588	0.410	0.315	0.754	0.514	0.651
Time MAE ↓								
EventFM	2.4k	0.160	0.047	0.677	0.285	0.175	0.842	**0.641**
Fewer heavy-tailed delays	2.4k	0.159	0.047	0.674	**0.285**	**0.172**	0.837	0.645
More heavy-tailed delays	2.4k	0.159	0.047	0.676	0.287	0.176	0.841	0.650
Tighter delay dispersion	2.4k	0.159	0.047	0.676	0.290	0.175	0.841	0.648
Looser delay dispersion	2.4k	0.159	**0.046**	0.675	0.290	0.177	**0.837**	0.649
Fewer Markov worlds	**2.4k**	0.163	0.047	0.683	0.302	0.183	0.841	0.653
More Markov worlds	2.4k	**0.157**	0.046	**0.671**	0.292	0.176	0.840	0.657
Time RMSE ↓								
EventFM	**16.1k**	0.315	0.131	1.136	0.370	0.284	1.528	**1.257**
Fewer heavy-tailed delays	16.2k	0.315	0.130	1.124	**0.365**	**0.276**	1.530	1.261
More heavy-tailed delays	16.2k	0.315	0.131	1.132	0.374	0.286	1.528	1.269
Tighter delay dispersion	16.2k	0.315	0.130	1.128	0.366	0.279	1.527	1.261
Looser delay dispersion	16.2k	0.316	0.131	1.132	0.374	0.289	1.534	1.266
Fewer Markov worlds	16.1k	0.321	0.132	1.173	0.376	0.305	1.540	1.259
More Markov worlds	16.2k	**0.313**	**0.130**	**1.105**	0.371	0.278	**1.526**	1.274
CRPS ↓								
EventFM	2.2k	0.137	0.042	0.556	0.225	0.145	0.701	**0.525**
Fewer heavy-tailed delays	2.2k	0.136	0.042	0.552	**0.223**	**0.142**	**0.697**	0.530
More heavy-tailed delays	2.2k	0.136	0.042	0.555	0.224	0.146	0.699	0.535
Tighter delay dispersion	2.2k	0.136	0.042	0.554	0.225	0.144	0.699	0.533
Looser delay dispersion	2.2k	0.137	0.041	0.556	0.226	0.146	0.698	0.533
Fewer Markov worlds	**2.2k**	0.141	0.042	0.566	0.235	0.155	0.702	0.541
More Markov worlds	2.2k	**0.133**	**0.041**	**0.547**	0.224	0.143	0.697	0.539
Brier score ↓								
EventFM	0.510	0.263	**0.560**	0.759	0.840	0.277	0.610	**0.438**
Fewer heavy-tailed delays	0.510	0.262	0.561	0.758	0.840	0.259	0.611	0.438
More heavy-tailed delays	**0.509**	0.251	0.562	0.759	**0.838**	0.275	0.609	0.442
Tighter delay dispersion	0.511	0.256	0.562	0.758	0.840	0.268	0.611	0.442
Looser delay dispersion	0.510	0.259	0.560	0.758	0.842	0.268	**0.608**	0.438
Fewer Markov worlds	0.509	0.250	0.564	**0.758**	0.841	**0.254**	0.610	0.443
More Markov worlds	0.510	**0.248**	0.567	0.760	0.844	0.326	0.611	0.466

Table 6: Eight-event rollout sensitivity to the EventSS prior on the same real datasets and grouped synthetic scenarios as Table 5. Synthetic OTD is normalized by scenario mean inter-arrival time before averaging; per-type count RMSE is averaged unweighted. Best results are bold and second-best results are underlined.

	Real					Synthetic		
EventSS prior variant	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				OTD ↓				
EventFM	421.9k	4.259	**1.086**	21.213	11.005	5.902	21.602	16.097
Fewer heavy-tailed delays	418.7k	4.253	1.097	21.165	11.036	**5.487**	21.641	**16.036**
More heavy-tailed delays	420.5k	4.240	1.092	20.946	11.011	6.327	21.625	16.322
Tighter delay dispersion	421.6k	4.242	1.097	20.954	10.950	6.002	21.581	16.215
Looser delay dispersion	425.4k	4.234	1.095	20.950	11.013	5.660	21.676	16.036
Fewer Markov worlds	420.6k	**4.205**	1.087	**20.942**	**10.924**	5.933	21.635	16.171
More Markov worlds	**418.3k**	4.271	1.107	21.085	11.041	6.815	**21.564**	16.361
				Per-type count RMSE ↓				
EventFM	1.742	0.860	**1.857**	1.544	1.527	0.488	2.347	**0.876**
Fewer heavy-tailed delays	1.686	0.873	1.914	1.512	1.530	**0.461**	2.327	0.889
More heavy-tailed delays	1.690	0.815	1.908	1.519	1.532	0.571	2.325	0.951
Tighter delay dispersion	1.709	0.860	1.917	1.515	1.524	0.531	**2.322**	0.952
Looser delay dispersion	1.764	0.868	1.870	**1.512**	1.531	0.481	2.335	0.885
Fewer Markov worlds	1.702	**0.787**	1.884	1.530	**1.519**	0.473	2.333	0.940
More Markov worlds	**1.685**	0.855	1.943	1.526	1.528	0.547	2.341	0.962

C EventFM Training Details

Training-setting selection. Training uses batches generated on the fly by EventSS, so the model receives newly sampled synthetic sequences throughout optimization. Held-out EventSS data were the only validation data used to select training settings; no real-world or targeted synthetic benchmark results were consulted. The learning rate, mark-loss weight, schedule, and related optimization settings were chosen to avoid sharp spikes in training and validation loss. We reduced the number of backbone layers and the embedding size to make training and inference as computationally efficient as possible, stopping when further reductions caused clear degradation on EventSS validation. Table 7 reports the selected configuration. To check that these choices are not narrowly tuned, Table 8 varies learning rate, batch size, training duration, embedding size, depth, and mark-loss weight one at a time.

Table 7: Architecture and training configuration for EventFM.

Setting	**Value**
Training steps	200,000
Batch size	128
Accelerator	1 GPU
Optimizer	AdamW
Peak learning rate	5×10^{-5}
Learning-rate schedule	10,000-step linear warmup, then cosine decay
Minimum learning rate	5×10^{-6}
Time-loss weight	1.0
Mark-loss weight	0.1
Embedding size	96
Backbone layers	6
Attention heads	3
Training precision	32-bit floating point
Gradient clipping	0.1

Training-setting sensitivity. Most perturbations remain close to the selected configuration across the real and synthetic evaluations. Increasing the learning rate, batch size, training duration, embedding size, or depth does not yield a consistent improvement across metrics or datasets. The smaller 64-dimensional and three-layer models remain competitive on the real datasets but lose more accuracy and OTD performance on structured synthetic scenarios, supporting the selected model as a compact efficiency–performance trade-off. The main exception is the 10^{-5} learning rate, which substantially degrades type prediction, especially on Taxi and the deterministic and mixed synthetic groups, consistent with under-training rather than sensitivity around the selected setting. Overall, the broad stability of the remaining variants indicates that the reported configuration is not overtuned.

Table 8: Training-setting sensitivity on five EasyTPP datasets and grouped synthetic scenarios. Each variant changes one setting relative to the chosen EventFM configuration. Synthetic accuracy and Brier score are averaged within each length-128 scenario group; RMSE, CRPS, and OTD are normalized by scenario mean inter-arrival time before averaging. OTD uses the eight-event autoregressive rollout. Best results are bold.

	Real					Synthetic		
Setting	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				Type accuracy ↑				
EventFM (chosen)	0.592	0.855	0.596	0.414	0.319	0.801	0.515	0.687
LR 1×10^{-5}	**0.594**	0.372	0.527	**0.422**	**0.332**	0.350	0.383	0.339
LR 1×10^{-4}	0.594	0.861	0.592	0.414	0.317	0.806	0.515	0.675
Batch size 64	0.590	0.854	0.598	0.415	0.323	**0.822**	0.514	0.691
Batch size 256	0.593	**0.867**	0.591	0.413	0.319	0.809	0.510	0.690
100k steps	0.588	0.845	0.597	0.413	0.323	0.788	0.515	0.678
400k steps	0.594	0.854	0.590	0.414	0.319	0.808	0.512	0.688
Embedding size 64	0.589	0.850	0.595	0.413	0.326	0.788	0.511	0.682
Embedding size 192	0.594	0.850	0.589	0.416	0.320	0.818	0.513	**0.703**
3 layers	0.585	0.847	**0.600**	0.412	0.320	0.775	**0.524**	0.679
12 layers	0.594	0.862	0.589	0.413	0.315	0.812	0.509	0.676
Mark weight 0.05	0.590	0.854	0.597	0.415	0.323	0.781	0.522	0.685
Mark weight 0.2	0.592	0.854	0.590	0.414	0.315	0.766	0.511	0.659
				Time RMSE ↓				
EventFM (chosen)	16.1k	0.315	0.131	1.136	0.370	0.284	1.528	1.257
LR 1×10^{-5}	16.3k	0.318	0.131	1.152	0.368	0.474	1.532	1.448
LR 1×10^{-4}	16.1k	0.315	0.130	1.131	0.371	0.279	1.530	1.263
Batch size 64	16.2k	0.317	0.130	**1.125**	**0.367**	0.300	1.529	1.274
Batch size 256	16.1k	0.315	0.131	1.133	0.371	0.273	1.536	1.259
100k steps	16.2k	0.320	0.131	1.172	0.371	0.324	1.535	1.279
400k steps	16.2k	**0.315**	**0.130**	1.127	0.368	0.266	1.526	1.267
Embedding size 64	16.2k	0.320	0.132	1.163	0.368	0.311	1.532	1.291
Embedding size 192	16.1k	0.315	0.130	1.134	0.376	**0.266**	**1.521**	**1.246**
3 layers	16.1k	0.316	0.131	1.148	0.367	0.313	1.530	1.301
12 layers	16.2k	0.317	0.131	1.140	0.372	0.273	1.524	1.257
Mark weight 0.05	**16.1k**	0.317	0.131	1.139	0.367	0.297	1.531	1.270
Mark weight 0.2	16.2k	0.317	0.131	1.139	0.372	0.294	1.534	1.291
				CRPS ↓				
EventFM (chosen)	2.2k	0.137	0.042	0.556	0.225	0.145	0.701	0.525
LR 1×10^{-5}	2.2k	0.136	0.042	**0.550**	0.222	0.265	0.704	0.650
LR 1×10^{-4}	2.2k	0.137	0.042	0.557	0.224	0.142	0.700	0.519
Batch size 64	2.2k	0.137	**0.042**	0.553	**0.220**	0.155	**0.697**	0.540
Batch size 256	2.2k	0.136	0.042	0.555	0.230	0.137	0.698	0.520
100k steps	2.2k	0.137	0.042	0.554	0.223	0.163	0.700	0.553
400k steps	2.2k	**0.135**	0.042	0.554	0.224	0.136	0.700	0.521
Embedding size 64	2.2k	0.137	0.042	0.555	0.223	0.159	0.698	0.550
Embedding size 192	2.2k	0.137	0.042	0.558	0.226	**0.134**	0.703	**0.512**
3 layers	**2.2k**	0.136	0.042	0.555	0.222	0.156	0.698	0.546
12 layers	2.2k	0.136	0.042	0.558	0.229	0.137	0.703	0.523
Mark weight 0.05	2.2k	0.137	0.042	0.557	0.224	0.151	0.699	0.550
Mark weight 0.2	2.2k	0.136	0.042	0.555	0.229	0.148	0.697	0.529
				Brier score ↓				
EventFM (chosen)	0.510	0.263	0.560	0.759	0.840	0.277	0.610	0.438
LR 1×10^{-5}	**0.506**	0.630	0.628	**0.752**	**0.830**	0.681	0.676	0.711
LR 1×10^{-4}	0.508	0.247	0.568	0.757	0.843	0.268	0.609	0.440
Batch size 64	0.512	0.255	0.560	0.758	0.839	0.272	0.609	0.453
Batch size 256	0.509	0.247	0.566	0.759	0.841	0.258	0.611	0.427
100k steps	0.515	0.265	0.560	0.760	0.839	0.327	0.609	0.473
400k steps	0.509	0.246	0.567	0.758	0.844	0.261	0.611	0.435
Embedding size 64	0.512	0.261	0.562	0.758	0.837	0.317	0.612	0.466
Embedding size 192	0.508	0.247	0.572	0.756	0.842	**0.249**	0.612	**0.409**
3 layers	0.515	0.272	**0.559**	0.759	0.840	0.321	0.607	0.457
12 layers	0.508	0.245	0.569	0.757	0.844	0.265	0.613	0.441
Mark weight 0.05	0.512	0.261	0.560	0.758	0.837	0.322	**0.605**	0.471
Mark weight 0.2	0.510	**0.243**	0.567	0.757	0.845	0.309	0.612	0.458
				OTD ↓				
EventFM (chosen)	421.9k	4.259	**1.086**	21.213	11.005	5.902	21.602	16.097
LR 1×10^{-5}	418.4k	4.701	1.095	21.169	11.088	14.102	**20.967**	18.984
LR 1×10^{-4}	420.6k	4.304	1.108	21.090	10.933	6.276	21.664	16.173
Batch size 64	421.4k	4.239	1.102	20.910	10.856	6.447	21.684	16.387
Batch size 256	421.2k	4.244	1.095	21.135	10.963	5.579	21.700	**16.035**
100k steps	427.8k	4.215	1.107	20.999	11.063	7.997	21.686	16.597
400k steps	421.5k	4.322	1.092	20.981	**10.830**	5.652	21.646	16.147
Embedding size 64	421.3k	4.222	1.099	20.893	11.014	7.468	21.555	16.341
Embedding size 192	418.8k	4.365	1.101	21.068	10.881	5.497	21.298	16.043
3 layers	420.4k	**4.204**	1.112	21.085	11.148	7.224	21.744	16.611
12 layers	419.9k	4.259	1.100	**20.836**	10.861	**5.403**	21.445	16.112
Mark weight 0.05	**417.3k**	4.262	1.091	20.915	10.926	6.828	21.649	16.438
Mark weight 0.2	419.1k	4.209	1.106	21.099	11.016	6.861	21.561	16.484

Loss and predictive distribution. The time loss is evaluated on next-event delays normalized by the context mean inter-event time, using the same scale as the normalized time quantiles in equation 3. The timing and mark losses are combined using the weights in equation 9; examples whose target mark has not appeared in the context do not contribute to either term. Candidate-conditional CDFs are formed by linear interpolation between quantile knots, with zero delay as the lower endpoint and a finite extrapolated upper endpoint. Their type-probability-weighted mixture is inverted to obtain the marginal quantiles used for prediction.

D Extended Results

D.1 Targeted Scenario Metrics

Each targeted scenario is constructed as described in §B.1 ("Synthetic processes") and evaluated with the protocol in §B.3 ("Targeted scenarios"). Table 9 reports per-scenario type accuracy, time RMSE, Brier score, CRPS, and eight-event OTD. The grouped synthetic columns of Table 1 instead summarize type accuracy and CRPS.

Because the scenarios' mean inter-arrival times span more than a factor of five, an unweighted average of raw CRPS values would be dominated by scenarios with larger time scales. Table 1 therefore normalizes each scenario's CRPS by its mean inter-arrival time before averaging, whereas type accuracy is averaged unweighted. The per-scenario RMSE and CRPS values here remain in raw units so their absolute magnitudes are interpretable.

D.2 Additional Benchmark Metrics

Tables 10, 11, and 12 report all evaluation metrics with sequence-bootstrap confidence intervals for the five EasyTPP datasets and grouped synthetic scenarios. They include the zero-shot baseline block and EventFM from Table 1; ablation results are reported separately in the preceding ablation sections. In accordance with Appendix B.4, the EVIL row uses the synthetic-data variant.

Table 9: Results on the targeted length-128 synthetic datasets. Columns are Event Cycle, Overlapping Cycles, Counting Cycle, Event Bursts, Type-Specific Seasonality, Stochastic Request–Response, and Shipment Deliveries. Higher is better for accuracy. Lower is better for RMSE, Brier score, CRPS, and OTD. Best results are bold and second-best results are underlined.

