FOUNDATION MODELS FOR EVENT PREDICTION ACROSS HETEROGENEOUS EVENT LOGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Event logs are ubiquitous across real-world domains, including business processes, manufacturing workflows, healthcare, and many other operational systems. Yet event logs can differ substantially in their event semantics, label spaces, interaction patterns, and temporal scales, making it difficult to build a single foundation model that generalizes directly to previously unseen systems. We address this challenge with the **Foundation Event Predictor (FEP)**, a simple and scalable *in-context* architecture that performs next-event prediction through learned, event-level nonparametric inference. Causal self-attention encodes context and target histories into key and query representations, respectively, while the observed next context events are encoded as values. Cross-attention then performs learned, relevance-weighted aggregation of next-event outcomes over all context transitions, allowing the entire nonparametric prediction procedure to be expressed as standard attention operations and efficiently parallelized with modern kernels such as FlashAttention. To promote cross-system generalization, we introduce permutation-robust embedding learning and random fixed embeddings to decouple event IDs from persistent semantics, together with empirical quantile transforms and robust normalization to handle diverse temporal scales. We pretrain FEP on 696 real-world event datasets spanning heterogeneous event systems and evaluate it on 50 held-out datasets unseen during pretraining. Across these datasets, FEP generalizes effectively to new event systems without parameter adaptation, frequently approaching the performance of strong models trained separately on each target dataset while substantially outperforming existing foundation-model baselines, particularly on challenging prediction tasks.

1 INTRODUCTION

Event logs arise across numerous real-world domains and modern operational systems. Examples include loan applications and claims processing in financial services, purchasing and replenishment in supply chains, order-to-cash and procure-to-pay workflows in enterprise systems, production and maintenance events in manufacturing, patient visits and clinical procedures in healthcare, and alarms and incidents in computing infrastructure. In each setting, an event log records sequences of discrete activities together with their occurrence times and potentially other descriptive information. Given an ongoing event sequence, predicting what event will occur next and when it will occur is a fundamental problem for understanding, monitoring, and anticipating the evolution of the underlying process. Accurate next-event prediction can support applications ranging from process monitoring, optimization, and resource planning to anomaly detection and proactive intervention (Di Francescomarino & Ghidini, 2022; Ceravolo et al., 2024; Lee et al., 2022).

Building a general-purpose predictive model for event logs is challenging because event systems can differ substantially in their event semantics, label spaces, interaction patterns, sequence characteristics, and temporal scales. An event ID representing a purchase order in one system may have no relationship to the same ID in a clinical or manufacturing log. Consequently, most existing predictive approaches train a separate model for each event log, tying the learned predictor to the activity vocabulary, temporal scale, and dynamics of that particular system (Mei & Eisner, 2017; Zuo et al., 2020; Shchur et al., 2020; Evermann et al., 2017; Tax et al., 2017; Bukhsh et al., 2021). Recent foundation-model approaches have begun to relax this restriction (Berghaus et al., 2026; Berti &

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.

van der Aalst, 2026b), but generalizing directly across highly heterogeneous event systems remains challenging.

To address this problem, we propose **Foundation Event Predictor (FEP)**, a simple and scalable in-context architecture for next-event prediction across heterogeneous event logs. FEP treats observed context sequences as a collection of event-level predictive examples, each pairing an observed history with its subsequent event. Causal self-attention encodes context histories into keys and target histories into queries, while the observed subsequent context events form the corresponding values. Cross-attention then performs learned, relevance-weighted aggregation of next-event outcomes over the context, implementing an event-level nonparametric predictor. Its output is decoded into the next event type and inter-arrival time. In this way, FEP exploits the finest-grained supervision available in an event log—*the multitude of observed history-to-next-event transitions*—and learns across systems how to transfer these predictive relationships to unseen systems. At the same time, expressing the entire nonparametric inference procedure through standard attention enables highly parallel computation over large context sets and direct use of optimized kernels such as FlashAttention (Dao et al., 2022; Dao, 2024).

To further improve generalization across systems with unrelated event vocabularies, we prevent FEP from associating fixed event IDs with persistent semantics. We introduce two complementary strategies: permutation training, which randomly permutes the mapping from event types to a shared learned embedding table in each training episode, and random fixed embeddings, which generate fresh non-trainable event-type representations in each episode. Both encourage FEP to infer predictive structure from the observed context rather than memorize dataset-specific label identities. To accommodate heterogeneous temporal scales and large within-log variation, we further introduce two context-dependent transformations of event inter-arrival times: an empirical quantile transform and a robust log-based normalization. FEP is trained using these normalized time representations to improve robustness across diverse temporal distributions.

We systematically evaluate FEP on a large and heterogeneous collection of real-world event logs, pretraining on 696 datasets and testing on 50 held-out datasets unseen during pretraining. We compare against strong event-prediction models trained separately on each held-out dataset as well as existing foundation models for event data, including FIM-PP (Berghaus et al., 2026) and FM-v2 (Berti & van der Aalst, 2026b). Across different difficulty levels, FEP consistently approaches the strongest local models in both event-type accuracy and inter-arrival-time prediction error while substantially outperforming competing foundation models. Moreover, FEP substantially outperforms an architecture-matched local counterpart on average, demonstrating the benefit of cross-dataset pretraining for generalization. We further conduct ablations on context size, the number of pretraining datasets, model dimension, and other design choices to characterize the factors driving FEP's performance, and analyze its inference efficiency and scalability.

2 Related Work

Temporal point processes. Temporal point processes provide a classical framework for modeling irregular event sequences (Daley & Vere-Jones, 2003; Hawkes, 1971). Hawkes processes capture self- and cross-excitation among event types, while modern neural point processes use recurrent or attention-based architectures to learn richer history-dependent dynamics, including RMTPP (Du et al., 2016), Neural Hawkes Processes (Mei & Eisner, 2017), and Transformer-based models (Zuo et al., 2020; Zhang et al., 2020; Yang et al., 2022). Intensity-free approaches instead directly parameterize next-event-time distributions (Shchur et al., 2020). These models are typically trained separately for each event system and operate within a fixed event-type vocabulary, whereas we seek a single predictor that generalizes directly across heterogeneous, previously unseen event systems.

Predictive process monitoring. In process mining (van der Aalst, 2016), predictive process monitoring forecasts future behavior from prefixes of running cases. Deep models based on recurrent networks (Evermann et al., 2017; Tax et al., 2017) and Transformers (Bukhsh et al., 2021) have been developed for tasks such as next-activity, next-event-time, and remaining-time prediction. Despite strong predictive performance, these approaches generally train a separate model for each event log, tying the learned predictor to the activity vocabulary, temporal scale, and dynamics of that particular process.

Foundation inference for temporal point processes. The work most closely related to ours from the point-process perspective is FIM-PP (Berghaus et al., 2026), which pretrains a Foundation Inference Model on a large synthetic distribution of marked point processes and infers conditional intensity functions for unseen systems from context sequences. FIM-PP and FEP thus share the goal of transferring a single pretrained model to new event systems, but differ substantially in both training regime and inference mechanism. FIM-PP performs amortized inference of a Hawkes-like conditional intensity under a synthetic Hawkes-family prior. In contrast, FEP is trained across heterogeneous real-world event logs and treats each observed history-to-next-event transition as an in-context predictive example: context histories form attention keys and their observed next events form values. Cross-attention then learns to aggregate these event-level outcomes for direct next-event prediction, without explicitly estimating an intensity function.