	Deterministic			**Stochastic**		**Mixed**	
Model	**Event Cycle**	**Overlap.**	**Counting**	**Bursts**	**Season.**	**Request–Resp.**	**Shipment**
			Type accuracy ↑				
Mean/mode	0.063	0.061	0.706	0.415	0.354	0.479	0.249
Last	0.000	0.000	0.700	**0.634**	**0.471**	0.072	0.038
Qwen3.5 9B	0.909	0.305	0.729	0.550	0.382	0.490	0.605
FIM-PP (zs)	0.247	0.171	0.714	0.349	0.334	0.415	0.269
EVIL (zs)	0.978	0.634	0.755	0.573	0.402	**0.662**	**0.765**
EventFM	**0.979**	**0.648**	**0.775**	0.602	0.428	0.657	0.718
			Time RMSE ↓				
Mean/mode	1.260	0.918	**0.000**	**2.470**	**1.677**	10.840	1.579
Last	2.000	1.485	0.000	3.180	2.257	16.789	2.126
Qwen3.5 9B	0.654	1.115	0.000	2.721	1.901	12.091	1.895
FIM-PP (zs)	1.909	1.120	0.391	2.508	1.763	11.153	1.740
EVIL (zs)	0.860	0.948	0.000	2.544	1.763	10.562	1.439
EventFM	**0.404**	**0.768**	0.044	2.512	1.732	**10.169**	**1.404**
			Brier score ↓				
Mean/mode	0.782	0.859	0.404	0.665	0.687	0.648	0.781
Last	2.000	2.000	0.599	0.732	1.057	1.856	1.923
Qwen3.5 9B	0.182	1.391	0.543	0.900	1.236	1.019	0.790
FIM-PP (zs)	0.973	1.017	0.488	0.774	0.788	0.698	0.856
EVIL (zs)	**0.044**	0.731	0.490	0.854	1.197	0.676	0.470
EventFM	0.045	**0.469**	**0.318**	**0.552**	**0.667**	**0.450**	**0.425**
			CRPS ↓				
Mean/mode	1.042	0.734	**0.000**	1.578	1.130	7.147	1.111
Last	1.623	1.152	0.000	1.724	1.419	10.709	1.446
Qwen3.5 9B	0.172	0.752	0.000	1.322	1.084	5.722	1.054
FIM-PP (zs)	1.379	0.724	0.297	1.137	0.901	5.165	0.949
EVIL (zs)	0.657	0.724	0.000	1.360	1.081	5.535	0.908
EventFM	**0.137**	**0.431**	0.018	**1.103**	**0.839**	**4.081**	**0.624**
			Eight-event autoregressive OTD ↓				
Mean/mode	35.999	22.219	8.418	**31.156**	**25.077**	119.542	26.072
Last	64.019	35.152	13.359	36.665	33.931	180.126	40.658
Qwen3.5 9B	22.873	31.771	9.379	41.729	35.821	181.677	38.642
FIM-PP (zs)	49.787	27.033	9.426	31.948	26.977	124.385	28.813
EVIL (zs)	7.879	13.407	9.289	36.696	31.907	129.720	**19.108**
EventFM	**3.861**	**9.576**	**8.089**	33.097	26.714	**118.873**	20.642

Table 10: Type accuracy and point-time errors on the five EasyTPP datasets and grouped synthetic scenarios. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Synthetic accuracy is averaged unweighted; MAE and RMSE are normalized by scenario mean inter-arrival time before group averaging. Best point estimates are bold and second-best point estimates are underlined.

	Real					Synthetic		
Model	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				Type accuracy ↑				
Mean/mode	<u>0.589</u> [0.584,0.594]	0.191 [0.173,0.209]	0.515 [0.494,0.537]	**0.430** [0.415,0.445]	**0.338** [0.331,0.345]	0.277 [0.271,0.282]	0.384 [0.381,0.388]	0.364 [0.363,0.365]
Last	0.534 [0.530,0.538]	0.000 [0.000,0.000]	**0.602** [0.583,0.621]	0.303 [0.290,0.316]	0.218 [0.212,0.225]	0.233 [0.231,0.236]	**0.553** [0.549,0.556]	0.055 [0.054,0.057]
Qwen3.5 9B	0.582 [0.577,0.587]	0.749 [0.731,0.766]	0.567 [0.548,0.587]	0.401 [0.384,0.417]	0.318 [0.311,0.326]	0.647 [0.643,0.652]	0.466 [0.462,0.470]	0.548 [0.544,0.551]
FIM-PP (zs)	0.521 [0.515,0.528]	0.280 [0.266,0.294]	0.373 [0.351,0.397]	0.330 [0.313,0.345]	0.255 [0.247,0.263]	0.377 [0.374,0.380]	0.342 [0.337,0.346]	0.342 [0.340,0.345]
EVIL (zs)	0.571 [0.566,0.576]	**0.858** [0.848,0.868]	0.564 [0.543,0.584]	0.376 [0.362,0.391]	0.292 [0.286,0.299]	<u>0.789</u> [0.785,0.793]	0.487 [0.484,0.491]	**0.714** [0.710,0.717]
EventFM	**0.592** [0.587,0.596]	<u>0.855</u> [0.844,0.865]	<u>0.596</u> [0.577,0.615]	<u>0.414</u> [0.399,0.429]	<u>0.319</u> [0.312,0.327]	**0.801** [0.797,0.805]	<u>0.515</u> [0.511,0.519]	<u>0.687</u> [0.684,0.691]
				Time MAE ↓				
Mean/mode	2.5k [2.5k,2.6k]	0.174 [0.167,0.182]	0.061 [0.058,0.065]	0.757 [0.726,0.789]	<u>0.282</u> [0.278,0.287]	0.346 [0.340,0.353]	0.975 [0.962,0.989]	0.924 [0.911,0.938]
Last	2.8k [2.7k,2.9k]	0.216 [0.209,0.224]	0.067 [0.064,0.071]	0.896 [0.860,0.933]	0.301 [0.297,0.307]	0.542 [0.531,0.554]	1.136 [1.120,1.152]	1.313 [1.294,1.332]
Qwen3.5 9B	2.7k [2.6k,2.8k]	0.163 [0.159,0.168]	**0.044** [0.042,0.047]	0.717 [0.688,0.747]	0.320 [0.313,0.328]	<u>0.234</u> [0.227,0.241]	<u>0.869</u> [0.858,0.881]	0.795 [0.784,0.806]
FIM-PP (zs)	2.5k [2.5k,2.6k]	0.177 [0.173,0.182]	0.048 [0.045,0.051]	<u>0.698</u> [0.671,0.726]	0.466 [0.462,0.471]	0.532 [0.525,0.540]	0.878 [0.867,0.891]	0.856 [0.845,0.869]
EVIL (zs)	**2.4k** [2.3k,2.4k]	**0.158** [0.154,0.164]	0.049 [0.046,0.051]	0.714 [0.685,0.743]	**0.270** [0.265,0.275]	0.292 [0.285,0.298]	0.881 [0.869,0.893]	<u>0.732</u> [0.722,0.742]
EventFM	<u>2.4k</u> [2.3k,2.5k]	<u>0.160</u> [0.155,0.165]	<u>0.047</u> [0.045,0.050]	**0.677** [0.649,0.705]	0.285 [0.280,0.291]	**0.175** [0.171,0.179]	**0.842** [0.831,0.854]	**0.641** [0.633,0.650]
				Time RMSE ↓				
Mean/mode	16.5k [16.0k,17.1k]	<u>0.314</u> [0.297,0.337]	**0.129** [0.124,0.135]	**1.101** [1.057,1.148]	**0.338** [0.335,0.341]	0.428 [0.420,0.435]	**1.492** [1.472,1.513]	1.367 [1.348,1.388]
Last	16.6k [16.1k,17.3k]	0.403 [0.379,0.432]	0.171 [0.164,0.178]	1.398 [1.346,1.451]	0.460 [0.455,0.464]	0.686 [0.674,0.700]	1.958 [1.931,1.986]	2.011 [1.982,2.043]
Qwen3.5 9B	18.2k [17.2k,19.7k]	0.328 [0.311,0.349]	0.133 [0.127,0.139]	1.220 [1.174,1.269]	0.476 [0.470,0.482]	0.401 [0.391,0.411]	1.664 [1.642,1.687]	1.569 [1.547,1.594]
FIM-PP (zs)	16.4k [15.9k,17.0k]	0.343 [0.327,0.363]	<u>0.130</u> [0.124,0.135]	1.184 [1.142,1.226]	0.540 [0.537,0.543]	0.702 [0.694,0.710]	1.538 [1.518,1.559]	1.445 [1.425,1.467]
EVIL (zs)	**15.8k** [15.3k,16.4k]	**0.314** [0.297,0.335]	0.132 [0.127,0.138]	<u>1.122</u> [1.079,1.167]	0.377 [0.373,0.380]	<u>0.382</u> [0.375,0.390]	1.550 [1.529,1.571]	<u>1.299</u> [1.280,1.319]
EventFM	<u>16.1k</u> [15.6k,16.7k]	0.315 [0.299,0.336]	0.131 [0.125,0.137]	1.136 [1.086,1.196]	<u>0.370</u> [0.366,0.373]	**0.284** [0.279,0.290]	<u>1.528</u> [1.507,1.549]	**1.257** [1.239,1.276]

Table 11: Distributional next-event metrics on the five EasyTPP datasets and grouped synthetic scenarios. Entries report point estimates and sequence-bootstrap 95% confidence intervals; lower is better. Synthetic Brier score is averaged unweighted; CRPS is normalized by scenario mean inter-arrival time before group averaging. Best point estimates are bold and second-best point estimates are underlined.

	Real					Synthetic		
Model	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				Brier score ↓				
Mean/mode	**0.508** [0.505,0.512]	0.640 [0.633,0.648]	0.638 [0.614,0.660]	**0.747** [0.735,0.760]	**0.831** [0.825,0.836]	0.682 [0.679,0.684]	0.676 [0.674,0.677]	0.715 [0.715,0.715]
Last	0.933 [0.925,0.940]	2.000 [2.000,2.000]	0.796 [0.758,0.833]	1.394 [1.369,1.421]	1.563 [1.550,1.576]	1.533 [1.529,1.538]	0.895 [0.888,0.901]	1.890 [1.887,1.893]
Qwen3.5 9B	0.836 [0.826,0.846]	0.502 [0.468,0.538]	0.865 [0.825,0.904]	1.199 [1.166,1.231]	1.363 [1.348,1.377]	0.705 [0.696,0.714]	1.068 [1.061,1.075]	0.905 [0.898,0.911]
FIM-PP (zs)	0.601 [0.596,0.606]	0.832 [0.820,0.844]	0.805 [0.788,0.822]	0.848 [0.837,0.859]	0.923 [0.917,0.929]	0.826 [0.821,0.832]	0.781 [0.777,0.785]	0.777 [0.775,0.779]
EVIL (zs)	0.858 [0.848,0.868]	0.283 [0.264,0.304]	0.872 [0.832,0.913]	1.247 [1.217,1.277]	1.416 [1.402,1.429]	0.422 [0.414,0.430]	1.025 [1.018,1.033]	0.573 [0.566,0.580]
EventFM	0.510 [0.506,0.513]	**0.263** [0.249,0.279]	**0.560** [0.537,0.581]	0.759 [0.747,0.771]	0.840 [0.834,0.845]	**0.277** [0.273,0.282]	**0.610** [0.607,0.612]	**0.438** [0.433,0.442]
				CRPS ↓				
Mean/mode	2.5k [2.5k,2.6k]	0.174 [0.167,0.182]	0.061 [0.058,0.065]	0.757 [0.726,0.789]	0.282 [0.278,0.287]	0.346 [0.340,0.353]	0.975 [0.962,0.989]	0.924 [0.911,0.938]
Last	2.8k [2.7k,2.9k]	0.216 [0.209,0.224]	0.067 [0.064,0.071]	0.896 [0.860,0.933]	0.301 [0.297,0.307]	0.542 [0.531,0.554]	1.136 [1.120,1.152]	1.313 [1.294,1.332]
Qwen3.5 9B	2.7k [2.6k,2.8k]	0.163 [0.159,0.168]	0.044 [0.042,0.047]	0.717 [0.688,0.747]	0.320 [0.313,0.328]	0.234 [0.227,0.241]	0.869 [0.858,0.881]	0.795 [0.784,0.806]
FIM-PP (zs)	2.5k [2.4k,2.5k]	0.166 [0.162,0.171]	**0.042** [0.039,0.044]	0.642 [0.616,0.670]	0.440 [0.436,0.445]	0.488 [0.482,0.495]	0.736 [0.726,0.746]	0.717 [0.707,0.727]
EVIL (zs)	2.4k [2.3k,2.4k]	0.158 [0.154,0.164]	0.049 [0.046,0.051]	0.714 [0.685,0.743]	0.270 [0.265,0.275]	0.292 [0.285,0.298]	0.881 [0.869,0.893]	0.732 [0.722,0.742]
EventFM	**2.2k** [2.2k,2.3k]	**0.137** [0.133,0.142]	0.042 [0.040,0.045]	**0.556** [0.533,0.580]	**0.225** [0.222,0.228]	**0.145** [0.142,0.148]	**0.701** [0.691,0.710]	**0.525** [0.518,0.532]

Table 12: Eight-event rollout metrics on the five EasyTPP datasets and grouped synthetic scenarios. Entries report point estimates and sequence-bootstrap 95% confidence intervals; lower is better. Synthetic OTD is normalized by scenario mean inter-arrival time before group averaging; per-type count RMSE is averaged unweighted. Best point estimates are bold and second-best point estimates are underlined.

	Real					Synthetic		
Model	**Retweet**	**Taxi**	**Taobao**	**StackOv.**	**Amazon**	**Det.**	**Stoch.**	**Mixed**
				OTD ↓				
Mean/mode	<u>428.0k</u> [412.5k,443.5k]	4.543 [4.217,4.929]	**1.080** [0.982,1.190]	**20.371** [19.262,21.545]	<u>10.829</u> [10.643,11.024]	13.899 [13.645,14.154]	**20.308** [19.475,21.166]	17.955 [17.382,18.597]
Last	527.8k [508.3k,547.5k]	6.159 [5.788,6.583]	1.177 [1.063,1.306]	26.421 [25.005,27.894]	13.345 [13.112,13.594]	22.959 [22.565,23.358]	25.577 [24.462,26.700]	27.511 [26.723,28.359]
Qwen3.5 9B	509.7k [491.0k,529.1k]	5.617 [5.262,6.028]	1.188 [1.079,1.316]	26.234 [24.837,27.711]	13.728 [13.497,13.961]	15.122 [14.732,15.510]	28.048 [26.869,29.258]	26.961 [26.149,27.840]
FIM-PP (zs)	475.8k [458.7k,493.0k]	5.567 [5.220,5.971]	1.253 [1.143,1.376]	24.380 [23.028,25.791]	14.026 [13.800,14.259]	17.449 [17.161,17.744]	<u>21.304</u> [20.435,22.233]	19.243 [18.645,19.895]
EVIL (zs)	494.3k [476.2k,512.6k]	**4.180** [3.826,4.598]	1.123 [1.018,1.246]	21.777 [20.523,23.104]	**10.470** [10.283,10.666]	<u>7.919</u> [7.709,8.147]	24.820 [23.656,25.983]	<u>16.429</u> [15.776,17.141]
EventFM	**421.9k** [406.7k,437.5k]	<u>4.259</u> [3.903,4.663]	<u>1.086</u> [0.986,1.201]	<u>21.213</u> [19.982,22.506]	11.005 [10.808,11.208]	**5.902** [5.723,6.087]	21.602 [20.614,22.617]	**16.097** [15.487,16.763]
				Per-type count RMSE ↓				
Mean/mode	**1.653** [1.601,1.704]	1.462 [1.387,1.535]	<u>1.952</u> [1.854,2.060]	**1.479** [1.426,1.536]	<u>1.579</u> [1.551,1.606]	1.069 [1.040,1.098]	**2.138** [2.072,2.206]	1.332 [1.293,1.374]
Last	3.341 [3.260,3.421]	3.874 [3.827,3.924]	2.614 [2.440,2.794]	3.407 [3.310,3.501]	3.354 [3.309,3.399]	3.339 [3.287,3.396]	3.739 [3.636,3.843]	3.517 [3.485,3.551]
Qwen3.5 9B	2.878 [2.807,2.954]	2.918 [2.802,3.027]	2.550 [2.382,2.725]	3.190 [3.106,3.277]	2.938 [2.898,2.980]	1.892 [1.845,1.940]	4.126 [4.025,4.227]	3.127 [3.089,3.162]
FIM-PP (zs)	2.239 [2.190,2.290]	1.753 [1.677,1.833]	2.260 [2.186,2.340]	1.711 [1.655,1.770]	1.739 [1.713,1.767]	1.778 [1.744,1.815]	2.630 [2.547,2.712]	1.816 [1.769,1.862]
EVIL (zs)	2.541 [2.472,2.612]	**0.711** [0.631,0.799]	2.384 [2.218,2.552]	2.248 [2.152,2.346]	1.925 [1.884,1.966]	<u>0.936</u> [0.900,0.976]	3.581 [3.474,3.685]	<u>0.949</u> [0.907,0.992]
EventFM	<u>1.742</u> [1.690,1.795]	<u>0.860</u> [0.783,0.948]	**1.857** [1.749,1.965]	<u>1.544</u> [1.487,1.610]	**1.527** [1.503,1.554]	**0.488** [0.458,0.519]	<u>2.347</u> [2.273,2.426]	**0.876** [0.841,0.912]

E EVENT STRUCTURAL SIMULATOR DETAILS

This appendix provides a detailed description of the EventSS prior from Section 3.2 and specifies the configuration used for the results in Table 1. Following the hierarchy in equation 4, EventSS samples a new event-generating world for each training sequence. Structural and Markov worlds are sampled with equal probability as two parameterizations of the same queue simulator. Structural worlds use a directed dependency graph, while Markov worlds restrict each event to one successor sampled from a transition matrix. The sections below detail the graph, source, state, policy, and delay components.