Foundation models for predictive process monitoring. Berti and van der Aalst (Berti & van der Aalst, 2026a) study in-context prediction on unseen event logs without parameter updates. Their model learns prefix embeddings and applies two separate nonparametric heads: prototype-based classification for next activity and kernel regression for remaining case time, i.e., the time until process completion rather than the time to the next activity. For each activity type, the prototype is computed as the mean embedding of support prefixes whose next activity has that type. The model uses balanced contexts or retrieval-based support selection. Their follow-up FM-v2 (Berti & van der Aalst, 2026b) makes retrieval central to both training and inference, learning representations explicitly optimized for nearest-neighbor selection of target-log support prefixes.

These works are related to FEP in their use of in-context examples and locally defined activity vocabularies. FEP, however, directly models the *next event*, predicting both its event type and inter-arrival time, and unifies nonparametric inference through learned cross-attention over event-level history-to-next-event pairs. This formulation jointly learns the relevance function and outcome representation while exploiting the finest-grained supervision available in the log. Moreover, the nonparametric prediction is expressed entirely through standard attention operations, enabling highly parallel batched computation using optimized attention implementations such as FlashAttention and FlashAttention-2 (Dao et al., 2022; Dao, 2024).

3 Methodology

3.1 Problem Setting

Suppose we are given N event datasets $\mathcal{P} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ for pretraining, where each $\mathcal{D}_i$ ($1 \le i \le N$) contains event sequences from a particular system,

$$\mathcal{D}_i = \{\Gamma_n^i\}_n, \qquad \Gamma_n^i = (\mathbf{e}_{n1}^i, \mathbf{e}_{n2}^i, \ldots),$$

and each event $\mathbf{e}_{nj}^i = (\Delta t_{nj}^i, s_{nj}^i)$ consists of an inter-arrival time and an event type. Different datasets may have unrelated event semantics and label spaces, different numbers of event types, distinct transition and interaction patterns, and drastically different temporal scales.

Our goal is to learn a single model that directly generalizes to previously unseen event systems without parameter adaptation. At inference time, a new system provides a *context set* of exemplar sequences $\mathcal{D}^C = \{\Gamma_n^C\}_n$ and a set of ongoing query sequences $\mathcal{D}^Q = \{\Gamma_n\}_n$. Given the observed history of a query sequence up to event $\mathbf{e}_m$, FEP uses $\mathcal{D}^C$ as in-context support to predict the next event $\mathbf{e}_{m+1} = (\Delta t_{m+1}, s_{m+1})$.

3.2 Event-Level In-Context Prediction

The central idea of FEP is to decompose the context $\mathcal{D}^C$ into event-level predictive examples

$$H_{nj}^C \longrightarrow \mathbf{e}_{n,j+1}^C,$$

where $H_{nj}^C = (\mathbf{e}_{n1}^C, \ldots, \mathbf{e}_{nj}^C)$ denotes the history up to event $\mathbf{e}_{nj}^C$. FEP then performs learned nonparametric inference over this collection using standard attention. See Figure 1.

Specifically, for each event $\mathbf{e} = (\Delta t, s)$ in $\mathcal{D}^C$ and $\mathcal{D}^Q$, we construct an input representation $\mathbf{h} = [\alpha(\Delta t); A(s)]$, where $\alpha(\cdot)$ and $A(\cdot)$ denote time and event-type embeddings, respectively. The

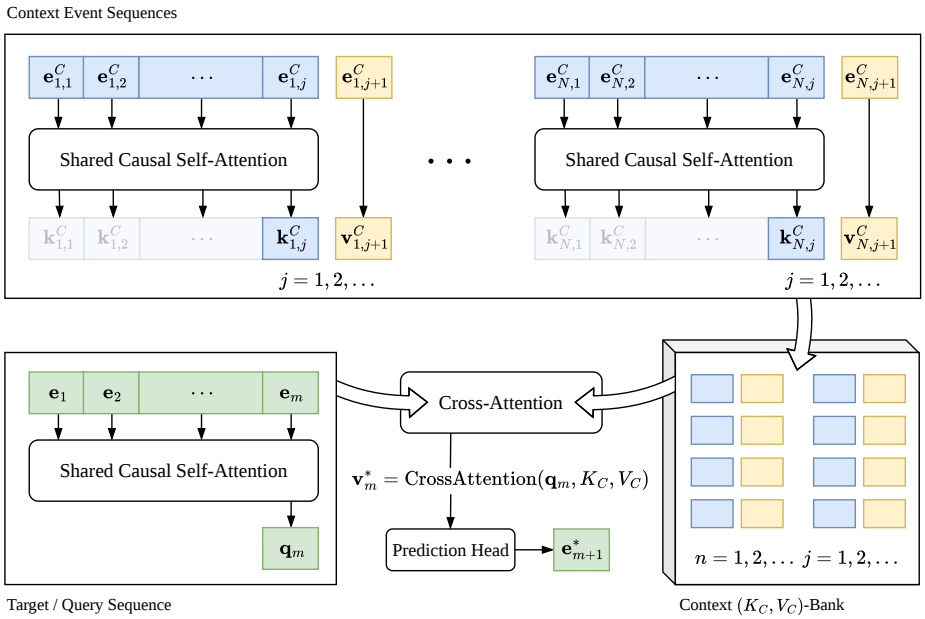


Figure 1: The architecture of Foundation Event Predictor (FEP). Context histories and observed next events form keys and values, respectively, while target histories form queries for cross-attention-based nonparametric prediction.

time embedding operates on the normalized time representation introduced later. We add sinusoidal positional encodings and encode both context and query sequences with a *shared* causal Transformer. Specifically, letting $\mathbf{x}_j = \mathbf{h}_j + \mathbf{p}_j$, where $\mathbf{p}_j$ denotes the sinusoidal positional encoding at position j, we compute

$$(\mathbf{z}_1, \ldots, \mathbf{z}_L) = \text{CausalSelfAttention}(\mathbf{x}_1, \ldots, \mathbf{x}_L).$$

The causal mask ensures that $\mathbf{z}_j$ summarizes only the history up to position j. We then construct

$$\mathbf{k}^C_{nj} = W_K \mathbf{z}^C_{nj}, \qquad \mathbf{q}_m = W_Q \mathbf{z}_m,$$

so that each context history is represented by a key and each target history by a query. The corresponding value encodes the *observed outcome* of the context history, i.e., the next event. Specifically, for $\mathbf{e}^C_{n,j+1} = (\Delta t^C_{n,j+1}, s^C_{n,j+1})$, we define

$$\mathbf{v}^C_{nj} = [\mathbf{v}^C_{\Delta,nj}; \mathbf{v}^C_{s,nj}],$$

where

$$\mathbf{v}^C_{\Delta,nj} = \text{MLP}_{V,\Delta}\left(\alpha(\Delta t^C_{n,j+1})\right), \qquad \mathbf{v}^C_{s,nj} = \text{MLP}_{V,s}\left(\text{CenterOneHot}(s^C_{n,j+1})\right).$$

Here, $\text{CenterOneHot}(\cdot)$ denotes the centered one-hot encoding of the event type. Thus, $(\mathbf{k}^C_{nj}, \mathbf{v}^C_{nj})$ explicitly represents a history-to-next-event example.

Collecting all context keys and values into K_C and V_C, FEP predicts the next event of the target history through cross-attention,

$$\mathbf{v}^*_m = \text{CrossAttention}(\mathbf{q}_m, K_C, V_C).$$

For a single attention head, this corresponds to

$$\mathbf{v}^*_m = \sum_{n,j} a_{m,nj} \mathbf{v}^C_{nj}, \qquad a_{m,nj} = \frac{\exp(\mathbf{q}_m^\top \mathbf{k}^C_{nj}/\sqrt{d})}{\sum_{n',j'} \exp(\mathbf{q}_m^\top \mathbf{k}^C_{n'j'}/\sqrt{d})},$$

where d is the key/query dimension. Hence, attention performs a learned, relevance-weighted nonparametric aggregation over all observed context transitions: more relevant context histories receive larger weights, whereas irrelevant transitions can be effectively ignored.

Writing $\mathbf{v}_m^* = [\mathbf{z}_\Delta^*; \mathbf{z}_s^*]$, two lightweight prediction heads decode

$$\widehat{y}_{m+1} = \mathrm{MLP}_\Delta(\mathbf{z}_\Delta^*), \qquad \boldsymbol{\ell}_{m+1} = \mathrm{MLP}_s(\mathbf{z}_s^*),$$

where $\widehat{y}_{m+1}$ is the predicted normalized inter-arrival time and $\boldsymbol{\ell}_{m+1}$ contains logits over the target system's event types. We optimize

$$\mathcal{L} = \mathrm{CE}(\boldsymbol{\ell}_{m+1}, s_{m+1}) + \lambda \cdot \mathrm{MSE}(\widehat{y}_{m+1}, y_{m+1}),$$

where y denotes the normalized inter-arrival-time representation defined below.

This architecture directly exploits the finest-grained predictive information available in the context set—the multitude of observed history-to-next-event transitions. Importantly, both causal history encoding and context aggregation are standard attention operations: keys and queries for many sequences can be computed in parallel, while cross-attention over large context sets can exploit efficient implementations such as FlashAttention (Dao et al., 2022; Dao, 2024).

3.3 Robust Representation of Time

Time scales can differ dramatically across event systems and even within a single process. For example, activities in a loan application may be separated by seconds or minutes, while later approval stages may take days. We therefore normalize inter-arrival times using statistics computed exclusively from the context set $\mathcal{D}^C$.

We consider two alternatives. The first constructs an empirical quantile transform F_C from context inter-arrival times and maps $y = F_C(\Delta t) \in [0, 1]$. FEP predicts y, and the prediction is mapped back to the original time scale using F_C^{-1}.

The second performs robust normalization in log time. For each context inter-arrival time Δt_i, let $r_i = \log(\Delta t_i + \epsilon)$, and define: $m_C = \mathrm{median}_i(r_i)$, $\mathrm{MAD}_C = \mathrm{median}_i\, |r_i - m_C|$. We then set

$$s_C = \max(\tau\ \mathrm{MAD}_C, s_{\min}), \qquad y = \frac{\log(\Delta t + \epsilon) - m_C}{s_C},$$

where $\tau = 1/\Phi^{-1}(0.75) \approx 1.4826$ makes the scaled MAD a consistent estimator of the standard deviation under a Gaussian distribution (Rousseeuw & Croux, 1993). This normalization remains robust to extreme waiting times. Predictions are mapped back to the original time scale through $\widehat{\Delta t} = \exp(m_C + s_C \widehat{y}) - \epsilon$.

3.4 Removing Spurious Semantics from Event IDs

A shared event-ID embedding table is problematic across heterogeneous logs because the same numerical ID can represent completely unrelated activities in different systems. We therefore prevent FEP from relying on persistent ID semantics and instead encourage it to infer event relationships from the observed context.

We study two strategies. **Permutation training** maintains a shared learned embedding table while randomly permuting the correspondence between event types and embedding IDs in each training episode. The embedding table thus provides a shared representational basis without assigning persistent semantics to particular IDs. Alternatively, **random fixed embeddings** generate a fresh set of random event-type vectors for each episode. These vectors distinguish event types within the current system but are neither optimized nor preserved across episodes, thereby eliminating persistent semantics associated with particular event IDs.

3.5 Training Across Event Systems

Training proceeds episodically across the pretraining datasets. For each event system, a context loader samples sequences used to construct (K_C, V_C), while an inference loader samples query sequences and their next-event targets. FEP computes query representations, performs cross-attention against the context using FlashAttention, evaluates the prediction loss, and immediately backpropagates the loss for the current query batch. The corresponding computation graph is then released before moving to the next system, reducing peak memory consumption. Gradients are accumulated across systems, followed by a parameter update at the end of each episode. We further parallelize training across GPUs using PyTorch DistributedDataParallel (DDP) (Li et al., 2020). The training procedure is summarized in Algorithm 1 in the appendix.

4 EXPERIMENTS

4.1 GENERALIZATION PERFORMANCE

To evaluate generalization to unseen event systems, we test FEP on 50 public real-world datasets spanning diverse domains, including business processes, finance, healthcare, mobility, environmental monitoring, and public administration. Table 9b in the appendix summarizes the datasets by domain, while Table 10 provides the full list and descriptions. FEP is pretrained on 696 other real-world datasets collected from public repositories such as 4TU.ResearchData and Kaggle. We ensure that the 50 test datasets do not overlap with the pretraining data in dataset or process identity, including duplicate or derived versions of the same underlying event log. A summary of the pretraining datasets is provided in Table 9a, with further details in Appendix C.

Competing Methods. We compare against strong task-specific local methods for event prediction and existing foundation models for event data. The local baselines include a Transformer designed specifically for next-event prediction, GRU (Cho et al., 2014), Neural Hawkes Process (Mei & Eisner, 2017), Transformer Hawkes Process (Zuo et al., 2020), and Intensity-Free Temporal Point Process (IFTPP) (Shchur et al., 2020). Each method is trained separately on the training set of each dataset. For each model, we select the best hyperparameter configuration and training epoch based on validation performance and report the corresponding test performance.

We also compare FEP with existing foundation models for event data, including FM-v2 (Berti & van der Aalst, 2026b) and FIM-PP (Berghaus et al., 2026). For these models, we provide in-context examples from the training sets and evaluate predictive performance on the corresponding test sets. We use the published checkpoints for FM-v2 and FIM-PP. Finally, to quantify the benefit of cross-dataset pretraining, we evaluate an architecture-matched local counterpart of FEP, denoted as FEP-Local. Full implementation, validation, and configuration details are provided in Appendix A.

Metrics. For each test dataset, we report next-event-type prediction accuracy and normalized mean absolute error (NMAE) for next-event inter-arrival time. We define $\mathrm{NMAE} = \frac{\mathrm{MAE}}{\mu_\Delta + 1\ \text{second}}$, where μ_Δ is the mean ground-truth inter-arrival time on the test set and MAE is measured in seconds. The one-second constant is included for numerical robustness.