E.1 DEPENDENCY GRAPH FOR STRUCTURAL WORLDS

Conditional on structural mode, EventSS samples the following:

1. A directed acyclic graph structure with a hyperprior on several different types of DAGs defined in Table 13,
2. An active mark count K, sampled from a geometric distribution with probability 0.55 and uniformly from $\{2, \ldots, K_{\text{req}}\}$ otherwise. The geometric draw is clipped at the batch cap K_{req}. Markov worlds use the same prior.
3. A root count equal to one with probability 0.60, two with probability 0.25, and three to five with total probability 0.15, clipped at K. Edges are removed or added to realize this count.

Together, these choices define a dependency graph G that determines which event types can affect the future generation of other types. The graph is acyclic. Self-triggering is sampled separately by the recurrence policy in Appendix E.6.

Chain motifs. With probability 0.35, EventSS selects a subset of a structural graph and converts it into a chain that follows the graph's topological order. Incoming edges to chain nodes are removed except for the preceding chain edge.

Table 13: Structural graph-family prior.

Family	Probability	Additional range
Pipeline	0.25	Directed chain.
Fan-out	0.15	Out-degree 2–5.
Fan-in	0.15	In-degree 2–5.
Diamond	0.10	2–4 parallel branches.
Layered DAG	0.15	3–6 layers; edge probability $[0.2, 0.6]$.
Cascade tree	0.10	2–4 children per internal node.
Grown DAG	0.10	Incremental random attachment.

E.2 TRANSITION MATRICES FOR MARKOV WORLDS

Conditional on Markov mode, EventSS samples one of the transition structures in Table 14. Each connected component is seeded with one event at time zero.

Table 14: Markov graph-family prior.

Family	Probability
Dense	0.15
Bursty	0.35
Sparse	0.35
Block	0.15

Dense rows place mass over all marks with Dirichlet concentration sampled log-uniformly from $[0.1, 3]$. Sparse rows select one to four destinations and use concentration sampled log-uniformly

from $[0.03, 0.5]$. Bursty worlds designate $\max(1, \lfloor 0.4K \rfloor)$ marks as bursty, sample concentration log-uniformly from $[0.01, 0.5]$, assign bursty self-transition probability uniformly on $[0.4, 0.8]$, and assign other self-transition probabilities uniformly on $[0, 0.15]$. Block worlds create two or three disconnected recurrent components and use dense rows within each component. After each event, the transition row selects one successor and its delay law schedules one opportunity. Markov execution does not evaluate the structural map or opportunity policy.

Cycle motifs. With probability 0.35, EventSS samples an ordered tuple of marks, increases the corresponding cycle-transition probabilities, and proportionally reduces other transition weights.

E.3 Root Source Processes B_r and Markov Initialization

For structural worlds, root nodes follow one of three arrival processes: stationary renewal, periodic Poisson, or decaying Poisson, with probabilities given in Table 15. These processes initialize and subsequently update the opportunity queue. Markov worlds do not use exogenous root processes; instead, each connected component places one initial opportunity at time zero.

Table 15: Root source-process prior.

Root mode	**Probability**
Stationary renewal	0.60
Periodic Poisson	0.20
Decaying Poisson	0.20

Stationary roots follow a renewal process with a delay distribution from Appendix E.7. Periodic and decaying roots follow Poisson processes with initial rate multipliers sampled log-uniformly from $[0.5, 2]$. Periodic amplitudes are uniform on $[0.3, 1.5]$, with periods spanning 4–20 characteristic root intervals. Decaying roots select linear or exponential decay with equal probability, use floors between 0.02 and 0.15 of the initial rate, and sample scaled decay parameters log-uniformly from $[0.005, 0.08]$.

E.4 State Encoder S_κ

Each event type has a state encoder that summarizes its mark-restricted history up to time t for use by the opportunity policy. From $\mathcal{H}_t^\kappa = \{(\tau_i, \kappa_i) \in \mathcal{H}_t : \kappa_i = \kappa\}$, it computes summary statistics $a_\kappa(t) = A_\kappa(\mathcal{H}_t^\kappa)$, including count, span, recency, and rate, over a history capacity sampled uniformly from $\{1, 2, 3\}$. A perceptron then produces $s_\kappa(t) = \boldsymbol{\eta}_\kappa \odot \left[\sigma_\kappa^S(W_\kappa a_\kappa(t)) + \mathbf{b}_\kappa\right]$. Its per-type parameters $\boldsymbol{\eta}_\kappa, \sigma_\kappa^S, W_\kappa, \mathbf{b}_\kappa$ are sampled with the world as specified in Table 16.

Table 16: Sample space for the state encoder S_κ.

Parameter	**Sample space**
Raw statistics $a_\kappa(t)$	Capacity-normalized count, normalized span, transformed recency, normalized empirical rate, exponentially decayed activity, and inverted clipped inter-arrival coefficient of variation.
Projection matrix W_κ	Gaussian entries with Xavier scale $2/(d+6)$.
Bias $\mathbf{b}_\kappa$	Each coordinate uniform on $[-0.5, 0.5]$.
Activation σ_κ^S	Uniform over identity, tanh, sine, absolute value, SiLU, and sigmoid.
Output amplitude $\boldsymbol{\eta}_\kappa$	Each coordinate uniform on $[-1, 1]$.
Output dimension	$d = 6$.

E.5 Structural Map ϕ_κ

Each mark receives a fixed attribute vector a_κ^{attr} sampled with the world. For a trigger $j \to \kappa$ at time t, the structural map combines the attributes and current states of the target, its parents, and the

triggering parent:

$$x_{\kappa\leftarrow j}(t) = \mathrm{Concat}\big(a_\kappa^{\mathrm{attr}}, s_\kappa(t), \{a_p^{\mathrm{attr}}, s_p(t)\}_{p\in\mathcal{P}_\kappa}, a_j^{\mathrm{attr}}, s_j(t)\big)\,, \tag{20}$$

and maps it to the latent structural response

$$z_{\kappa\leftarrow j}(t) = \phi_\kappa(x_{\kappa\leftarrow j}(t)) = \sigma_\kappa^\phi\big(\mathrm{Norm}\big(W_\kappa^\phi x_{\kappa\leftarrow j}(t) + b_\kappa^\phi\big)\big)\,. \tag{21}$$

The response $z_{\kappa\leftarrow j}(t)$ conditions both the opportunity policy and delay law.

Each world samples complexity $c \sim \mathrm{Beta}(2.5, 2.0)$. The state-dependence scale is zero with probability 0.25 and otherwise uniform on $[0.1, 1]$. It scales the Xavier-Gaussian projection W_κ^ϕ and enables normalization. Zero state dependence sets the map weights to zero and uses the identity activation. The activation σ_κ^ϕ is sampled according to the complexity regime in Table 17.

Table 17: Complexity-dependent activation vocabulary for ϕ_κ.

Complexity regime	**Activation vocabulary**
$c < 0.33$	Identity or tanh.
$0.33 \le c < 0.66$	Identity, tanh, sine, SiLU, or sigmoid.
$c \ge 0.66$	The preceding functions plus absolute value, logarithmic, smooth-ReLU, stepwise, and modulo.
Zero state dependence	Identity activation with zero map weights.

E.6 Opportunity Policy Π_κ

The opportunity policy Π_κ converts the structural response $z_{\kappa\leftarrow j}(t)$ produced by a trigger $j \to \kappa$ into a queue update

$$\mathcal{O}^+ = \Pi_\kappa\big(\mathcal{O}^-, z_{\kappa\leftarrow j}(t)\big)\,. \tag{22}$$

Unlike an intensity update, this operation acts directly on individually represented pending opportunities: it may create new opportunities, reschedule existing ones, suppress them, or gate creation on the current structural response.

Base policy families. Each structural non-root type draws one of three base policies:

Table 18: Base opportunity-policy prior for structural non-root marks.

Policy	**Base probability**	**Parameter**
One-to-one	0.80	Creation probability $p_\kappa \sim \mathrm{Uniform}(0.3, 1)$
One-to-many	0.10	Poisson mean $\lambda_\kappa \sim \mathrm{Uniform}(2, 4)$
Many-to-one	0.10	Threshold $\theta_\kappa \sim \mathrm{Uniform}(-2, 2)$

For a *one-to-one* policy, each activation creates at most one new opportunity, with a sampled creation probability. A *one-to-many* policy draws the number of new opportunities from a Poisson distribution, allowing a single event to initiate several concurrent downstream consequences. A *many-to-one* policy instead applies a sampled threshold to the structural response, arming or disarming a gate so that multiple upstream events can jointly control downstream creation.

The one-to-many and many-to-one mixture weights are multiplied by world complexity c before normalization. Creation probabilities and Poisson means are scaled by $0.25 + 0.75s$, where $s \sim \mathrm{Uniform}(0.6, 1)$ is the structure strength, and then by a shared offspring-budget factor. After all type-specific policies are sampled, this factor is chosen so that the largest expected child count from any source mark respects the sampled world-level offspring budget.

Offspring budget. A world selects the low, medium, or high regime with probabilities 0.20, 0.55, and 0.25. Their ranges are $[0.05, 0.20]$, $[0.20, 0.55]$, and $[0.55, 0.85]$. After type-specific policies are sampled, a shared creation scale ensures that the largest expected child count from any source mark does not exceed b.

Pending-opportunity updates. Each type receives a pending capacity sampled uniformly from $\{1, 2, 3\}$. Beyond its base creation rule, a policy may alter opportunities already in the queue. For non-self-triggering types, rescheduling is enabled with probability $0.15c$. A rescheduled opportunity retains its original uniform delay draw but re-evaluates its scheduled time under the current structural response and delay law. Thus, new evidence can move an existing consequence in time without replacing its identity.

For types with at least two parents, suppression is enabled with probability $0.05c$. When active, pending opportunities are removed individually with a cancellation probability sampled uniformly from $[0.3, 0.7]$. Because opportunities remain distinct queue objects, suppression can cancel selected future consequences rather than reducing a shared aggregate rate.

Recurrence. Self-triggering is enabled with probability $0.15c$. Recursive one-to-one creation uses probability sampled uniformly from $[0.3, 0.6]$, allowing an executed event to schedule a future event of the same type. Conditional on self-triggering, the self-aware-only mode is selected with probability $0.30(0.25 + 0.75c)$.

Together, creation, gating, rescheduling, suppression, and recurrence allow Π_κ to represent one-to-one triggers, branching workflows, many-to-one gates, cancellable consequences, and overlapping event chains within the same queue.

E.7 Delay Law F_κ

For both Markov and structural worlds, delay times for opportunities of each mark follow a selected distribution family from Table 19 with the given prior probabilities. All non-root (child) marks share the same delay family; for structural worlds, the root delay family equals the child family with probability 0.5 and is otherwise sampled independently from the same list.

Table 19: Delay-family prior for EventSS roots and children.

Family	Probability	Shape range
Deterministic	0.05	Not applicable
Exponential	0.25	Not applicable
Rayleigh	0.15	Not applicable
Log-normal	0.20	Log standard deviation $[0.2, 1.3]$
Gamma	0.15	Shape $[1.2, 8]$
Weibull	0.10	Shape $[0.5, 3]$
Uniform	0.05	Width fraction $[0.3, 0.8]$
Inverse Gaussian	0.05	Shape multiplier $[1, 10]$

Each mark receives independently sampled delay parameters.

For a non-deterministic law, the structural response modulates the mean as

$$\mu_\kappa(z) = \mu_{\kappa,0}\left[1 + \frac{9}{11} m \tanh(w_{\kappa,\mu}^{\mathsf{T}} \bar{z})\right], \tag{23}$$

where $\bar{z}$ is the normalized structural response, $w_{\kappa,\mu}$ is a sampled projection, and m is zero with probability 0.30 and otherwise uniform on $[0.05, 1]$. A second projected coordinate controls shape where required. Each delay uses inverse-CDF sampling with fresh uniform noise.

F Synthetic Scenario Forecasts

This appendix extends Figure 4 with representative trajectories from the targeted zero-shot synthetic datasets. Examples were selected to be representative of the aggregate metrics. Each plot places EventFM above the ground-truth continuation and shows zero-shot FIM-PP and EVIL in the two rows below it. Marker shape and color identify event type, and capped bars show P10–P90 intervals.

Strengths and weaknesses. Figures 5–6 show EventFM's clearest advantage on processes that require local state: following a type-dependent cycle, composing interleaved schedules, or distinguishing a pending response from a new request. FIM-PP's additive type-pair intensity prior often preserves coarse cadence but loses this fine ordering, while EVIL frequently classifies structured event types well but is less accurate on narrow type-dependent timing.

The examples also expose limits. EVIL is stronger than EventFM on aggregate shipment type accuracy and OTD, and FIM-PP remains competitive on aggregate burst timing and OTD. Type-specific seasonality is a shared failure mode: the models often settle on one dominant period class instead of tracking the mixed continuation. Thus the qualitative trajectories support the aggregate results without implying that EventFM dominates every process or metric.

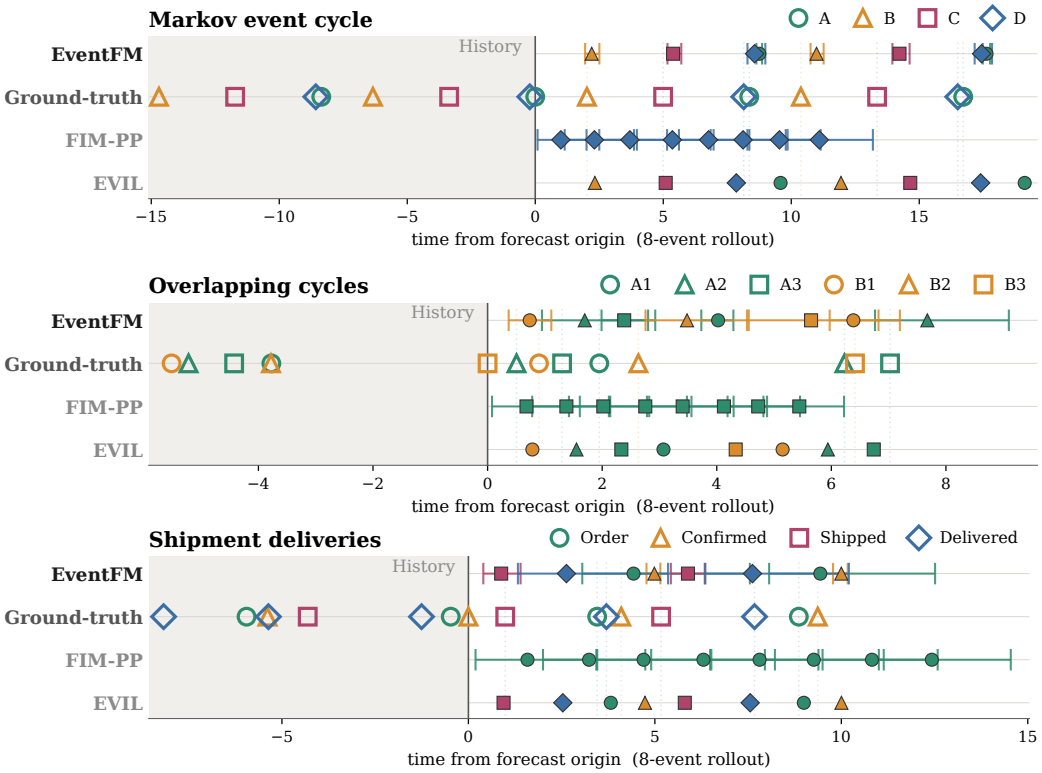


Figure 5: **Deterministic cycles and workflow structure.** Top: on the Markov event cycle, EventFM and EVIL recover the event types, but EVIL's point forecasts drift in time and FIM-PP collapses to one type. Middle: EventFM follows both interleaved cycles, whereas FIM-PP again concentrates on one type and EVIL only partially preserves the ordering. Bottom: on overlapping shipment workflows, EventFM and EVIL recover the stage sequence while FIM-PP repeats a single type; EVIL is stronger on aggregate type accuracy and OTD, while EventFM has slightly lower timing error.

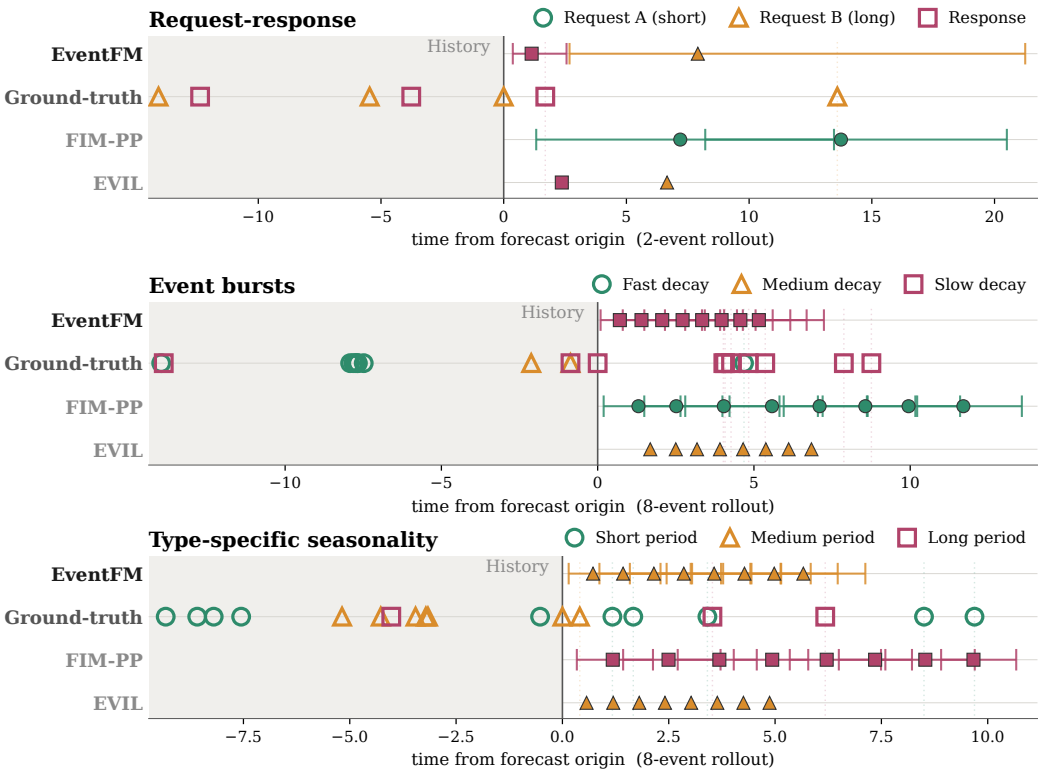


Figure 6: **Request–response and stochastic structure.** Top: after a request, EventFM predicts the response and the following request with distinct uncertainty, while FIM-PP and EVIL fail in different ways. Middle: EventFM identifies the active slow-decay burst, whereas FIM-PP and EVIL lock onto different event types. Bottom: type-specific seasonality remains difficult: all three models reduce the mixed continuation to a coarse dominant-period forecast rather than following the changing event types.

G Problem Statement: An Idealized Model of EventSS and EventFM Training

This appendix provides an idealized mathematical abstraction of the EventSS prior and EventFM training objective used in the analyses in Appendices H–I. The abstraction retains the key design choices of EventSS and EventFM: (a) a new world is sampled for every sequence; (b) the world is executed as an explicit queue of pending opportunities, processed in time order; (c) generative mechanisms read finite-memory, mark-local states; (d) triggered stochastic queue updates schedule delays by inverse-CDF sampling; (e) the model sees the history only up to time shift, time rescaling and mark relabeling; and (f) the model is trained with the loss specified in Section 3. Implementation-specific details are omitted. Exact thinning for root processes is replaced by an equivalent time-change construction; retry and fallback sampling, data-structure choices, and batch-level caps on K are omitted, with the latter absorbed into the prior over K. Regularity is imposed only where required and is stated explicitly as a restriction (Definition 2).

G.1 Event sequences

A world has a finite mark vocabulary $\mathcal{V} = \{1, \dots, K\}$, where K is itself random. An event is a pair $e_i = (\tau_i, \kappa_i) \in \mathbb{R}_{\geq 0} \times \mathcal{V}$. Sequences are ordered with $\tau_1 \leq \tau_2 \leq \cdots$. Simultaneous events are possible, for example under deterministic delays, and they are ordered by the fixed rule of Section G.3. We write $H_n = (e_1, \dots, e_n)$ for the first n events, $H_t = \{e_i : \tau_i \leq t\}$ for the history up to time t, and $H_t^\kappa = \{e_i \in H_t : \kappa_i = \kappa\}$ for its type-κ restriction. The *candidate set* $\mathcal{K}_n = \{\kappa_1, \dots, \kappa_n\}$ is the set of marks that appear in H_n.

G.2 Structural worlds

Definition 1 (World)**.** *A world is a pair $W = (S, \vartheta)$. The* structure $S \in \mathcal{S}$ *takes values in a finite set. It comprises the execution mode (structural or Markov), the number of mark types K, the dependency graph G, the policy family of each type, the delay family of each type, the activation of each structural map, and the mode of each source, and discrete settings such as pending capacity and memory capacity. The* parameters $\vartheta \in \Theta_S \subset \mathbb{R}^{p(S)}$ *are continuous: projection weights, rates, shapes, creation probabilities, thresholds and time scales. The prior factorizes as $P(W) = P(S)\, P(\vartheta \mid S)$, with the coupling through the world-level variables ξ absorbed into this joint law.*

The split into S and ϑ matters later because S has finite Shannon entropy and ϑ does not. The information-theoretic bounds we derive that use $H(W)$ apply at the level of S, and parameters require separate treatment.

Conditionally on W, a world specifies the following components.

(i) Graph and sources. $G = (\mathcal{V}, E)$ is a directed dependency graph. In structural worlds, G is acyclic and recurrence is represented separately by a self-triggering policy. In Markov worlds, the transition graph may contain cycles and self-loops. Let $\mathcal{P}_\kappa = \{j : (j, \kappa) \in E\}$ denote the parents of κ and $\mathcal{C}_j = \{\kappa : (j, \kappa) \in E\}$ its children. A set $\mathcal{R} \subseteq \mathcal{V}$ of *source types* receives exogenous activity. In structural worlds $\mathcal{R}$ is the set of parentless types. In Markov worlds $\mathcal{R} = \emptyset$ and activity comes only from the initial configuration $\mathcal{O}_0$, which contains one opportunity at time 0 per connected component.

(ii) Mark-local memory and state. Each type κ carries a finite-dimensional memory $m_\kappa(t) \in \mathcal{M}_\kappa$. The memory is updated by a map $m_\kappa \leftarrow U_\kappa(m_\kappa, \tau)$ when an event of type κ occurs at τ, and is constant between such events. The state is

$$s_\kappa(t) = S_\kappa\big(m_\kappa(t),\, t\big) \in \mathbb{R}^d, \tag{24}$$

where S_κ is continuous in t. This covers the finite-capacity statistics of Appendix E.4: counts, span and rate over the last c_κ events, recency, and exponentially decayed activity.

(iii) Opportunities. The queue $\mathcal{O}$ is a finite set of individually identified opportunities $o = (\kappa_o, \tilde{\tau}_o, u_o, \zeta_o)$. Here κ_o is the target mark, $\tilde{\tau}_o$ the scheduled time, $u_o \in (0,1)$ the stored uniform draw, and ζ_o the trigger record (triggering event, mark and time). Each type also carries a mechanism state g_κ from a finite set; this holds the gate status of many-to-one policies.

(iv) Delay laws. For each type, $F_\kappa(\cdot \mid z)$ is a distribution on $(0,\infty)$ indexed by a response z. It may have atoms, as with deterministic families. Scheduling is by inverse CDF:

$$\tilde{\tau}_o = t_{\mathrm{ref}} + F_\kappa^{-1}(u_o \mid z). \tag{25}$$

A newly created opportunity uses a fresh draw $u_o \sim \mathrm{Unif}(0,1)$ and sets t_{ref} equal to the trigger time. A rescheduled opportunity keeps its stored u_o and re-evaluates equation 25 under the current response. Delays are strictly positive, so every opportunity is scheduled strictly after the event that created or rescheduled it.

(v) Sources. The source of type $r \in \mathcal{R}$, denoted B_r in Appendix E.3, always holds exactly one pending source opportunity. When it executes at time t, the next one is scheduled at $t + F_r^{-1}(u \mid t, m_r(t))$ with a fresh u. Stationary renewal sources do not depend on t. Periodic, decaying and envelope-modulated Poisson sources do, through the time change $F_r(\delta \mid t) = 1 - \exp\{-\int_t^{t+\delta} \lambda_r(v)\, dv\}$. This generates the same law as thinning.

(vi) Triggered response. When an event of type j occurs at time t, the queue and mechanism states are updated by a stochastic operator

$$(\mathcal{O}^+, g^+) \sim \Gamma_j^W\big(\cdot \mid \mathcal{O}^-, g^-, s(t), t\big), \tag{26}$$

where $s(t) = (s_1(t), \ldots, s_K(t))$. Two constraints encode the queue semantics of EventSS: (a) Γ_j^W may create, cancel or reschedule only opportunities whose target mark is a child of j, and may change only the mechanism states of those children; (b) it respects the per-type pending capacities. The two execution modes are instances of this operator.

- *Structural.* The children $\kappa \in \mathcal{C}_j$ are processed in a fixed order. For each child, a latent response $z_{\kappa \leftarrow j}(t) = \phi_\kappa\big(s_\kappa(t), s_{\mathcal{P}_\kappa}(t), s_j(t)\big)$ is computed, and the child's policy then applies $(\mathcal{O}, g_\kappa) \leftarrow \Pi_\kappa(\mathcal{O}, g_\kappa, z_{\kappa \leftarrow j}(t))$. The policy covers creation (one-to-one, one-to-many or gated), suppression and rescheduling, with delays drawn by equation 25. Persistent mark attributes are constants of W and are absorbed into ϕ_κ.
- *Markov.* A single successor $\kappa \sim T_W(j, \cdot)$ is drawn, and one opportunity for κ is created with delay law F_κ.

G.3 Execution

The simulator state is $Z_t = \big(t, \mathcal{O}_t, (m_\kappa(t))_\kappa, (g_\kappa(t))_\kappa\big)$. Between events it changes only through t, so (Z_t) is a piecewise-deterministic Markov process. For worlds whose sources do not depend on absolute time, the t coordinate can be dropped. Starting from $\mathcal{O}_0$, which contains the initial source opportunities or Markov seeds, the simulator repeats four steps:

1. Select $o^\star = \arg\min_{o \in \mathcal{O}}(\tilde{\tau}_o, \iota_o)$, where ι_o is a fixed tie-breaking key such as the creation index. Emit $e_{n+1} = (\tilde{\tau}_{o^\star}, \kappa_{o^\star})$ and remove $o^\star$ from the queue.
2. Update the memory $m_{\kappa_{n+1}}$.
3. If $\kappa_{n+1} \in \mathcal{R}$, schedule the next source opportunity as in item (v).
4. Apply $\Gamma^W_{\kappa_{n+1}}$ from equation 26.

We write $P(\cdot \mid W)$ for the induced law of the event sequence, and $P(e_{n+1} \in \cdot \mid H_n, W)$ for its next-event conditional. We assume that every world produces infinitely many events almost surely. This holds for Markov worlds, and for structural worlds with at least one source, provided each Poisson-type source has rate bounded below; renewal sources always generate infinitely many events. Retry and fallback sampling is therefore not modeled.

Remark 2 (The queue is latent). The history H_n does not determine Z_{τ_n}. Pending opportunities, their stored draws and the gate states are all unobserved. Hence $P(e_{n+1} \mid H_n, W)$ is a filtering distribution that integrates over the hidden queue, and the event sequence is a *hidden*-state process even when W is known. Properties of (Z_t), such as stability and ergodicity, do not transfer automatically to properties of $P(e_{n+1} \mid H_n, W)$.

Definition 2 (Regularity restrictions)**.** *Later results may assume a subset of the following conditions.*

(R1) Continuous delays. *Every $F_\kappa(\cdot \mid z)$ and every source delay law has a density with respect to Lebesgue measure. This excludes the deterministic family and deterministic motif delays; under (R1), ties occur with probability zero apart from initial seeds.*

(R2) Time homogeneity. *Sources do not depend on absolute time. This excludes periodic and decaying modes and the global envelope.*

(R3) Single seeding. *$\mathcal{O}_0$ contains at most one opportunity at any given time.*

G.4 Model class and training objective

Invariances. For $n \geq 2$ with $s_n = \tau_n - \tau_1 > 0$, let $\nu(H_n)$ be the equivalence class of H_n under time shifts, positive time rescalings and relabelings of marks. This is the information EventFM reads. Its inputs are $(\tau_i - \tau_1)/s_n$, the ratios $\Delta\tau_i/\bar{\Delta}_n$ with $\bar{\Delta}_n = s_n/(n-1)$, per-pass random mark codes, and the relabeling-invariant mark-context features (Section 3). When all context events share one timestamp, ν is undefined; under (R1) and (R3) this has probability zero. Write $\delta_{n+1} = (\tau_{n+1} - \tau_n)/\bar{\Delta}_n$ for the normalized next delay, which is invariant to shift and rescaling.

Because ν forgets mark identities, marks are identified by their canonical labels, i.e., by order of first appearance in H_n, with a new mark receiving label $|\mathcal{K}_n| + 1$ (Appendix H). Events such as $\{\kappa_{n+1} = k\}$ are read in these labels, and $e^\circ_{n+1} = (\ell_{n+1}(\kappa_{n+1}), \delta_{n+1})$.

Outputs. Given H_n, the model outputs two quantities, both of which depend on H_n only through $\nu(H_n)$, equivariantly in the candidate labels:

- a distribution $p_\theta(\cdot \mid H_n)$ on $\mathcal{K}_n$;
- for each $k \in \mathcal{K}_n$, nondecreasing normalized-delay quantiles $q_{\theta,\alpha}(k, H_n)$, $\alpha \in \mathcal{A}$, with absolute quantiles $\hat{Q}_\alpha(k, H_n) = \tau_n + \bar{\Delta}_n\, q_{\theta,\alpha}(k, H_n)$.

Training distribution. Each training example is generated by drawing $N \sim P_N$ on $\{3, \dots, N_{\max}\}$ (uniform, with $N_{\max} = 256$), then $W \sim P(W)$ and $H_N \sim P(\cdot \mid W)$. The first event is shifted to time 0, and the example is (H_n, e_{n+1}) with $n = N - 1$. Let $\mathbb{P}$ denote the joint law of (N, W, H_N).

Objective. EventFM minimizes

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbb{P}}\Big[\mathbb{1}\{\kappa_{n+1} \in \mathcal{K}_n\}\Big(\lambda_\kappa\big(-\log p_\theta(\kappa_{n+1} \mid H_n)\big) + \frac{\lambda_\tau}{|\mathcal{A}|}\sum_{\alpha\in\mathcal{A}} \rho_\alpha\big(\delta_{n+1} - q_{\theta,\alpha}(\kappa_{n+1}, H_n)\big)\Big)\Big], \tag{27}$$

where $\rho_\alpha(u) = u(\alpha - \mathbb{1}\{u < 0\})$ is the pinball loss. This is equation 9 written in the normalized units in which it is evaluated (Appendix C). Examples whose target mark does not appear in the context contribute to neither term. Evaluating the pinball loss on normalized delays matters for what is learned. In absolute time, the identity $\rho_\alpha(\bar{\Delta}_n x) = \bar{\Delta}_n \rho_\alpha(x)$ would weight each example by $\bar{\Delta}_n$, which is not a function of $\nu(H_n)$. The learned quantiles would then be quantiles of a scale-reweighted law rather than of the conditional law in Proposition 2(b).

Decoding. The quantities used at inference are constructed from the outputs above. Let $\mathcal{A} = \{\alpha_1 < \cdots < \alpha_J\}$. The *candidate CDF* $\hat{F}_k(\cdot \mid H_n)$ is the continuous, nondecreasing function obtained by linear interpolation through the knots $(q_{\theta,\alpha_j}(k, H_n), \alpha_j)$. Below the first knot it interpolates to the lower endpoint $(0, 0)$; above the last knot it interpolates to a finite extrapolated upper endpoint with value 1. The *marginal CDF* is $\hat{F}(\cdot \mid H_n) = \sum_{k\in\mathcal{K}_n} p_\theta(k \mid H_n)\hat{F}_k(\cdot \mid H_n)$. The time

point forecast is the marginal median $\hat{m}_n$, a solution of $\hat{F}(\hat{m}_n \mid H_n) = 1/2$, reported in absolute time as $\tau_n + \bar{\Delta}_n \hat{m}_n$. The mark point forecast is $\arg\max_{k \in \mathcal{K}_n} p_\theta(k \mid H_n)$. Autoregressive decoding samples a mark from p_θ and uses the median of $\hat{F}_k$ for its delay.

Assumption 1 (Realizability)**.** *The model class contains every pair (p, q) that satisfies the invariance constraints above.*

Proposition 2 (Population target of EventFM)**.** *Suppose Assumption 1 holds and $\theta^\star$ minimizes equation 27. Then, for $\mathbb{P}$-almost every history H_n with $n \in \{2, \dots, N_{\max} - 1\}$ and $\mathbb{P}(\kappa_{n+1} \in \mathcal{K}_n \mid \nu(H_n)) > 0$:*

(a) $p_{\theta^\star}(k \mid H_n) = \mathbb{P}\big(\kappa_{n+1} = k \mid \nu(H_n),\, \kappa_{n+1} \in \mathcal{K}_n\big)$ for every $k \in \mathcal{K}_n$;

(b) for each k with positive conditional probability, $q_{\theta^\star,\alpha}(k, H_n)$ is an α-quantile of $\mathbb{P}\big(\delta_{n+1} \in \cdot \mid \nu(H_n),\, \kappa_{n+1} = k\big)$, and it is the unique such quantile whenever that quantile is unique.