Evaluation. For each dataset, we use the newest 15% of traces, ordered by sequence start time, as the test set. We randomly select 15% of the remaining traces for validation and use the rest for training.

To characterize performance across different levels of empirical prediction difficulty, we group the 50 datasets into *easy*, *moderate*, and *hard* categories based on the test performance of GRU and Neural Hawkes, whose hyperparameters and training epochs are selected independently on the validation sets. We choose these two models because they are well established and represent distinct modeling paradigms: GRU performs direct sequence prediction, whereas Neural Hawkes models event dynamics through a temporal point process. For each dataset, we average the two models' event-type accuracies and, separately, their NMAEs. Datasets are ranked in descending order of accuracy and ascending order of NMAE, and the two ranks are averaged to obtain an overall difficulty score. We then divide the datasets into three approximately equal-sized groups according to this score. Note that this grouping is used only for stratified analysis; all 50 datasets are included in the overall evaluation.

Since average metrics can be disproportionately influenced by a small number of catastrophic predictions, we additionally report Bradley–Terry scores for both event-type accuracy and NMAE (Bradley & Terry, 1952). For each dataset, methods are compared pairwise, with higher accuracy (or lower NMAE) counted as a win. We aggregate the resulting pairwise outcomes across datasets and jointly estimate a latent performance parameter for each method under the Bradley–Terry model. A higher score therefore indicates that a method wins pairwise comparisons more consistently across datasets, rather than being dominated by a small number of unusually large gains or failures.

Results and Discussion. Tables 1a and 1b report the average event-type accuracy and inter-arrival-time NMAE across the three difficulty categories. The four combinations

Table 1: Generalization performance by dataset difficulty, averaged over five runs across 50 test datasets. The best results among local methods and foundation models are highlighted in bold. FEP (Quantile) uses permutation training with the empirical quantile transform, while FEP (Robust log) uses random fixed embeddings with robust log-time normalization; all four combinations are reported in Appendix B. FEP-Local is the architecture-matched local counterpart of FEP (Quantile), trained separately on each target dataset. FIM-PP supports at most 22 activities; we therefore retain the 21 most frequent activities and merge the rest into an "other" activity. FIM-PP (FT) denotes FIM-PP further fine-tuned on the training set of each target dataset. FM-v2 is omitted from NMAE because it predicts remaining case time, while Transformer Hawkes and IFTPP are omitted because their NMAEs exceed 10,000 across all difficulty levels.

(a) Next-event-type accuracy.

Method	Easy	Moderate	Hard
Transformer	$\mathbf{0.83 \pm 0.00}$	$\mathbf{0.66 \pm 0.00}$	$\mathbf{0.38 \pm 0.03}$
GRU	$\mathbf{0.83 \pm 0.00}$	$\mathbf{0.66 \pm 0.01}$	0.37 ± 0.01
Neural Hawkes	$\mathbf{0.83 \pm 0.00}$	0.65 ± 0.00	0.31 ± 0.01
Transformer Hawkes	0.76 ± 0.02	0.57 ± 0.05	0.24 ± 0.01
IFTPP	0.81 ± 0.01	0.62 ± 0.01	0.31 ± 0.03
FEP-Local	0.77 ± 0.02	0.60 ± 0.02	0.26 ± 0.04
FM-v2	0.77 ± 0.00	0.55 ± 0.00	0.24 ± 0.00
FIM-PP	0.30 ± 0.01	0.38 ± 0.00	0.14 ± 0.00
FIM-PP (FT)	0.57 ± 0.04	0.53 ± 0.02	0.32 ± 0.05
FEP (Quantile)	$\mathbf{0.82 \pm 0.00}$	$\mathbf{0.64 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$
FEP (Robust log)	$\mathbf{0.82 \pm 0.00}$	$\mathbf{0.64 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$

(b) Next-event inter-arrival-time NMAE.

Method	Easy	Moderate	Hard
Transformer	0.52 ± 0.36	$\mathbf{0.54 \pm 0.01}$	11.22 ± 17.78
GRU	$\mathbf{0.28 \pm 0.00}$	$\mathbf{0.54 \pm 0.01}$	$\mathbf{0.83 \pm 0.01}$
Neural Hawkes	0.38 ± 0.03	0.76 ± 0.04	1.36 ± 0.81
FEP (Local)	0.31 ± 0.04	1.75 ± 0.87	0.88 ± 0.05
FIM-PP	0.79 ± 0.00	0.83 ± 0.00	1.27 ± 0.00
FIM-PP (FT)	0.45 ± 0.02	0.82 ± 0.00	1.07 ± 0.01
FEP (Quantile)	$\mathbf{0.30 \pm 0.00}$	$\mathbf{0.53 \pm 0.00}$	$\mathbf{0.83 \pm 0.00}$
FEP (Robust log)	$\mathbf{0.30 \pm 0.00}$	0.55 ± 0.00	0.84 ± 0.00

of event-ID representation and time normalization perform similarly; we therefore report two representative variants here and provide all four configurations in Appendix B. FEP consistently achieves the best performance among the foundation models in both event-type accuracy and time prediction. In particular, FIM-PP remains substantially behind FEP even after target-specific fine-tuning. Compared with task-specific local models, FEP is only slightly behind, and in several cases on par with, the strongest local baselines—Transformer for event-type accuracy and GRU for NMAE. In contrast, FEP-Local shows substantially larger gaps from the best local models across all three difficulty categories. Cross-dataset pretraining markedly improves over FEP-Local and brings FEP close to the strongest task-specific models, demonstrating the benefit of learning predictive structure across heterogeneous event systems.

Table 2: Overall pairwise ratings across 50 test datasets, reported as Bradley–Terry scores (higher is better). Scores are computed jointly from pairwise comparisons among all local, foundation, and fine-tuned models.

Method	MAE score	Accuracy score
Transformer	1644	**1664**
GRU	**1690**	1631
Neural Hawkes	1518	1619
Transformer Hawkes	1377	1422
IFTPP	1169	1540
FEP (Local)	1641	1436
FM-v2	—	1382
FIM-PP	1396	1176
FIM-PP (FT)	1433	1332
FEP (Quantile)	**1675**	**1561**
FEP (Robust log)	1631	1539

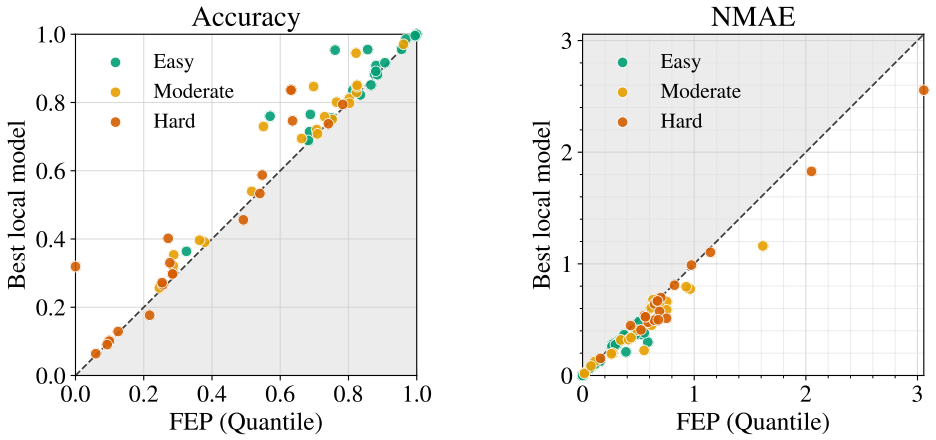


Figure 2: Dataset-wise comparison between FEP and the best locally trained model. Colors indicate empirical dataset difficulty, and the dashed line denotes equal performance. The shaded region indicates the side of the diagonal where FEP performs better.