Proof. The outputs depend on H_n only through $\nu(H_n)$, so $\mathcal{L}$ can be minimized separately for each value of $\nu(H_n)$, by conditioning on it. The mark term and the time term involve disjoint outputs. Each can therefore be minimized separately, and the weights $\lambda_\kappa, \lambda_\tau > 0$ do not affect the minimizer. For (a): minimizing $\mathbb{E}[\mathbb{1}\{\kappa \in \mathcal{K}_n\}(-\log p(\kappa)) \mid \nu]$ over distributions on $\mathcal{K}_n$ gives $p(k) \propto \mathbb{P}(\kappa_{n+1} = k \mid \nu)$, by strict propriety of the log score. For (b): for fixed k and α, the conditional expected pinball loss is minimized exactly on the α-quantiles of the conditional law of δ_{n+1}. The true quantiles are nondecreasing in α, so the monotonicity constraint does not bind. □

Remark 3 (Interpretation). By construction,

$$\mathbb{P}\big(e^\circ_{n+1} \in \cdot \mid \nu(H_n)\big) = \int \mathbb{P}\big(e^\circ_{n+1} \in \cdot \mid \nu(H_n), W\big)\, \mathbb{P}\big(dW \mid \nu(H_n)\big),$$

where $e^\circ_{n+1} = (\delta_{n+1}, \ell_{n+1}(\kappa_{n+1}))$. EventFM therefore targets a Bayesian posterior predictive in which the world, the time origin, the time unit and the mark identities are all unknown and are inferred from the single observed history. This is the precise sense in which it performs in-context inference. The target differs from the posterior predictive of the raw history, because it discards absolute scale. Under the EventSS prior, absolute scale can be informative about the latent world, so discarding it is a cost incurred by the invariance design.

Remark 4 (What training does and does not determine). Proposition 2 determines EventFM's outputs only in the following sense.

(i) It fixes the restricted mark probabilities and the conditional quantiles at the levels in $\mathcal{A}$. It does not fix a time density or the CDF between knots. The candidate and marginal CDFs, the marginal median, samples and CRPS values are therefore post-hoc constructions from these outputs. At the population optimum, $\hat{F}_k$ agrees with the Bayes conditional CDF to within the level spacing of $\mathcal{A}$; see Proposition 3(d). The marginal median is the Bayes decision for absolute error, not for squared error.

(ii) It says nothing about first occurrences. Marks outside $\mathcal{K}_n$ receive probability zero, so the model cannot forecast the first occurrence of a type.

(iii) It covers only context lengths $n \le N_{\max} - 1$.

(iv) It covers only histories in the support of $\mathbb{P}$. On real-world histories outside that support the minimizer is not constrained by equation 27, and behavior there reflects the inductive bias of the architecture.

For analysis it is convenient to work with the full normalized predictive $\mathbb{P}(e^\circ_{n+1} \mid \nu(H_n))$ and its log loss, which we call the *idealized log-loss predictor*. EventFM's outputs are functionals of this predictor, and Appendices H and I state which of their conclusions transfer to these functionals.

H In-context excess risk of the Bayes-optimal predictor

This appendix examines how quickly predictions from a single history approach those of an oracle that knows the generating world. The object of study is the Bayes predictor of Proposition 2. We

bound its cumulative and average excess log loss relative to the oracle and derive rates. We then transfer these bounds to the outputs EventFM actually produces, namely mark probabilities and conditional quantiles. None of the results requires stability, ergodicity or identifiability of the simulator. The techniques are classical: we adapt the information-theoretic analysis of Bayes mixture predictors (Clarke & Barron, 1990; Barron, 1999) to the PFN setting (Müller et al., 2022; Nagler, 2023). What is specific to EventFM is the observation model (normalized, relabeled histories of a hidden-queue event process) and the transfer of the bounds to the mark probabilities and quantiles it actually outputs. The bounds are asymptotic in character and serve to interpret EventFM's behavior, not to certify it. In this appendix and the next, we assume that the true world $W^\star$ satisfies (R1) and (R3) of Definition 2. The prior may contain worlds that violate them; on histories whose context events all share one timestamp we set X_n to a fixed symbol $\star$, an event of $P_{W^\star}$-probability zero, so the results below are unaffected. Corollary 3, which averages over the prior, requires (R1) and (R3) for P-almost every world.

H.1 Normalized observations

For $n \geq 2$ let $\bar{\Delta}_n = (\tau_n - \tau_1)/(n-1)$. The *canonical form* of a history is

$$X_n = c(H_n) = \Big(\tfrac{\tau_i - \tau_1}{\bar{\Delta}_n},\, \ell_n(\kappa_i) \Big)_{i=1}^{n},$$

where ℓ_n relabels marks by order of first appearance. X_n is a maximal invariant for time shift, time rescaling and mark relabeling, so it carries exactly the information in $\nu(H_n)$. The *normalized next event* is

$$e^\circ_{n+1} = \big(\delta_{n+1},\, \ell_{n+1}(\kappa_{n+1}),\big), \qquad \delta_{n+1} = \frac{\tau_{n+1} - \tau_n}{\bar{\Delta}_n}.$$

Its label is either an existing label in $\{1, \dots, |\mathcal{K}_n|\}$ or the new label $|\mathcal{K}_n| + 1$.

Lemma 1 (Sequential structure)**.** *For $n \geq 2$, X_{n+1} is a measurable bijective function of (X_n, e°_{n+1}).*

Proof. Write $t'_i = (\tau_i - \tau_1)/\bar{\Delta}_n$ for $i \leq n+1$. For $i \leq n$ these are the time coordinates of X_n, and $t'_{n+1} = t'_n + \delta_{n+1}$. Since $\bar{\Delta}_{n+1} = \bar{\Delta}_n t'_{n+1}/n$, the time coordinates of X_{n+1} are $n\, t'_i/t'_{n+1}$ for $i \leq n+1$. Labels by first appearance are unchanged for existing marks, and a new mark receives label $|\mathcal{K}_n| + 1$. Conversely, X_n is obtained from X_{n+1} by dropping the last event and rescaling, and δ_{n+1} is the rescaled difference of its last two time coordinates. □

For a world W and $N \geq 2$, let $P_W^{(N)}$ denote the law of X_N under $P(\cdot \mid W)$. Let

$$M^{(N)} = \int P_W^{(N)}\, P(dW)$$

be the corresponding mixture. The *Bayes predictor* $B_n(\cdot \mid X_n) = M(e^\circ_{n+1} \in \cdot \mid X_n)$ satisfies

$$B_n(\cdot \mid X_n) = \int P_W(e^\circ_{n+1} \in \cdot \mid X_n)\, P(dW \mid X_n).$$

Because N is drawn independently of W, B_n equals the conditional law $\mathbb{P}(e^\circ_{n+1} \in \cdot \mid \nu(H_n))$ of Proposition 2 at every context length $n \leq N_{\max} - 1$. The *oracle predictor* $P_{W^\star}(e^\circ_{n+1} \in \cdot \mid X_n)$ knows the world but sees the same normalized history. By Remark 2 it is still a filter over the hidden queue, so it does not predict perfectly.

Definition 3 (In-context excess risk)**.** *For a world $W^\star$ and $n \geq 2$, the excess risk at step $n+1$ is*

$$R_n(W^\star) = \mathbb{E}_{X_n \sim P_{W^\star}^{(n)}} \Big[D_{\mathrm{KL}}\big(P_{W^\star}(e^\circ_{n+1} \in \cdot \mid X_n) \,\big\|\, B_n(\cdot \mid X_n) \big) \Big].$$

This is the expected additional log loss of the Bayes predictor over the oracle. The cumulative excess risk is $C_N(W^\star) = \sum_{n=2}^{N-1} R_n(W^\star)$. If $D_{\mathrm{KL}}(P_{W^\star}^{(N)} \| M^{(N)}) = \infty$, the Bayes predictor need not be defined on the support of $P_{W^\star}^{(n)}$, and we set $C_N(W^\star) = +\infty$.

H.2 BOUNDS

Lemma 2 (Redundancy identity). *For every $W^\star$ and $N \geq 3$ with $D_{\mathrm{KL}}(P_{W^\star}^{(N)} \| M^{(N)}) < \infty$,*

$$C_N(W^\star) = D_{\mathrm{KL}}\big(P_{W^\star}^{(N)} \,\|\, M^{(N)}\big) - D_{\mathrm{KL}}\big(P_{W^\star}^{(2)} \,\|\, M^{(2)}\big) \leq D_{\mathrm{KL}}\big(P_{W^\star}^{(N)} \,\|\, M^{(N)}\big).$$

Proof. By Lemma 1, X_N is in bijection with $(X_2, e_3^\circ, \ldots, e_N^\circ)$. Apply the chain rule for the KL divergence to this sequence under $P_{W^\star}$ and M. The conditional laws of M are B_n. The chain rule holds for general laws on standard Borel spaces, so no densities are required. □

Lemma 3 (Prior-mass bound). *Let $W^\star = (S^\star, \vartheta^\star)$ with $P(S^\star) > 0$, and let $A \subseteq \Theta_{S^\star}$ satisfy $\Pi(A) = P(\vartheta \in A \mid S^\star) > 0$. Then*

$$D_{\mathrm{KL}}\big(P_{W^\star}^{(N)} \,\|\, M^{(N)}\big) \leq \log \frac{1}{P(S^\star)} + \log \frac{1}{\Pi(A)} + \sup_{\vartheta \in A} D_{\mathrm{KL}}\big(P_{W^\star}^{(N)} \,\|\, P_{(S^\star,\vartheta)}^{(N)}\big).$$

Proof. Let $M_A = \int_A P_{(S^\star,\vartheta)}^{(N)} \, \Pi(d\vartheta)/\Pi(A)$. Then $M^{(N)} \geq P(S^\star)\,\Pi(A)\,M_A$ as measures. Hence $\log \frac{dP_{W^\star}}{dM^{(N)}} \leq \log \frac{dP_{W^\star}}{dM_A} + \log \frac{1}{P(S^\star)\Pi(A)}$ wherever the left side is defined. Integrating under $P_{W^\star}$ gives the first two terms plus $D_{\mathrm{KL}}(P_{W^\star} \| M_A)$. Convexity of the KL divergence in its second argument gives $D_{\mathrm{KL}}(P_{W^\star} \| M_A) \leq \sup_{\vartheta \in A} D_{\mathrm{KL}}(P_{W^\star} \| P_{(S^\star,\vartheta)})$. If the supremum is infinite, the bound holds trivially. □

Assumption 2 (Local regularity at $W^\star$). *Let $p = p(S^\star)$, and suppose $\vartheta^\star$ lies in the interior of $\Theta_{S^\star} \subseteq \mathbb{R}^p$. There exist $r_0 > 0$, $L < \infty$, $\beta \in (0, 2]$ and $\pi_{\min} > 0$ such that for every ϑ with $\|\vartheta - \vartheta^\star\| \leq r_0$ the following hold.*

*(a) (*KL growth*) For all $N \geq 3$, $D_{\mathrm{KL}}\big(P_{W^\star}^{(N)} \,\|\, P_{(S^\star,\vartheta)}^{(N)}\big) \leq L\,N\,\|\vartheta - \vartheta^\star\|^\beta$.*

*(b) (*Prior density*) $\vartheta \mid S^\star$ has a Lebesgue density of at least $\pi_{\min}$ at ϑ.*

By the data-processing inequality, condition (a) may be checked on the raw histories H_N, or on any richer record that determines them (Remark 7). The case $\beta = 2$ is the smooth case. Smaller β covers mechanisms whose laws change non-smoothly in ϑ.

Theorem 1 (In-context excess risk). *Suppose Assumption 2 holds, and let v_p denote the volume of the unit ball in $\mathbb{R}^p$. Then for all $N \geq \max(3, r_0^{-\beta})$,*

$$C_N(W^\star) \;\leq\; \underbrace{\log \frac{1}{P(S^\star)}}_{\textit{structure}} \;+\; \underbrace{\frac{p}{\beta} \log N}_{\textit{parameters}} \;+ L + \log \frac{1}{\pi_{\min} v_p}. \tag{28}$$

Consequently, the average excess risk satisfies

$$\frac{1}{N-2} \sum_{n=2}^{N-1} R_n(W^\star) = O\Big(\frac{\log(1/P(S^\star)) + (p/\beta)\log N}{N}\Big) \to 0.$$

Proof. By Lemma 3, the KL divergence to $M^{(N)}$ is finite, so Lemma 2 applies. Combine Lemmas 2 and 3 with the ball $A = \{\vartheta : \|\vartheta - \vartheta^\star\| \leq N^{-1/\beta}\}$. This ball lies within radius r_0 because $N \geq r_0^{-\beta}$. On it, Assumption 2(a) gives a KL divergence of at most L, and (b) gives $\Pi(A) \geq \pi_{\min} v_p N^{-p/\beta}$. □

Corollary 2 (All but logarithmically many steps). *Under the conditions of Theorem 1, for every $t > 0$,*

$$\#\{n \in \{2, \ldots, N-1\} : R_n(W^\star) > t\} \leq C_N(W^\star)/t = O\big((\log(1/P(S^\star)) + (p/\beta)\log N)/t\big).$$

Proof. $R_n \geq 0$, so the sum of the terms exceeding t is at most C_N. □

Corollary 3 (Prior-averaged risk)**.** *For every $N \geq 3$, with I denoting mutual information under the joint law of (W, X_N),*

$$\mathbb{E}_{W^\star \sim P}\big[C_N(W^\star)\big] = I(W; X_N) - I(W; X_2) \leq H(S) + I(\vartheta; X_N \mid S).$$

If Assumption 2 holds with constants $r_0, L, \beta, \pi_{\min}$ that do not depend on $W^\star$, then for $N \geq \max(3, r_0^{-\beta})$,

$$\mathbb{E}_{W^\star}\big[C_N(W^\star)\big] \leq H(S) + \mathbb{E}\Big[\tfrac{p(S)}{\beta} \log N + \log \tfrac{1}{\pi_{\min} v_{p(S)}}\Big] + L.$$

Proof. For the identity, average Lemma 2 over $W^\star$ and use $\mathbb{E}_W D_{\mathrm{KL}}(P_W \| M) = I(W; X)$. For the first inequality, apply the chain rule $I(W; X_N) = I(S; X_N) + I(\vartheta; X_N \mid S)$ and use $I(S; X_N) \leq H(S) < \infty$, since S is finite-valued. For the second, average equation 28 and use $\mathbb{E}[\log(1/P(S))] = H(S)$. □

H.3 Transfer to EventFM's outputs

EventFM outputs the restricted mark probabilities and the conditional quantiles of the Bayes predictor (Proposition 2). Excess log loss controls both. For a history X_n, define the following quantities:

- the oracle mark probabilities $\pi_k(X_n) = P_{W^\star}(\ell_{n+1}(\kappa_{n+1}) = k \mid X_n)$;
- the oracle probability that the next mark is already in the context, $a(X_n) = \sum_{k \leq |\mathcal{K}_n|} \pi_k(X_n)$;
- the total-variation gap $\mathrm{TV}_n(X_n) = \sup_E \big|P_{W^\star}(e^\circ_{n+1} \in E \mid X_n) - B_n(E \mid X_n)\big|$.

Proposition 3 (Mark and quantile accuracy)**.** *Suppose Assumption 1 holds, $2 \leq n \leq N_{\max} - 1$, and $D_{\mathrm{KL}}(P^{(N)}_{W^\star} \| M^{(N)}) < \infty$ for some $N > n$ (for example under Assumption 2, by Lemma 3). Then $P^{(n)}_{W^\star} \ll M^{(n)}$, and for $P_{W^\star}$-almost every X_n the following hold.*

(a) EventFM's mark distribution satisfies $\big\|p_{\theta^\star}(\cdot \mid H_n) - \pi(\cdot \mid X_n, \kappa_{n+1} \in \mathcal{K}_n)\big\|_{\mathrm{TV}} \leq 2\,\mathrm{TV}_n(X_n)/a(X_n)$. Here the second argument is the oracle's mark distribution restricted to marks in the context.

(b) Fix a mark k with $\pi_k(X_n) > 0$, and let $\eta_k = 2\,\mathrm{TV}_n(X_n)/\pi_k(X_n)$. Let $F^\star_k$ be the oracle's conditional distribution function of δ_{n+1} given label k. Then every EventFM quantile $q = q_{\theta^\star,\alpha}(k, H_n)$ satisfies

$$F^\star_k(q^-) \leq \alpha + \eta_k \quad \text{and} \quad F^\star_k(q) \geq \alpha - \eta_k.$$

That is, q is an oracle quantile at a level within η_k of α.

(c) In expectation, $\mathbb{E}\big[\mathrm{TV}_n\big] \leq \sqrt{R_n(W^\star)/2}$, and under Assumption 2

$$\frac{1}{N-2} \sum_{n=2}^{N-1} \mathbb{E}\big[\mathrm{TV}_n\big] \leq \sqrt{\frac{C_N(W^\star)}{2(N-2)}} = O\Big(\sqrt{\tfrac{\log(1/P(S^\star)) + (p/\beta)\log N}{N}}\Big).$$

*(d) (*Marginal median.*) Let $\mathcal{A} = \{\alpha_1 < \cdots < \alpha_J\}$ and define the level spacing $\Delta_{\mathcal{A}} = \max\{\alpha_1,\ \max_j(\alpha_{j+1} - \alpha_j),\ 1 - \alpha_J\}$. Let $F^\star$ be the oracle's distribution function of δ_{n+1}, restricted to marks in the context, i.e. conditioned on $\ell_{n+1}(\kappa_{n+1}) \leq |\mathcal{K}_n|$. Then EventFM's normalized marginal median $\hat{m}_n$ (Appendix G, Decoding) satisfies*

$$F^\star(\hat{m}_n^-) \leq \tfrac{1}{2} + \varepsilon_n \quad \text{and} \quad F^\star(\hat{m}_n) \geq \tfrac{1}{2} - \varepsilon_n, \qquad \varepsilon_n = \Delta_{\mathcal{A}} + \frac{2\,\mathrm{TV}_n(X_n)}{a(X_n)}.$$

That is, $\hat{m}_n$ is an oracle quantile at a level within ε_n of $1/2$.