We next examine performance on individual datasets. Figure 2 compares FEP with the best local model separately for each test dataset, where the identity of the best local model may *vary across datasets*. Most points lie close to the diagonal for both accuracy and NMAE, showing that FEP generally approaches the performance of the best locally trained model despite requiring no target-specific parameter adaptation. A small number of datasets exhibit larger gaps.

Finally, the pairwise comparisons summarized by the Bradley–Terry scores in Table 2 show that FEP achieves the highest scores among foundation models and remains competitive with the local baselines. Results on six representative datasets are further provided in Appendix B (Table 8), where FEP matches or outperforms local models in either event-type accuracy or NMAE and outperforms the competing foundation models in all but one case, where FM-v2 attains the same event-type prediction accuracy.

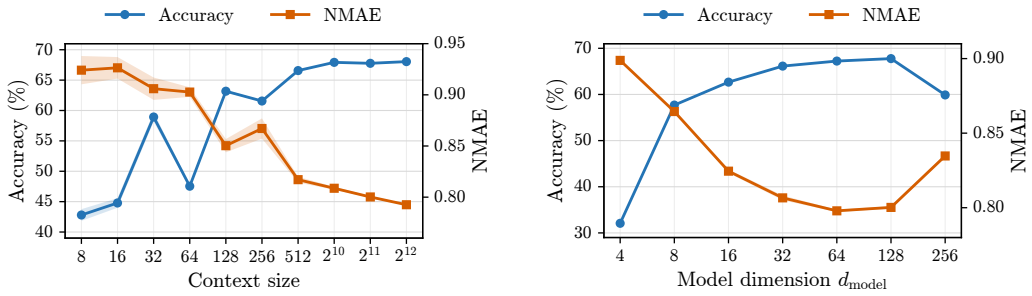
4.2 Ablation Studies

We conduct a series of ablation studies to examine the factors affecting FEP's performance. For computational efficiency, unless otherwise stated, we pretrain on 33 datasets randomly sampled from the full pretraining pool and report accuracy and NMAE averaged over a fixed subset of 23 held-out datasets from the 50 test datasets.

Context size. Figure 3a shows the effect of varying the number of context traces used during both training and inference. Increasing the context size typically improves both accuracy and NMAE, with diminishing returns as more context is provided. Because each trace contains multiple events, even a moderate number of traces yields a large event-level context for cross-attention. The continued improvement with increasing context size suggests that FEP can effectively exploit additional context without evident attention dilution over the range considered.

Number of Pretraining Datasets. Table 3a examines the effect of pretraining scale. With only 3 or 10 datasets, FEP generalizes poorly to unseen event systems, including unstable time prediction when trained on 10 datasets. Increasing the pretraining set to 30 datasets substantially improves both event-type accuracy and NMAE. Performance changes only modestly from 30 to 50 datasets, while pretraining on the full collection of 696 datasets provides a further improvement. These results indicate that exposure to diverse event systems is important for cross-system generalization, with gains becoming smaller as the number of pretraining datasets increases.

Model Size. Figure 3b and Table 3b examine the effects of model dimension and depth. FEP performs well with relatively shallow networks: increasing depth from one to a few layers provides



(a) Context size *vs.* performance.

(b) Model dimension *vs.* performance.

Figure 3: Ablation results for context size and model dimension.

Table 3: Ablation results for the number of pretraining datasets and model depth. ∞ denotes NMAE larger than 10,000.

(a)

#Pretraining datasets	Accuracy (%)	NMAE
3	39.09 ± 0.00	0.95 ± 0.00
10	49.52 ± 0.05	∞
30	67.45 ± 0.01	0.80 ± 0.00
50	67.59 ± 0.02	0.80 ± 0.00
696	$\mathbf{68.70 \pm 0.02}$	$\mathbf{0.77 \pm 0.00}$

(b)

#Layers	Accuracy (%)	NMAE
1	66.77 ± 0.03	0.83 ± 0.00
2	$\mathbf{67.91 \pm 0.02}$	0.80 ± 0.00
4	67.77 ± 0.03	0.80 ± 0.00
6	67.70 ± 0.04	$\mathbf{0.79 \pm 0.00}$
8	62.34 ± 0.04	0.81 ± 0.00

modest gains, whereas excessive depth degrades event-type accuracy. Increasing the model dimension yields substantial improvements up to moderate widths, with performance largely saturating around $d_{\text{model}} = 64$–128; increasing it further to 256 degrades both accuracy and NMAE. Overall, these results suggest that FEP does *not* require an excessively large backbone to achieve strong cross-system generalization.

Additional ablation results, including the full combinations of event-ID representation and time normalization, as well as an analysis of inference efficiency and scalability, are provided in Appendix B.

5 Limitations

A primary limitation of the current FEP framework is that it only models event types and timestamps, without incorporating additional event attributes or sequence-level metadata. In practice, event logs often contain rich auxiliary information that can provide strong predictive signals and further improve next-event prediction. However, such information is itself highly heterogeneous across datasets: feature schemas, value ranges, categorical cardinalities, availability, and noise levels can differ substantially from one event log to another. Designing a representation that can flexibly incorporate these heterogeneous attributes while preserving cross-system generalization therefore remains a challenging open problem. We leave this extension to future work.

6 Conclusion

We have presented FEP, an in-context foundation model that generalizes to unseen heterogeneous event logs. The model is pretrained and evaluated on diverse real-world event data. Its next-event prediction performance without parameter adaptation is encouraging, often approaching strong local models trained specifically on target datasets. Future work will incorporate heterogeneous event features, expand the pretraining pool with synthetic data, and investigate uncertainty calibration for both prediction and process simulation.

AI USE STATEMENT

In this work, we used generative AI tools to assist with polishing the writing, conducting literature searches and identifying additional relevant work, implementing data-processing scripts, and building basic pretraining and evaluation pipelines. We did not use generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, support qualitative or thematic data analysis, or interpret results. We did not use generative AI for research ideation. Generating synthetic datasets and assisting with proof writing are not applicable to this work.

We reviewed all AI-assisted work. We proofread and edited all AI-assisted text and verified and tested all AI-assisted code for correctness. We take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of the proposed architecture, training procedure, and evaluation protocol in the main text and appendices. In particular, Section 3 and Appendix A specify the model architecture, hyperparameter settings, training configurations, baseline implementations, model-selection procedures, and evaluation protocol. Additional experimental results are reported in Appendix B.

We also provide the complete list of the 50 test datasets, together with their sources and preprocessing details, in Table 10. The pretraining datasets are collected from publicly available online sources; however, due to current proprietary constraints, we are unable to release the exact identities of the 696 pretraining datasets, the processed pretraining data, or the training code. The methodological and experimental details provided in the paper are intended to support independent reimplementation of FEP and reproduction of the evaluation protocol and main comparisons.

REFERENCES

Nuno Antonio, Ana de Almeida, and Luis Nunes. Hotel booking demand datasets. *Data in Brief*, 22: 41–49, 2019. doi: 10.1016/j.dib.2018.11.126.

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

Petr Berka and Marta Sochorova. PKDD'99 discovery challenge: Financial data set, 1999. URL https://github.com/jlacko/berka-dataset. Prepared for the PKDD'99 Discovery Challenge.

Alessandro Berti and Wil M. P. van der Aalst. An in-context foundation model for predictive process monitoring on event logs. *IEEE Access*, 14:16959–16983, 2026a. doi: 10.1109/ACCESS.2026.3658877.

Alessandro Berti and Wil M. P. van der Aalst. Retrieval-augmented in-context foundation model for predictive process monitoring. Preprints.org, 2026b. URL https://www.preprints.org/manuscript/202607.0705.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Zaharah A. Bukhsh, Aaqib Saeed, and Remco M. Dijkman. ProcessTransformer: Predictive business process monitoring with transformer network, 2021. URL https://arxiv.org/abs/2104.00721.

Paolo Ceravolo, Marco Comuzzi, Jochen De Weerdt, Chiara Di Francescomarino, and Fabrizio Maria Maggi. Predictive process monitoring: Concepts, challenges, and future research directions. *Process Science*, 1(1):2, 2024. doi: 10.1007/s44311-024-00002-4.

Song Chen. Beijing multi-site air quality, 2017. URL https://archive.ics.uci.edu/dataset/501/beijing+multi+site+air+quality+data.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179/.

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

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, volume 35, pp. 16344–16359, 2022.

Chiara Di Francescomarino and Chiara Ghidini. Predictive process monitoring. In Wil M. P. van der Aalst and Josep Carmona (eds.), *Process Mining Handbook*, volume 448 of *Lecture Notes in Business Information Processing*, pp. 320–346. Springer, 2022. doi: 10.1007/978-3-031-08848-3_10.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1555–1564, 2016.

Joerg Evermann, Jana-Rebecca Rehse, and Peter Fettke. Predicting process behaviour using deep learning. *Decision Support Systems*, 100:129–140, 2017. doi: 10.1016/j.dss.2017.04.003.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, 2015. doi: 10.1145/2827872.

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

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 387–395, 2018. doi: 10.1145/3219819.3219845.

Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus A. Brubaker. Time2vec: Learning a vector representation of time. *CoRR*, abs/1907.05321, 2019.

Suhwan Lee, Xixi Lu, and Hajo A. Reijers. The analysis of online event streams: Predicting the next activity for anomaly detection. In Renata Guizzardi, Jolita Ralyté, and Xavier Franch (eds.), *Research Challenges in Information Science*, volume 446 of *Lecture Notes in Business Information Processing*, pp. 248–264. Springer, 2022. doi: 10.1007/978-3-031-05760-1_15.

Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. Pytorch distributed: Experiences on accelerating data parallel training. *Proceedings of the VLDB Endowment*, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Julian McAuley and Jure Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd International Conference on World Wide Web*, pp. 897–908, 2013. doi: 10.1145/2488388.2488466. URL https://snap.stanford.edu/data/web-FineFoods.html.

Michael McCann and Adrian Johnston. SECOM, 2008. URL https://archive.ics.uci.edu/dataset/179/secom.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, pp. 6754–6764, 2017.

Peter J. Rousseeuw and Christophe Croux. Alternatives to the median absolute deviation. *Journal of the American Statistical Association*, 88(424):1273–1283, 1993. doi: 10.1080/01621459.1993.10476408.

Mohammed Saeed, Mauricio Villarroel, Andrew Reisner, Gari Clifford, Li wei Lehman, George Moody, Thomas Heldt, Tin Kyaw, Benjamin Moody, and Roger Mark. MIMIC-II Clinical Database, 2011. URL https://physionet.org/content/mimic-ii/2.6.0/. Version 2.6.0.

Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, pp. 108–116, 2018. doi: 10.5220/0006639801080116. URL https://www.unb.ca/cic/datasets/ids-2017.html.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HygOjhEYDH.

Fabian Spaeh, Jingxing Fang, Shandian Zhe, and Bin Shen. Universal multi-modal traceformer: Integrating heterogeneous context for process event prediction, 2026. URL https://arxiv.org/abs/2609.24579.

Mark Svoboda, Doug LeComte, Mike Hayes, Richard Heim, Karin Gleason, Jim Angel, Brad Rippey, Rich Tinker, Mike Palecki, David Stooksbury, David Miskus, and Scott Stephens. The drought monitor. *Bulletin of the American Meteorological Society*, 83(8):1181–1190, 2002. doi: 10.1175/1520-0477(2002)083<1181:TDM>2.3.CO;2.

Niek Tax, Ilya Verenich, Marcello La Rosa, and Marlon Dumas. Predictive business process monitoring with LSTM neural networks. In *Advanced Information Systems Engineering*, volume 10253 of *Lecture Notes in Computer Science*, pp. 477–492. Springer, 2017. doi: 10.1007/978-3-319-59536-8_30.

Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. Springer, Berlin, Heidelberg, 2 edition, 2016. ISBN 978-3-662-49850-7. doi: 10.1007/978-3-662-49851-4.

Boudewijn F. van Dongen. BPI Challenge 2015, 2015. URL https://research.tue.nl/en/publications/bpi-challenge-2015/.

Siqiao Xue, Xiaoming Shi, Zhixuan Chu, Yan Wang, Hongyan Hao, Fan Zhou, Caigao JIANG, Chen Pan, James Y Zhang, Qingsong Wen, et al. Easytpp: Towards open benchmarking temporal point processes. In *The Twelfth International Conference on Learning Representations*, 2024.

Chenghao Yang, Hongyuan Mei, and Jason Eisner. Transformer embeddings of irregularly spaced events and their participants. In *Proceedings of the Tenth International Conference on Learning Representations (ICLR)*, 2022.

Qiang Zhang, Aldo Lipani, Omer Kirnap, and Emine Yilmaz. Self-attentive hawkes process. In *International Conference on Machine Learning*, pp. 11183–11193. PMLR, 2020.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. In *International Conference on Machine Learning*, pp. 11692–11702. PMLR, 2020.

APPENDIX

A EXPERIMENTAL DETAILS

FEP configuration and training. The shared causal self-attention and cross-attention modules described in the main text use a hidden dimension of 128, four attention heads, four attention layers, a feed-forward dimension of 512, and dropout of 0.05. The maximum training pool for an individual event system is 2,560 traces; for larger systems, a new subset is sampled in each training episode. The context and query batch sizes are 512 and 128 traces, respectively. On each worker, four context minibatches are combined to form a context set of up to 2,048 traces for cross-attention.

We optimize FEP using AdamW (Loshchilov & Hutter, 2019) with a learning rate of 10^{-3} and weight decay of 10^{-5}. Matrix multiplications use TF32 where supported, while the forward pass uses bfloat16 automatic mixed precision. We train FEP for 500 episodes and use the resulting pretrained model for all evaluations reported in the paper.