Proof. For probability measures P, Q and an event E with $P(E) > 0$, the conditional laws satisfy $\|P(\cdot \mid E) - Q(\cdot \mid E)\|_{\mathrm{TV}} \leq 2\|P - Q\|_{\mathrm{TV}}/P(E)$. To see this, write

$$P(D \mid E) - Q(D \mid E) = \frac{P(D \cap E) - Q(D \cap E)}{P(E)} + \frac{Q(D \cap E)}{Q(E)} \cdot \frac{Q(E) - P(E)}{P(E)},$$

and note that each term is at most $\|P - Q\|_{\mathrm{TV}}/P(E)$ in absolute value. Part (a) follows by taking $E = \{\text{label} \le |\mathcal{K}_n|\}$, marginalizing to the mark, and using Proposition 2(a). For part (b), take $E = \{\text{label} = k\}$. The Kolmogorov distance between the two conditional delay laws is bounded by their total-variation distance, which is at most η_k. By Proposition 2(b), q is an α-quantile of the Bayes conditional law F_k^B, so $F_k^B(q^-) \le \alpha \le F_k^B(q)$. Moving from F_k^B to $F_k^\star$ changes each side by at most η_k. For part (c), fix a history X_n and apply Pinsker's inequality to the pair of next-event laws $P_{W^\star}(e_{n+1}^\circ \in \cdot \mid X_n)$ (the oracle) and $B_n(\cdot \mid X_n)$ (the Bayes predictor). This gives

$$\mathrm{TV}_n(X_n) \le \sqrt{\tfrac{1}{2}\, D_{\mathrm{KL}}\big(P_{W^\star}(e_{n+1}^\circ \in \cdot \mid X_n) \,\big\|\, B_n(\cdot \mid X_n)\big)}.$$

Take expectations over $X_n \sim P_{W^\star}^{(n)}$. Since $x \mapsto \sqrt{x}$ is concave, Jensen's inequality lets us move the expectation inside the square root, and the expected KL divergence is $R_n(W^\star)$ by Definition 3. This yields $\mathbb{E}[\mathrm{TV}_n] \le \sqrt{R_n(W^\star)/2}$. For the averaged bound, apply Jensen's inequality once more, now to the uniform average over the $N-2$ steps:

$$\frac{1}{N-2}\sum_{n=2}^{N-1}\sqrt{\frac{R_n(W^\star)}{2}} \le \sqrt{\frac{1}{N-2}\sum_{n=2}^{N-1}\frac{R_n(W^\star)}{2}} = \sqrt{\frac{C_N(W^\star)}{2(N-2)}}.$$

The rate then follows from Theorem 1.

For part (d), let F_k^B be the Bayes conditional CDF of δ_{n+1} given label k, and $F^B = \sum_k p_{\theta^\star}(k \mid H_n) F_k^B$ the Bayes marginal restricted to $\mathcal{K}_n$. The argument has three steps.

(1) *Interpolation error.* By Proposition 2(b), the knots of $\hat{F}_k$ are Bayes quantiles q_j at levels α_j, so $F_k^B(q_j^-) \le \alpha_j \le F_k^B(q_j)$. Since both functions are nondecreasing, for $x \in [q_j, q_{j+1})$ both $\hat{F}_k(x)$ and $F_k^B(x)$ lie in $[\alpha_j, \alpha_{j+1}]$, and for $x \in (q_j, q_{j+1}]$ so do their left limits. Below q_1 both lie in $[0, \alpha_1]$, and above q_J both lie in $[\alpha_J, 1]$. Hence $\sup_x |\hat{F}_k(x) - F_k^B(x)| \le \Delta_{\mathcal{A}}$, and the same bound holds with left limits. Taking the mixture with weights $p_{\theta^\star}$ gives $\|\hat{F} - F^B\|_\infty \le \Delta_{\mathcal{A}}$.

(2) *Bayes versus oracle.* By Proposition 2(a), F^B is the delay marginal of the Bayes law conditioned on $E = \{\text{label} \le |\mathcal{K}_n|\}$. By the conditioning inequality above, its Kolmogorov distance to $F^\star$ is at most $2\,\mathrm{TV}_n/a(X_n)$.

(3) *Median.* $\hat{F}$ is continuous, so $\hat{F}(\hat{m}_n) = 1/2$. Combining this with (1) and (2) gives the claim.

□

H.4 Interpretation, scope and limitations

Remark 5 (What the bound says). Equation 28 splits the cost of learning in context into two parts.

- *Structure.* Identifying the structure (execution mode, graph, mechanism families) costs at most $\log(1/P(S^\star))$ nats in total, however long the history. Averaged over the prior, this is at most the finite entropy $H(S)$. Structures to which EventSS assigns little prior mass are therefore learned more slowly. This makes explicit how the design of the prior determines which kinds of dynamics EventFM adapts to quickly.
- *Parameters.* Continuous parameters cost $(p/\beta)\log N$. This is the familiar $\frac{p}{2}\log N$ in the smooth case. The average excess risk therefore decays as $O(p \log N/N)$, not $O(1/N)$.

Corollary 2 converts this into a per-step statement: only logarithmically many steps can have excess risk above any fixed level. We do not claim a rate for R_n at each individual step. Such a rate would follow if R_n were nonincreasing, but this is guaranteed only for the prior-averaged risk under exchangeability, not for a fixed world or for hidden-state, possibly nonstationary event processes. Proposition 3 shows how the guarantee reaches EventFM's outputs. Mark probabilities and quantile levels converge at rate $O(\sqrt{p \log N/N})$ on average. The guarantee is weaker for marks the oracle itself considers unlikely, because of the factor $1/\pi_k$. The timing of rare marks is therefore learned more slowly than their occurrence. The marginal median used for point forecasts avoids this factor;

it divides only by the probability $a(X_n)$ that the next mark has already been seen. It does, however, carry a discretization floor $\Delta_{\mathcal{A}}$ that does not shrink with N. This is a limitation of predicting finitely many quantiles, not of in-context inference.

Remark 6 (What is not needed). The bounds require neither identifiability, nor ergodicity, nor queue stability, nor time homogeneity; (R2) is not assumed. Directions of ϑ that do not affect the normalized law cost nothing, because Assumption 2(a) is an upper bound and is trivially satisfied along them. Examples are the absolute time scale, which the normalization removes, and reparametrizations that yield the same process. More precisely, split $\vartheta = (\vartheta_a, \vartheta_b)$ with $p_a = \dim \vartheta_a$, and replace Assumption 2 by the following variant, in which the KL bound depends only on ϑ_a. There exist a neighborhood U_b of $\vartheta_b^\star$, a radius $r_0 > 0$ and a constant $\pi_{a,\min} > 0$ such that, whenever $\vartheta_b \in U_b$ and $\|\vartheta_a - \vartheta_a^\star\| \le r_0$, two conditions hold. First, $D_{\mathrm{KL}}\big(P^{(N)}_{W^\star} \,\|\, P^{(N)}_{(S^\star,\vartheta)}\big) \le LN\|\vartheta_a - \vartheta_a^\star\|^\beta$. Second, the conditional prior density of ϑ_a given ϑ_b is at least $\pi_{a,\min}$. The proof of Theorem 1 then goes through with $A = \{\|\vartheta_a - \vartheta_a^\star\| \le N^{-1/\beta}\} \times U_b$, because $\Pi(A) \ge \Pi(\vartheta_b \in U_b)\, \pi_{a,\min}\, v_{p_a} N^{-p_a/\beta}$. In equation 28, p is replaced by p_a, and $\pi_{\min} v_p$ by $\Pi(\vartheta_b \in U_b)\, \pi_{a,\min}\, v_{p_a}$.

Remark 7 (Sufficient conditions for Assumption 2(a)). Consider the complete record of a run up to the N-th event: every mechanism invocation, the discrete choices it makes (creation counts, successors, cancellations, gate outcomes), and every realized delay. For a fixed structure, H_N is a world-independent function of this record. By data processing and the chain rule, the KL divergence on H_N is therefore at most the expected sum, over the invocations made before the N-th event, of the per-invocation KL divergences.

Each executed event triggers at most $K+1$ invocations, so at most $(K+1)N$ invocations precede the N-th event. Hence (a) holds with $\beta = 2$ whenever each invocation's law, including the delays of the opportunities it creates, is smooth in ϑ, with the expected Hessian of its negative log-likelihood bounded uniformly over states and over the ball $\|\vartheta - \vartheta^\star\| \le r_0$. This holds for the continuous delay families on bounded parameter ranges and for smooth activations.

Components that act *deterministically* through a continuous parameter can violate (a), and they are excluded from the analysis. Deterministic delays and hard gate thresholds can make the KL divergence infinite. Step and modulo activations typically reduce the bound to $\beta = 1$.

Remark 8 (Scope and limitations).

(i) *Continuous delays.* (R1) is essential for a log-loss analysis. Suppose a delay is deterministic with a parameter-dependent value. Before observing it, the Bayes predictor spreads mass continuously over the possible values, while the truth has an atom. The excess risk at that step is then infinite. Proposition 2 itself needs no densities, and quantile predictions remain well defined in this case.

(ii) *Finite contexts and dimension.* EventFM is trained only for $n \le N_{\max} - 1 = 255$. For even moderate K, the raw parameter count $p(S^\star)$ runs to hundreds or thousands, and equation 28 is numerically vacuous at $N \le 255$ unless the effective dimension (Remark 6) is much smaller. The theorem explains the *form* of in-context learning: a structural cost that does not grow with N, a logarithmic parameter cost, and only effective dimensions counting (Remark 6). It does not supply numerical guarantees.

(iii) *Comparator.* The oracle observes the normalized history. Two costs lie outside the bound. The first is the oracle's own irreducible risk, which reflects the hidden queue (Remark 2). The second is the cost of invariance,

$$\mathbb{E}\, D_{\mathrm{KL}}\big(P_{W^\star}(e^\circ_{n+1} \mid H_n) \,\|\, P_{W^\star}(e^\circ_{n+1} \mid X_n)\big) = I\big(e^\circ_{n+1}; H_n \mid X_n, W^\star\big) \ge 0,$$

which is the information about the next normalized event carried by the discarded origin, scale, and mark identities. It need not vanish, for example under nonstationary sources.

(iv) *Realizability and support.* The transfer to EventFM uses Assumption 1, and it concerns histories generated by worlds in the support of the prior. Approximation error and out-of-prior targets are treated in Appendix I.

I MISSPECIFIED TARGETS: PREDICTION OUTSIDE THE PRIOR

This appendix studies prediction on event data that were not generated by EventSS. Let D be an arbitrary law on event sequences, for example the law of sequences in a real dataset. We assume

only that, under D, timestamps are almost surely distinct and $n \geq 2$ events span positive time, so the canonical form X_n of Appendix H is well defined. Write $D^{(N)}$ for the law of X_N under D, and $D(e^\circ_{n+1} \in \cdot \mid X_n)$ for D's own next-event law given the normalized history.

The analysis has three parts. First, an oracle inequality compares the Bayes predictor with the best world in the prior's support, with an explicit estimation cost. Second, we identify when EventFM's outputs on D-histories still equal the Bayes functionals of Proposition 2. Third, we give accuracy guarantees for EventFM's outputs with an explicit approximation term. We make no claim that the posterior concentrates on a best world. The Bayes predictor is a mixture and can outperform every single world.

I.1 Risk and misspecification

Definition 4 (Target risk and misspecification)**.** *The excess risk of the Bayes predictor on D at step $n+1$ is*

$$R_n^D = \mathbb{E}_{X_n \sim D^{(n)}}\Big[D_{\mathrm{KL}}\big(D(e^\circ_{n+1} \in \cdot \mid X_n) \,\big\|\, B_n(\cdot \mid X_n)\big)\Big],$$

and the cumulative excess risk is $C_N^D = \sum_{n=2}^{N-1} R_n^D$. If $D_{\mathrm{KL}}\big(D^{(N)} \,\|\, M^{(N)}\big) = \infty$, the Bayes predictor need not be defined on the support of $D^{(n)}$, and we set $C_N^D = +\infty$. For a world W, the cumulative misspecification *is $\Psi_N(W) = D_{\mathrm{KL}}\big(D^{(N)} \,\|\, P_W^{(N)}\big)$, and the* misspecification gap *is the per-event average $\psi_N(W) = \mathcal{M}_N(W)/(N-2)$.*

By the chain rule, $\Psi_N(W)$ is the expected excess log loss, relative to D's own predictor, of an oracle that knows W and believes it to be true, accumulated over the first N events. The gap $\psi_N(W)$ is zero when $D = P_W$. If D lies outside the model, then $\psi_N(W)$ is typically bounded away from zero for every W, however large N becomes.

Lemma 4 (Redundancy identity under misspecification)**.** *For every $N \geq 3$ with $D_{\mathrm{KL}}(D^{(N)} \| M^{(N)}) < \infty$,*

$$C_N^D = D_{\mathrm{KL}}\big(D^{(N)} \,\|\, M^{(N)}\big) - D_{\mathrm{KL}}\big(D^{(2)} \,\|\, M^{(2)}\big) \leq D_{\mathrm{KL}}\big(D^{(N)} \,\|\, M^{(N)}\big).$$

Proof. The proof of Lemma 2 never uses that the first argument is a member of the model family. It therefore applies verbatim with D in place of $P_{W^\star}$. □

I.2 Oracle inequality

Theorem 2 (Oracle inequality)**.** *Let $S^\dagger$ be any structure with $P(S^\dagger) > 0$, and let $A \subseteq \Theta_{S^\dagger}$ be any set with $\Pi(A) > 0$. Then for every $N \geq 3$,*

$$C_N^D \leq \log \frac{1}{P(S^\dagger)} + \log \frac{1}{\Pi(A)} + \sup_{\vartheta \in A} \mathcal{M}_N\big((S^\dagger, \vartheta)\big). \tag{29}$$

Proof. If the right-hand side is infinite there is nothing to prove; otherwise Lemma 3 shows $D_{\mathrm{KL}}(D^{(N)} \| M^{(N)}) < \infty$, so Lemma 4 applies. Combine Lemma 4 with Lemma 3. The proof of the latter uses only the measure inequality $M^{(N)} \geq P(S^\dagger)\,\Pi(A)\,M_A$ and convexity of the KL divergence, so it holds with D in place of $P_{W^\star}$. □

The bound holds for every choice of $(S^\dagger, A)$, so the Bayes predictor competes with the best trade-off between how well a region of worlds fits D and how much prior mass that region receives. To obtain a rate, we localize around a comparator world.

Assumption 3 (Local regularity at a comparator)**.** *Let $W^\dagger = (S^\dagger, \vartheta^\dagger)$, and suppose $\vartheta^\dagger$ lies in the interior of $\Theta_{S^\dagger} \subseteq \mathbb{R}^p$, where $p = p(S^\dagger)$. There exist $r_0 > 0$, $L < \infty$, $\beta \in (0, 2]$ and $\pi_{\min} > 0$ such that for every ϑ with $\|\vartheta - \vartheta^\dagger\| \leq r_0$ the following hold.*

(a) For all $N \geq 3$, $\Psi_N\big((S^\dagger, \vartheta)\big) \leq \Psi_N(W^\dagger) + L\,N\,\|\vartheta - \vartheta^\dagger\|^\beta$.

(b) $\vartheta \mid S^\dagger$ has a Lebesgue density of at least $\pi_{\min}$ at ϑ.

When $D = P_{W^\star}$ and $W^\dagger = W^\star$, the first term on the right of (a) vanishes, and Assumption 3 reduces to Assumption 2. If $\vartheta^\dagger$ is a minimizer for all N (or of the per-event limit) and the dependence is smooth, the first-order term vanishes and $\beta = 2$ is natural. For an arbitrary comparator, a Lipschitz bound with $\beta = 1$ is the natural condition.

Corollary 4 (Rate under misspecification)**.** *Suppose Assumption 3 holds. Then for all* $N \geq \max(3, r_0^{-\beta})$,

$$\frac{C_N^D}{N-2} \;\leq\; \underbrace{\psi_N(W^\dagger)}_{\textit{misspecification}} \;+\; \underbrace{\frac{\log \frac{1}{P(S^\dagger)} + \frac{p}{\beta}\log N + L + \log \frac{1}{\pi_{\min} v_p}}{N-2}}_{\textit{in-context estimation}}. \tag{30}$$

In particular, the average excess risk of the Bayes predictor exceeds the comparator's misspecification gap by at most $O(p \log N/N)$.

Proof. Apply Theorem 2 with A the ball of radius $N^{-1/\beta}$ around $\vartheta^\dagger$, exactly as in the proof of Theorem 1, and divide by $N-2$. □

I.3 What EventFM computes on target histories

Proposition 2 identifies EventFM's outputs with the Bayes functionals only almost everywhere under the training law. A priori this says nothing about histories from D. The next result shows that the oracle inequality closes this gap whenever it is informative.

Proposition 4 (Target histories lie in the training support)**.** *Suppose* $D_{\mathrm{KL}}(D^{(N)} \,\|\, M^{(N)}) < \infty$. *This holds, for example, whenever the right-hand side of equation 29 is finite for some* $(S^\dagger, A)$. *Then* $D^{(n)} \ll M^{(n)}$ *for every* $2 \leq n \leq N$. *Consequently, under Assumption 1, the conclusions of Proposition 2 hold for* D*-almost every history* X_n *with* $n \leq \min(N, N_{\max} - 1)$*: EventFM's mark probabilities and quantiles on target histories are the Bayes functionals.*

Proof. Finite KL divergence implies absolute continuity, so $D^{(N)} \ll M^{(N)}$. The map $X_N \mapsto X_n$ is measurable (Lemma 1), so absolute continuity is preserved under marginalization. Since N is drawn independently of W, the training law of the context at length n is $M^{(n)}$, so a property that holds $M^{(n)}$-almost everywhere holds $D^{(n)}$-almost everywhere. □

Proposition 4 is an idealized statement. It assumes an exact minimizer. In practice, the trained network differs from the Bayes functionals, and this difference is controlled only on average under the training law. We therefore state accuracy guarantees with explicit approximation terms. Let θ be a trained model, and let $\theta^\star$ denote the idealized minimizer of Proposition 2. For a history H_n with canonical form X_n, define the approximation errors

$$\varepsilon_\kappa(X_n) = \big\| p_\theta(\cdot \mid H_n) - p_{\theta^\star}(\cdot \mid H_n) \big\|_{\mathrm{TV}}, \qquad \varepsilon_F(X_n) = \sup_x \big| \hat{F}_\theta(x \mid H_n) - \hat{F}_{\theta^\star}(x \mid H_n) \big|.$$

Here $\hat{F}$ is the decoded marginal CDF of Appendix G. Both errors lie in $[0,1]$. Define also $a_D(X_n)$, the D-probability that the next mark is already in the context; F^D, D's distribution function of δ_{n+1} restricted to marks in the context; and $\mathrm{TV}_n^D(X_n)$, the total-variation distance between $D(e^\circ_{n+1} \in \cdot \mid X_n)$ and $B_n(\cdot \mid X_n)$.

Proposition 5 (Accuracy of EventFM's outputs on target data)**.** *Suppose the conclusion of Proposition 4 holds, and let* $2 \leq n \leq \min(N, N_{\max} - 1)$. *Then for* D*-almost every* X_n *with* $a_D(X_n) > 0$ *the following hold.*

(a) Marks. $\big\| p_\theta(\cdot \mid H_n) - D(\cdot \mid X_n, \kappa_{n+1} \in \mathcal{K}_n) \big\|_{\mathrm{TV}} \leq \varepsilon_\kappa(X_n) + 2\,\mathrm{TV}_n^D(X_n)/a_D(X_n)$.

(b) Marginal median. *EventFM's normalized marginal median* $\hat{m}_n$ *satisfies* $F^D(\hat{m}_n^-) \leq \frac{1}{2} + \varepsilon_n$ *and* $F^D(\hat{m}_n) \geq \frac{1}{2} - \varepsilon_n$*, where*

$$\varepsilon_n = \Delta_{\mathcal{A}} + \varepsilon_F(X_n) + \frac{2\,\mathrm{TV}_n^D(X_n)}{a_D(X_n)}.$$

(c) Averages.

$$\frac{1}{N-2}\sum_{n=2}^{N-1}\mathbb{E}_D\big[\mathrm{TV}_n^D\big] \le \sqrt{\frac{C_N^D}{2(N-2)}},$$

which under Assumption 3 is at most $\sqrt{\psi_N(W^\dagger)/2 + O(p\log N/N)}$.

Proof. Parts (a) and (b) repeat the arguments of Proposition 3(a) and (d), with D in place of the oracle. Those arguments use only that the comparator is a probability law for e°_{n+1} given X_n. Each part adds one triangle-inequality step for the approximation error. In (b), $\hat{F}_\theta$ is continuous, so $\hat{F}_\theta(\hat{m}_n) = 1/2$. Then $\|\hat{F}_\theta - F^D\|_\infty \le \varepsilon_F + \Delta_{\mathcal{A}} + 2\,\mathrm{TV}_n^D/a_D$. Part (c) repeats the Pinsker and Jensen argument of Proposition 3(c), followed by Corollary 4. □

I.4 Interpretation and limitations

Remark 9 (Three sources of error). Corollary 4 and Proposition 5 separate the error of EventFM on non-EventSS data into three parts.

- *Misspecification*, $\psi_N(W^\dagger)$. This is fixed by the prior. It does not shrink with longer contexts or better training, and in part (c) it enters under a square root, as an error floor. Reducing it requires changing EventSS so that its support contains worlds closer to D.
- *In-context estimation*, $O\big((\log(1/P(S^\dagger)) + p\log N)/N\big)$. This is the same cost as in Theorem 1, now paid for the comparator.
- *Approximation*, ε_κ and ε_F. This reflects finite capacity and finite training.

Because equation 29 holds for every comparator, the Bayes predictor behaves like a minimum-description-length procedure (Grünwald, 2007). On short contexts it competes with simple worlds of high prior mass. As N grows, it competes with better-fitting worlds that are a priori rarer. This gives a precise sense in which prior design governs out-of-distribution behavior, and it is consistent with the prior-sensitivity results of Table 5.

Remark 10 (Approximation error off the prior). Training controls ε_κ and ε_F only on average under $M^{(n)}$, not under $D^{(n)}$. For any f with values in $[0,1]$, Cauchy–Schwarz gives

$$\mathbb{E}_D[f(X_n)] \le \sqrt{\big(1+\chi^2(D^{(n)}\,\|\,M^{(n)})\big)\,\mathbb{E}_M[f(X_n)]},$$

where χ^2 is the chi-squared divergence. Since $1+\chi^2 \ge \exp D_{\mathrm{KL}}$, the amplification factor is at least $\exp\big(D_{\mathrm{KL}}(D^{(n)}\|M^{(n)})/2\big)$. It typically grows with n when D is misspecified. In-distribution accuracy therefore implies little about accuracy on real data. EventFM's out-of-distribution performance rests on the misspecification analysis above together with the inductive biases of the architecture, and the empirical results are what establish it.

Remark 11 (Scope).

(i) If D assigns positive probability to events that every world in the prior's support rules out, then $\psi_N(W) = \infty$ for all W and the bounds are vacuous. Examples are marks beyond the vocabulary or deterministic regularities that the prior can represent only with continuous delays. Proposition 4 then does not apply, and EventFM's behavior on such histories is not constrained by training. The analysis does not apply for simultaneous events (datasets with tied timestamps) as we assume distinct timestamps in several results and proofs.

(ii) The comparator in Definition 4 is D's predictor given the *normalized* history and canonical labels. If absolute time carries information under D, for example time-of-day effects, or true label identities carry information, the cost of discarding this information, $I\big(e^\circ_{n+1}; H_n \mid X_n\big)$ under D, is additional to these bounds.

(iii) D is the law of the dataset's sequences, and all bounds are in D-expectation. They say nothing about any individual sequence.

(iv) All statements about EventFM's outputs require $n \le N_{\max} - 1 = 255$. For longer real sequences, they apply only if the context is truncated to at most 255 events.

J EVENT-SEQUENCE FEATURE SPACE AND GENERATOR COVERAGE

J.1 COVERAGE RESULTS

We compare EventSS with an additive intensity-kernel prior in a common, scale-invariant feature space. From each sequence, we draw one prefix, normalize its time axis by the observed span, and compute 32 features describing timing, mark dynamics, and type–time dependence. Features are standardized using medians and interquartile ranges fit on the pooled real data. Each comparison uses 400 real sources per corpus and an equal number of synthetic sequences, repeated 30 times.

Coverage is measured locally and globally. For each real sequence, local coverage is summarized by the distance to its nearest synthetic neighbour and by the number of same-corpus real neighbours that are closer. Global coverage is measured by energy distance, normalized by the mean pairwise distance within the real corpus. The UMAP in Figure 3 is fit only on real sequences and is used for visualization; all reported statistics are computed in the standardized feature space.

Table 20 shows that EventSS has a lower macro-average median rank (13.2 versus 19.0), a lower mean nearest-neighbour distance (4.267 versus 4.664), and a lower normalized energy distance (0.141 versus 0.211). EventSS is closer for 69.4% of the 2000 paired real sources. Its normalized energy distance is lower on every corpus. The kernel prior has slightly lower median rank on Amazon and StackOverflow, and the raw-distance difference on StackOverflow is not significant; thus, the advantage is in overall distributional and tail coverage rather than every local summary.

Table 20: **Local and global coverage of EventSS versus the additive intensity-kernel prior.** *Rank*: number of same-corpus real neighbours closer than the nearest synthetic sample. *Mean distance*: paired standardized distance over 400 sources per corpus. * EventSS is closer by t-, Wilcoxon-, and sign tests at $p < 10^{-3}$; ‡ Amazon by the first two only; † StackOverflow shows no significant difference. e: energy distance normalized by the real-set mean pairwise distance.

	prior	Taxi	Amazon	Retweet	StackOv.	Taobao	macro
local: median rank ↓	EventSS	**19.5**	24.3	**13.2**	5.2	**4.0**	**13.2**
	kernel prior	42.5	**22.9**	19.9	**4.5**	5.1	19.0
local: mean distance ↓	EventSS	**3.407***	**6.693**‡	**4.052***	3.278†	**3.903***	**4.267***
	kernel prior	3.923	7.210	4.824	3.306	4.056	4.664
global: e ↓	EventSS	**0.191**	**0.149**	**0.175**	**0.087**	**0.104**	**0.141**
	kernel prior	0.321	0.216	0.217	0.140	0.161	0.211

J.2 PREFIX SAMPLING AND NORMALIZATION

Sampling. For each source sequence of n events, we draw one context length uniformly from $\{5, \dots, n\}$. Sequences with fewer than five events, non-finite timestamps, or negative gaps are excluded; this removes 10 of the 2000 EventSS sources and no real sources. When several prefixes are used in the paired analysis, their distances are averaged within source so that the source remains the unit of analysis.

Time normalization. Each prefix is shifted to start at zero and divided by its observed span, mapping its timestamps to $[0, 1]$. The resulting features therefore describe temporal shape rather than absolute duration or time units.

Mark permutation and cardinality. Mark features depend on counts, transitions, and type-conditioned timing, but not on mark identities, so the representation is invariant to relabeling. Raw mark count is not included, although effective type count remains a feature; the representation is therefore relabeling-invariant but not fully invariant to mark cardinality.

Standardization and estimation. Each feature is standardized as $(x_j - \mathrm{median}_j)/\,\mathrm{IQR}_j$, with both statistics estimated from pooled real sequences. A zero IQR is replaced by one, and undefined values are imputed with the real-data median. Macro values are unweighted means over the five corpora; Monte Carlo intervals are percentiles over the 30 replicates.

J.3 EVENT-SEQUENCE FEATURE REPRESENTATION

The representation contains 32 closed-form statistics in four groups: timing, counting, marks, and type–time coupling (Table 21). Candidate features were screened using real data only. We removed near-constant features, one member of pairs with Spearman correlation above 0.8 both pooled and within every corpus, and features with rank-based variance inflation factor at least 10 under the same criterion.

Table 21: The 32 features in their fixed matrix-column order. A *gap* is an inter-event time on the normalized axis of Appendix J.2, making all timing features unitless. A *type* is a mark. Shuffle-null adjustments subtract the same statistic after permuting the sequence's marks to reduce short-sequence bias.

Feature	**What It Measures**
Timing (11)	*Gap distribution and order, independent of marks*
Log-gap dispersion	standard deviation of log inter-event gaps
Upper-decile gap ratio	90th-percentile gap divided by the median gap
Minimum-to-median gap ratio	minimum gap divided by the median gap
Gap autocorrelation, lag 1	serial dependence of consecutive gaps
Gap autocorrelation, lag 2	serial dependence at lag 2
Gap autocorrelation, lag 3	serial dependence at lag 3
Local variation	mean squared normalized difference between consecutive gaps
Gap skewness	asymmetry of the gap distribution
Gap excess kurtosis	excess tail weight relative to a Gaussian
Gap–index rank correlation	Spearman correlation between event index and gap
Terminal-gap surprise	final gap standardized by the preceding gaps
Counting (6)	*Nonstationarity and periodicity of arrivals*
Second-half rate ratio	event-rate ratio between the second and first halves
Changepoint statistic	normalized Pettitt statistic for a level shift in the gap series
Periodogram peak	peak of the Rayleigh point-process periodogram in $[0, 1]$
Dominant gap frequency	strongest Hann-windowed gap frequency in cycles per event
Spectral concentration	share of gap-series power at the dominant frequency
Seasonal-period power	largest power share over the predefined seasonal periods
Marks (9)	*Type usage and type-to-type dependence, independent of type identity*
Normalized type entropy	Chao–Shen entropy divided by $\log$(number of types)
Effective type count	exponentiated type entropy
Type-usage Gini	Gini coefficient of the type-count vector
Dominant-type share	share of events with the most frequent type
Conditional next-type entropy	$H(\text{next type} \mid \text{current type})$ in nats
Excess lag-1 type information	mutual information between consecutive types, minus its shuffle null
Excess lag-2 type information	conditional lag-2 type information, minus its shuffle null
Excess lag-3 type information	the same at lag 3, beyond the two previous types
Self-transition rate	fraction of consecutive pairs that share a type
Type–Timing Coupling (6)	*Timing explained by types through one-way η^2 on log gaps*
Preceding-type timing effect	share of log-gap variance explained by the *preceding* type
Excess over shuffle	preceding-type share minus its shuffle null
Arriving-type timing effect	share explained by the *arriving* type
Excess over shuffle	arriving-type share minus its shuffle null
Type-pair timing effect	share explained by the ordered (previous, arriving) type pair
Excess over shuffle	pair share minus its shuffle null

J.4 DISTRIBUTION-COMPARISON METRICS

Let $X = \{x_i\}_{i=1}^m$ denote a real sample, $Y = \{y_j\}_{j=1}^m$ a synthetic sample, and d Euclidean distance in the standardized feature space. For each real point, define its nearest-synthetic distance $d_i^{\text{syn}} = \min_j d(x_i, y_j)$ and rank

$$c_i = \sum_{i' \neq i} \mathbf{1}\{d(x_i, x_{i'}) < d_i^{\text{syn}}\}\,.$$

We report the median rank and mean nearest-synthetic distance within each corpus. Smaller values indicate that synthetic sequences lie closer to real sequences.

Global agreement is measured by normalized energy distance. Writing $\bar{d}_{XY}$, $\bar{d}_{XX'}$, and $\bar{d}_{YY'}$ for mean pairwise distances, we use

$$e = \frac{2\bar{d}_{XY} - \bar{d}_{XX'} - \bar{d}_{YY'}}{\bar{d}_{XX'}}.$$

This statistic is zero when the real and synthetic feature distributions agree.

J.5 The intensity-kernel prior

The comparison prior samples additive cross-mark excitation and inhibition kernels and generates sequences by thinning. Table 22 summarizes the sequence-shape supports used in the reported comparison.

Table 22: Sequence-shape support used for the coverage comparison.

Quantity	**EventSS**	**Intensity-kernel prior**
Events per sequence	3–256	50–500
Marks per sequence	2–30	2–10

The reported kernel configuration is the best-performing of three evaluated sequence-shape settings. Because effective mark count is one of the features, we also repeated the comparison after matching the kernel prior to EventSS's event and mark supports. The matched-support kernel prior attained macro median rank 22.4 and normalized energy distance 0.240, compared with 13.2 and 0.141 for EventSS. Thus, matching sequence length and mark-count support did not explain the observed coverage difference.

J.6 Local and global coverage

Table 23 separates typical local coverage, tail coverage, and global distributional agreement. The tail statistic $\Pr[c \geq 40]$ is the fraction of real sequences for which at least 10% of the same-corpus real sample is closer than the nearest synthetic sequence.

Table 23: Local and global coverage. $\Pr[c \geq 40]$ is the fraction of real points more than a tenth of the real corpus away from any synthetic point. EventSS attains the lower e in every corpus, its Monte-Carlo 95% interval disjoint from the kernel prior's on Taxi, StackOverflow and Taobao; the intervals overlap on Amazon (0.149 [0.108, 0.190] against 0.216 [0.155, 0.271]) and Retweet (0.175 [0.148, 0.202] against 0.217 [0.185, 0.240]).

	prior	Taxi	Amazon	Retweet	StackOv.	Taobao	macro
local:	EventSS	**19.5**	24.3	**13.2**	5.2	**4.0**	**13.2**
med. $c\downarrow$	kernel prior	42.5	**22.9**	19.9	**4.5**	5.1	19.0
local:	EventSS	**0.216**	**0.283**	**0.194**	**0.016**	**0.008**	**0.143**
$\Pr[c \geq 40]\downarrow$	kernel prior	0.525	0.330	0.339	0.055	0.051	0.260
global:	EventSS	**0.191**	**0.149**	**0.175**	**0.087**	**0.104**	**0.141**
$e\downarrow$	kernel prior	0.321	0.216	0.217	0.140	0.161	0.211

EventSS has lower tail failure probability and lower normalized energy distance on all five corpora. The kernel prior has slightly lower median rank on Amazon and StackOverflow, but its energy distance is higher on both. The comparison therefore supports broader aggregate coverage by EventSS while retaining two local exceptions.