Training is distributed across four NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs on a Linux machine with 96 Intel Xeon Scalable CPU cores and 1024 GiB of main memory. Each worker independently constructs its own context set of up to 2,048 traces, processes its own query batch, and computes local gradients. Gradients are accumulated across event systems and then averaged across workers by PyTorch DistributedDataParallel (DDP) (Li et al., 2020) before the parameter update. Cross-attention over each worker's context bank is implemented using FlashAttention-2.

Task-specific baselines. We train the local Transformer, GRU, Neural Hawkes, Transformer Hawkes, and IFTPP independently on the training split of each target dataset. For the Transformer and GRU, each activity is represented by a learned embedding. To accommodate multi-scale and potentially multimodal temporal distributions, the Transformer decomposes raw inter-arrival times into multiple time units, including days, hours, and minutes, following recent work on process-event prediction (Spaeh et al., 2026). The GRU instead applies a logarithmic transformation to raw inter-arrival times. Both models encode the resulting temporal representations using Time2Vec (Kazemi et al., 2019) followed by an MLP, and combine them with the activity embeddings to form event tokens. Both models use a linear classification head for next-activity prediction. The Transformer uses a factorized probabilistic time head over the day, hour, and minute components, whereas the GRU predicts a log-normal distribution for the next inter-arrival time.

All local models use AdamW, a batch size of 256, weight decay of 0.03, and gradient clipping with maximum norm 1.0. Training is conducted on the training split of each test dataset for at most 1,000 epochs, with early stopping after 15 consecutive validation evaluations without improvement. We evaluate six hyperparameter configurations for each model. Across the search grids, hidden dimensions are 64 or 128, learning rates are 10^{-3} or $3 \cdot 10^{-4}$, and dropout is 0 or 0.1 where applicable. The Transformer uses two or four layers with feed-forward dimensions of 128 or 256; GRU and Neural Hawkes use one or two layers; Transformer Hawkes uses two or four layers; and IFTPP uses one or two layers with three or five mixture components. Neural Hawkes, Transformer Hawkes, and IFTPP rescale time using the 90th percentile of the training inter-arrival times.

For each model–dataset pair, validation loss controls early stopping. Among the retained epochs, we rank validation activity accuracy in descending order and validation time MAE in ascending order, and select the epoch with the best equal-weight rank sum. We apply the same rank-sum criterion to select the hyperparameter configuration. The test set is evaluated only after model selection is complete. Final metrics are computed from five independently trained instances of the selected configuration, using different random initializations.

The Transformer and GRU are implemented directly in PyTorch. For Neural Hawkes, Transformer Hawkes, and IFTPP, we use the corresponding implementations from EasyTPP (Xue et al., 2024).

Foundation-model baselines. FM-v2 and the zero-shot FIM-PP baseline are evaluated without updating their pretrained parameters.[1] FM-v2 performs retrieval over prefixes from the target training partition, using the 20 nearest support prefixes for each query. It receives at most 2,048 support traces.

[1] We use the publicly available checkpoints for FM-v2 and FIM-PP.

Algorithm 1 Episodic pretraining of FEP

for each training episode **do**
  Initialize accumulated gradients to zero
  for each event dataset $\mathcal{D}_m \in \mathcal{P}$ **do**
    Sample context sequences $\mathcal{D}^C$ and query batch $\mathcal{B}_{\mathrm{Q}}$
    Causally encode $\mathcal{D}^C$ to construct context keys K_C and next-event values V_C
    Causally encode $\mathcal{B}_{\mathrm{Q}}$ to obtain query representations Q
    Compute $\mathrm{CrossAttention}(Q, K_C, V_C)$ and next-event predictions
    Compute $\mathcal{L}$ and backpropagate to accumulate gradients
    Release the current computation graph
  end for
  Update model parameters using the accumulated gradients
end for

For our evaluation, we use its next-activity predictions; its temporal prediction targets remaining case time rather than next-event inter-arrival time.

FIM-PP receives the same maximum number of support traces and predicts both the next activity and its time. The fine-tuned FIM-PP variant follows the model's target-system adaptation procedure using at most 500 traces sampled from the target training partition; the validation and test partitions remain unchanged. We use the official fine-tuning implementation released together with the FIM-PP checkpoint.

The released FIM-PP implementation supports at most 22 activity classes. For datasets containing more classes, we retain the 21 most frequent activities in the sampled support set and map all remaining activities to a single "other" class. We therefore report FIM-PP accuracy in this 22-class label space. Because collapsing multiple original activities into one class makes the prediction problem easier, this preprocessing favors FIM-PP in the comparison.

In-context evaluation. We evaluate on 50 test datasets, whose aggregate statistics are reported in Table 9b. FEP and the frozen foundation-model baselines use at most 2,048 traces from the target training partition as in-context support. When the training partition contains more than 2,048 traces, we evaluate five independently sampled context subsets; otherwise, each method uses all available training traces once. The fine-tuned FIM-PP evaluation uses five independently sampled fine-tuning sets subject to its 500-trace limit. Thus, variation for task-specific local models reflects independent training runs, whereas variation for frozen foundation models reflects sampling of the target-system context.

B Additional Results

B.1 Inference Efficiency and Scalability

Table 4 evaluates the efficiency and scalability of FEP's cached-context nonparametric inference. Because support-conditioned inference is expressed entirely as cross-attention, the same exact computation can be implemented either with standard attention, which explicitly materializes the attention score and probability matrices, or with FlashAttention-2, which performs exact attention using a fused, memory-efficient kernel. For FlashAttention-2, we explicitly select PyTorch's `SDPBackend.FLASH_ATTENTION` backend.

Measurements use a batch size of one, BF16 autocast, PyTorch 2.14.0 with CUDA 13.0, and one NVIDIA RTX PRO 6000 Blackwell Server Edition. Each query is a length-100 trace producing 99 next-event predictions. We perform seven independent timing runs, each averaging 50 CUDA-event-timed queries after 20 warm-up queries, and report the median across runs. Fixed-length synthetic inputs control trace length while preserving production tensor shapes and execution kernels. Context encoding is excluded because the encoded context is cached and reused across queries.

Table 4: Inference efficiency and scalability of standard attention and FlashAttention-2 (FA2) for cached-context nonparametric inference. Each context sequence and query trace has length 100, yielding 99 history-to-next-event transitions or predictions. Peak memory includes model parameters, cached context representations, query execution, and output predictions.

		Latency (ms)		Queries/s		Peak memory (GiB)	
# Context seq.	# Context K/V tokens	Standard	FA2	Standard	FA2	Standard	FA2
8	792	2.118	2.182	472.2	458.2	0.039	0.038
16	1,584	2.124	2.185	470.7	457.6	0.042	0.040
32	3,168	2.120	2.191	471.7	456.5	0.050	0.045
64	6,336	2.112	2.188	473.5	457.0	0.064	0.055
128	12,672	2.121	2.196	471.5	455.5	0.094	0.073
256	25,344	2.115	2.189	472.7	456.7	0.153	0.104
512	50,688	2.131	2.214	469.2	451.7	0.267	0.169
1,024	101,376	2.209	2.187	452.7	457.3	0.497	0.297
2,048	202,752	3.228	2.606	309.8	383.7	0.962	0.557
4,096	405,504	5.336	3.852	187.4	259.6	1.885	1.078
8,192	811,008	9.469	6.187	105.6	161.6	3.729	2.114
12,288	1,216,512	13.548	8.470	73.8	118.1	5.576	3.156
16,384	1,622,016	17.616	10.733	56.8	93.2	7.420	4.193

Table 5: Relative improvement of FlashAttention-2 over standard attention at large context sizes. Positive values indicate improvement.