J.7 PAIRED SIGNIFICANCE TEST ON THE RAW DISTANCE

We test the raw nearest-synthetic distances using paired real sources. Synthetic pools are truncated to the common size of 1986, and distances from multiple prefixes are averaged within source. This gives 400 independent observations per corpus and 2000 pooled observations. Table 24 reports a one-sided paired t-test; Wilcoxon signed-rank and sign tests are used as distribution-free checks.

Table 24: Paired test of nearest-synthetic distance over 2000 real sources. Δ is EventSS minus the intensity-kernel prior, so negative values favor EventSS. Win rate is the fraction of sources for which EventSS is closer.

prior	mean d	Δ	95% CI	t	p	win	geo. ratio
kernel prior	4.664	-0.397	$[-0.443, -0.351]$	-16.98	8×10^{-61}	0.694	0.924

EventSS reduces mean nearest-synthetic distance by 0.397 (95% CI $[-0.443, -0.351]$), with a paired effect size of $d_z = -0.38$ and a 69.4% win rate. The difference is strongest on Taxi, Retweet, and Taobao. On Amazon, the mean difference favors EventSS but the win rate is 57%. On StackOverflow, none of the three paired tests finds a significant difference.

K Full Synthetic Evaluation

K.1 In-Distribution Reconstruction

Table 25 reports the in-distribution reconstruction results: the frozen EventFM against the two naive zero-shot baselines on held-out sequences from the EventSS training mixture. EventFM is significantly best on every metric except sequence OTD, where it ties mean/mode and significantly beats last.

Table 25: In-distribution EventSS reconstruction (1000 held-out training-mixture sequences of 64 events each). EventFM versus the naive zero-shot baselines. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Best point estimates are bold and second-best point estimates are underlined.

Model	Acc. ↑	Time RMSE ↓	CRPS ↓	Brier ↓	Seq. OTD ↓	Count RMSE ↓
EventFM	**0.593** [0.582,0.603]	**28.80** [22.88,36.82]	**3.239** [2.801,3.867]	**0.528** [0.517,0.539]	129.9 [43.78,382.1]	**1.263** [1.062,1.477]
Mean/mode	0.533 [0.521,0.544]	29.24 [23.40,37.24]	4.342 [3.799,5.100]	0.588 [0.578,0.599]	**123.2** [44.48,340.1]	1.469 [1.258,1.697]
Last	0.435 [0.424,0.446]	38.42 [32.39,46.65]	5.913 [5.175,6.877]	1.130 [1.108,1.152]	191.8 [64.95,530.1]	2.826 [2.445,3.208]

K.2 OOD EventSS Shifts

The out-of-distribution results compare the frozen EventFM against the two naive zero-shot baselines on EventSS sequences shifted along the four controlled levers, one table per lever, each including the in-distribution reference for comparison. Table 26 (event-type cardinality, exact-K worlds with $K \in \{7, 8, 10\}$) and Table 27 (temporal scale, $a \in \{0.01, 0.1, 10, 100\}$) cover the two scaling levers: EventFM stays more accurate than both baselines at every K, and its accuracy and Brier are essentially constant as the time scale changes, confirming that its predictions are calibrated in relative rather than absolute time, while its raw-time CRPS scales with a by construction. Table 28 (unseen delay-law families) and Table 29 (mechanism compositions) cover the two structural levers: EventFM is best on type accuracy and CRPS throughout, while on sequence OTD it is on par with the mean/mode baseline and ahead of last. Because the raw-time OTD depends on each setting's time magnitude, it should be compared within a column rather than across columns.

Table 26: EventSS shift to higher event-type cardinality via exact-K worlds (the world contains exactly K event types), with the in-distribution reference. $K \in \{7, 8, 10\}$ is $1.3\times$ / $1.6\times$ / $2\times$ the in-distribution realized mark count. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Best point estimates are bold and second-best point estimates are underlined.

Model	In-dist.	7	8	10
		Type accuracy ↑		
EventFM	**0.593** [0.582,0.603]	**0.526** [0.515,0.538]	**0.489** [0.478,0.501]	**0.458** [0.447,0.469]
Mean/mode	0.533 [0.521,0.544]	0.349 [0.338,0.359]	0.296 [0.286,0.306]	0.260 [0.251,0.270]
Last	0.435 [0.424,0.446]	0.246 [0.236,0.256]	0.224 [0.215,0.233]	0.197 [0.188,0.206]
		Time RMSE ↓		
EventFM	**28.80** [22.88,36.82]	22.13 [16.78,33.40]	**11.74** [8.809,16.71]	12.41 [6.799,21.69]
Mean/mode	29.24 [23.40,37.24]	**21.32** [16.21,32.69]	12.40 [9.429,17.83]	**12.12** [6.653,21.31]
Last	38.42 [32.39,46.65]	31.25 [25.42,40.83]	15.89 [12.60,20.39]	15.96 [10.53,23.81]
		CRPS ↓		
EventFM	**3.239** [2.801,3.867]	**2.101** [1.762,2.641]	**1.385** [1.206,1.674]	**1.124** [0.945,1.501]
Mean/mode	4.342 [3.799,5.100]	2.824 [2.434,3.420]	1.984 [1.765,2.329]	1.538 [1.343,1.939]
Last	5.913 [5.175,6.877]	3.979 [3.368,4.778]	2.444 [2.146,2.860]	1.917 [1.647,2.401]
		Brier ↓		
EventFM	**0.528** [0.517,0.539]	**0.621** [0.611,0.631]	**0.661** [0.651,0.671]	**0.693** [0.684,0.703]
Mean/mode	0.588 [0.578,0.599]	0.752 [0.744,0.760]	0.803 [0.796,0.810]	0.832 [0.825,0.839]
Last	1.130 [1.108,1.152]	1.508 [1.489,1.527]	1.551 [1.533,1.570]	1.606 [1.588,1.623]
		Sequence OTD ↓		
EventFM	129.9 [43.78,382.1]	81.76 [30.38,312.0]	**27.07** [19.59,37.97]	26.49 [20.86,33.67]
Mean/mode	**123.2** [44.48,340.1]	**64.31** [30.72,202.6]	31.24 [22.26,53.32]	**25.96** [20.96,32.20]
Last	191.8 [64.95,530.1]	130.8 [41.65,513.0]	35.71 [26.03,52.06]	31.28 [25.08,39.64]
		Count RMSE ↓		
EventFM	**1.263** [1.062,1.477]	**1.303** [1.117,1.505]	**1.222** [1.020,1.461]	**1.265** [1.112,1.424]
Mean/mode	1.469 [1.258,1.697]	1.558 [1.393,1.738]	1.501 [1.342,1.675]	1.464 [1.337,1.602]
Last	2.826 [2.445,3.208]	3.259 [2.944,3.516]	3.091 [2.747,3.397]	2.905 [2.577,3.166]

Table 27: EventSS shift along temporal scale a, with the in-distribution reference (a=1). Accuracy is scale-invariant, while raw-time metrics scale with a. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Best point estimates are bold and second-best point estimates are underlined.

Model	In-dist.	0.01	0.1	10	100
			Type accuracy ↑		
EventFM	**0.593** [0.582,0.603]	**0.594** [0.583,0.605]	**0.595** [0.584,0.606]	**0.594** [0.583,0.605]	**0.594** [0.583,0.605]
Mean/mode	0.533 [0.521,0.544]	0.533 [0.521,0.544]	0.533 [0.521,0.544]	0.533 [0.521,0.544]	0.533 [0.521,0.544]
Last	0.435 [0.424,0.446]	0.435 [0.424,0.446]	0.435 [0.424,0.446]	0.435 [0.424,0.446]	0.435 [0.424,0.446]
			Time RMSE ↓		
EventFM	**28.80** [22.88,36.82]	**0.288** [0.229,0.368]	**2.880** [2.289,3.682]	**288.0** [228.8,368.0]	**2.9k** [2.3k,3.7k]
Mean/mode	29.24 [23.40,37.24]	0.292 [0.234,0.372]	2.924 [2.340,3.724]	292.4 [234.0,372.4]	2.9k [2.3k,3.7k]
Last	38.42 [32.39,46.65]	0.384 [0.324,0.467]	3.842 [3.239,4.665]	384.2 [323.9,466.5]	3.8k [3.2k,4.7k]
			CRPS ↓		
EventFM	**3.239** [2.801,3.867]	**0.033** [0.028,0.039]	**0.324** [0.281,0.387]	**32.40** [28.03,38.68]	**324.1** [280.4,387.1]
Mean/mode	4.342 [3.799,5.100]	0.043 [0.038,0.051]	0.434 [0.380,0.510]	43.42 [37.99,51.00]	434.2 [379.9,510.0]
Last	5.913 [5.175,6.877]	0.059 [0.052,0.069]	0.591 [0.517,0.688]	59.13 [51.75,68.77]	591.3 [517.5,687.7]
			Brier ↓		
EventFM	**0.528** [0.517,0.539]	**0.528** [0.518,0.539]	**0.528** [0.518,0.539]	**0.528** [0.518,0.539]	**0.528** [0.517,0.539]
Mean/mode	0.588 [0.578,0.599]	0.588 [0.578,0.599]	0.588 [0.578,0.599]	0.588 [0.578,0.599]	0.588 [0.578,0.599]
Last	1.130 [1.108,1.152]	1.130 [1.108,1.152]	1.130 [1.108,1.152]	1.130 [1.108,1.152]	1.130 [1.108,1.152]
			Sequence OTD ↓		
EventFM	129.9 [43.78,382.1]	1.297 [0.437,3.823]	12.95 [4.342,38.15]	1.3k [433.7,3.8k]	12.9k [4.3k,38.1k]
Mean/mode	**123.2** [44.48,340.1]	**1.232** [0.445,3.401]	**12.32** [4.448,34.01]	**1.2k** [444.8,3.4k]	**12.3k** [4.4k,34.0k]
Last	191.8 [64.95,530.1]	1.918 [0.650,5.301]	19.18 [6.495,53.01]	1.9k [649.5,5.3k]	19.2k [6.5k,53.0k]
			Count RMSE ↓		
EventFM	**1.263** [1.062,1.477]	**1.249** [1.039,1.462]	**1.228** [1.032,1.441]	**1.224** [1.022,1.438]	**1.244** [1.047,1.452]
Mean/mode	1.469 [1.258,1.697]	1.469 [1.258,1.697]	1.469 [1.258,1.697]	1.469 [1.258,1.697]	1.469 [1.258,1.697]
Last	2.826 [2.445,3.208]	2.826 [2.445,3.208]	2.826 [2.445,3.208]	2.826 [2.445,3.208]	2.826 [2.445,3.208]

Table 28: EventSS shift to unseen delay-law families, with the in-distribution reference. Raw-time metrics not comparable across families. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Best point estimates are bold and second-best point estimates are underlined.

Model	In-dist.	Pareto	Log-log.	Triang.	Mixed
			Type accuracy ↑		
EventFM	**0.593** [0.582,0.603]	**0.559** [0.548,0.570]	**0.561** [0.550,0.571]	**0.596** [0.585,0.607]	**0.600** [0.589,0.611]
Mean/mode	0.533 [0.521,0.544]	0.503 [0.492,0.514]	0.512 [0.500,0.523]	0.512 [0.501,0.523]	0.542 [0.530,0.553]
Last	0.435 [0.424,0.446]	0.431 [0.420,0.442]	0.416 [0.405,0.427]	0.419 [0.407,0.430]	0.455 [0.444,0.466]
			Time RMSE ↓		
EventFM	**28.80** [22.88,36.82]	**59.72** [38.16,102.7]	32.58 [19.18,53.39]	**28.66** [22.70,36.72]	**29.16** [24.45,35.55]
Mean/mode	29.24 [23.40,37.24]	59.92 [38.99,100.8]	**32.54** [19.25,53.16]	29.09 [23.23,37.13]	30.16 [25.10,37.01]
Last	38.42 [32.39,46.65]	76.67 [56.09,112.0]	45.21 [31.94,63.96]	38.23 [32.17,46.49]	36.05 [31.39,42.26]
			CRPS ↓		
EventFM	**3.239** [2.801,3.867]	**4.251** [3.403,5.977]	**3.775** [3.280,4.762]	**2.537** [2.099,3.166]	**3.544** [3.082,4.148]
Mean/mode	4.342 [3.799,5.100]	5.502 [4.470,7.331]	5.171 [4.632,6.148]	3.376 [2.834,4.126]	4.640 [4.045,5.412]
Last	5.913 [5.175,6.877]	7.503 [6.064,9.666]	6.725 [5.912,7.974]	4.667 [3.927,5.633]	5.946 [5.223,6.833]
			Brier ↓		
EventFM	**0.528** [0.517,0.539]	**0.563** [0.553,0.574]	**0.559** [0.549,0.570]	**0.526** [0.515,0.537]	**0.516** [0.505,0.527]
Mean/mode	0.588 [0.578,0.599]	0.597 [0.587,0.608]	0.604 [0.594,0.614]	0.595 [0.585,0.606]	0.569 [0.559,0.580]
Last	1.130 [1.108,1.152]	1.138 [1.116,1.160]	1.167 [1.145,1.190]	1.163 [1.141,1.185]	1.091 [1.069,1.113]
			Sequence OTD ↓		
EventFM	129.9 [43.78,382.1]	**66.68** [33.58,150.4]	108.8 [61.44,243.5]	115.6 [29.30,370.8]	**128.3** [54.53,302.7]
Mean/mode	**123.2** [44.48,340.1]	104.2 [44.26,256.3]	**98.24** [59.13,211.6]	**109.0** [30.31,326.7]	151.8 [61.24,381.9]
Last	191.8 [64.95,530.1]	132.8 [54.95,356.1]	156.2 [76.52,415.5]	170.8 [42.85,510.6]	163.1 [66.23,385.3]
			Count RMSE ↓		
EventFM	**1.263** [1.062,1.477]	**1.347** [1.125,1.570]	**1.676** [1.398,2.003]	**1.288** [1.108,1.496]	**1.359** [1.134,1.611]
Mean/mode	1.469 [1.258,1.697]	1.396 [1.159,1.639]	1.808 [1.540,2.108]	1.489 [1.277,1.726]	1.450 [1.257,1.684]
Last	2.826 [2.445,3.208]	2.978 [2.532,3.441]	3.058 [2.601,3.517]	2.871 [2.494,3.248]	2.551 [2.157,2.957]

Table 29: EventSS shift to enriched mechanism compositions, with the in-distribution reference. Entries report point estimates and sequence-bootstrap 95% confidence intervals. Best point estimates are bold and second-best point estimates are underlined.

Model	In-dist.	Excitatory	Many-to-one
	Type accuracy ↑		
EventFM	**0.593** [0.582,0.603]	**0.573** [0.562,0.584]	**0.634** [0.623,0.644]
Mean/mode	<u>0.533</u> [0.521,0.544]	<u>0.522</u> [0.511,0.533]	<u>0.597</u> [0.586,0.607]
Last	0.435 [0.424,0.446]	0.420 [0.409,0.431]	0.507 [0.496,0.518]
	Time RMSE ↓		
EventFM	**28.80** [22.88,36.82]	**21.12** [13.12,34.37]	**35.65** [28.51,45.39]
Mean/mode	<u>29.24</u> [23.40,37.24]	<u>21.96</u> [13.65,35.20]	<u>35.98</u> [28.94,45.69]
Last	38.42 [32.39,46.65]	29.07 [21.28,39.33]	43.51 [36.39,52.76]
	CRPS ↓		
EventFM	**3.239** [2.801,3.867]	**2.075** [1.767,2.661]	**3.698** [3.153,4.444]
Mean/mode	<u>4.342</u> [3.799,5.100]	<u>2.756</u> [2.385,3.433]	<u>4.790</u> [4.096,5.710]
Last	5.913 [5.175,6.877]	3.800 [3.268,4.603]	6.198 [5.349,7.265]
	Brier ↓		
EventFM	**0.528** [0.517,0.539]	**0.545** [0.535,0.556]	**0.465** [0.455,0.476]
Mean/mode	<u>0.588</u> [0.578,0.599]	<u>0.591</u> [0.581,0.601]	<u>0.500</u> [0.490,0.510]
Last	1.130 [1.108,1.152]	1.160 [1.138,1.182]	0.987 [0.965,1.009]
	Sequence OTD ↓		
EventFM	<u>129.9</u> [43.78,382.1]	<u>38.22</u> [28.82,54.01]	<u>52.56</u> [29.96,133.4]
Mean/mode	**123.2** [44.48,340.1]	**37.11** [28.69,50.01]	**50.20** [29.51,123.7]
Last	191.8 [64.95,530.1]	49.11 [37.72,64.41]	74.36 [39.32,204.4]
	Count RMSE ↓		
EventFM	**1.263** [1.062,1.477]	**1.177** [0.983,1.384]	**1.403** [1.175,1.713]
Mean/mode	<u>1.469</u> [1.258,1.697]	<u>1.425</u> [1.202,1.738]	<u>1.482</u> [1.243,1.768]
Last	2.826 [2.445,3.208]	2.647 [2.196,3.132]	2.899 [2.433,3.369]