# Context seq. (tokens)	Latency reduction	Throughput increase	Memory reduction
2,048 (203K)	19.3%	23.9%	42.1%
4,096 (406K)	27.8%	38.5%	42.8%
8,192 (811K)	34.7%	53.0%	43.3%
12,288 (1.2M)	37.5%	60.0%	43.4%
16,384 (1.6M)	39.1%	64.1%	43.5%

At small context sizes, both implementations have nearly constant latency because fixed execution costs dominate, with standard attention approximately 3–4% faster through 512 context sequences. FlashAttention-2 nevertheless already provides substantial memory savings in this regime. As the context grows, the latency trend reverses: FlashAttention-2 becomes faster at approximately 1,024 sequences, and its advantage increases steadily thereafter. At 2,048 sequences, FlashAttention-2 reduces latency by 19.3% and increases throughput by 23.9%; at 8,192 sequences, these gains grow to 34.7% and 53.0%, respectively. At the largest tested context of 16,384 sequences (1.62M context tokens), FlashAttention-2 reduces latency from 17.616 to 10.733 ms, increases throughput from 56.8 to 93.2 queries/s, and reduces peak memory from 7.420 to 4.193 GiB—corresponding to improvements of 39.1%, 64.1%, and 43.5%, respectively (Table 5).

These results directly support the scalability motivation of FEP: expressing support-conditioned nonparametric inference as standard attention allows FEP to directly benefit from optimized attention implementations. At fixed query length, exact cross-attention still requires computation linear in the context size, but FlashAttention-2 avoids materializing the full attention matrices, yielding increasingly substantial runtime and memory savings as the context grows.

Table 6: Effect of event-ID representation under the ablation protocol.

Setting	Accuracy (%)	NMAE
Random fixed	67.77 ± 0.03	0.80 ± 0.00
Learned + permutation	$\mathbf{67.95 \pm 0.03}$	$\mathbf{0.79 \pm 0.00}$
Learned, no permutation	64.14 ± 0.03	0.83 ± 0.00

Table 7: Performance of all four FEP configurations across dataset difficulty levels. The configurations combine two inter-arrival-time transformations (quantile and robust log) with two event-ID representation strategies (permutation training and random fixed embeddings).

(a) Next-event-type accuracy.

Δt Transform	Type Embedding	Easy	Moderate	Hard
Quantile	Permutation training	0.82 ± 0.00	0.64 ± 0.00	0.35 ± 0.00
Robust log	Permutation training	0.81 ± 0.00	0.64 ± 0.00	0.34 ± 0.00
Quantile	Random fixed	0.82 ± 0.00	0.64 ± 0.00	0.35 ± 0.00
Robust log	Random fixed	0.82 ± 0.00	0.64 ± 0.00	0.35 ± 0.00

(b) Next-event inter-arrival-time NMAE.

Δt Transform	Type Embedding	Easy	Moderate	Hard
Quantile	Permutation training	0.30 ± 0.00	0.53 ± 0.00	0.83 ± 0.00
Robust log	Permutation training	0.29 ± 0.00	0.54 ± 0.00	0.84 ± 0.00
Quantile	Random fixed	0.31 ± 0.00	0.54 ± 0.00	0.85 ± 0.00
Robust log	Random fixed	0.30 ± 0.00	0.55 ± 0.00	0.84 ± 0.00

B.2 Ablations on Event-ID Representations

Following the ablation protocol described in the main paper, all variants in Table 6 are pretrained on 33 datasets and evaluated on the 23 held-out datasets, with inter-arrival times represented using robust log normalization. We compare three strategies for constructing the event-type embeddings used by the causal self-attention layers. The first uses random fixed embeddings, freshly sampled in each training episode and not optimized. The second uses learned embeddings with permutation training, where the correspondence between event types and embedding IDs is randomly permuted in each episode. The third uses learned embeddings without permutation.

Both random fixed embeddings and learned embeddings with permutation training achieve strong and similar performance, with permutation training performing slightly better overall. In contrast, learned embeddings without permutation reduce accuracy by nearly four percentage points and also increase NMAE. These results indicate that preventing persistent semantics from being associated with numerical event IDs is important for cross-system generalization, while both random fixed embeddings and permutation training provide effective solutions.

B.3 Combinations of Event-ID Representation and Time Normalization

Table 7 reports all four combinations of event-ID representation and time normalization, with FEP pretrained on the full set of 696 datasets. The four configurations achieve close performance in both event-type accuracy and inter-arrival-time NMAE across all difficulty levels, with no clear winner. This suggests that FEP is robust to the particular choice between the two proposed event-ID representations and time-normalization strategies.

B.4 Dataset-wise Effect of Cross-Dataset Pretraining

Figure 4 compares FEP with FEP-Local on each test dataset. For event-type accuracy, most points lie below the diagonal, indicating higher accuracy for FEP, with substantial improvements on many datasets. For NMAE, most points lie above or close to the diagonal, likewise indicating lower or comparable time-prediction error for FEP. These dataset-wise results show that the gains from cross-dataset pretraining are broadly distributed rather than driven by only a few datasets. Consistent with this observation, Table 1 in the main paper shows that FEP substantially outperforms FEP-Local on average across all three difficulty levels. Together, these results demonstrate the benefit of cross-dataset pretraining for generalization to unseen event systems.

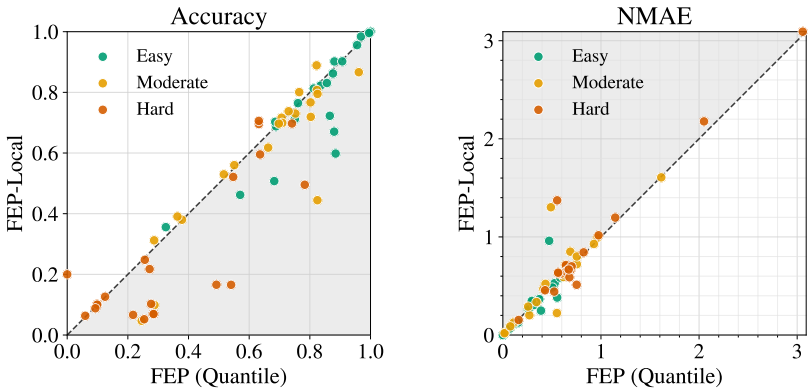


Figure 4: Dataset-wise comparison of FEP (Quantile) and FEP-Local on the 50 held-out datasets. Colors denote dataset difficulty, and the dashed line indicates equal performance. Points below the diagonal favor FEP for accuracy, whereas points above the diagonal favor FEP for NMAE. The FEP-favoring region is shaded in both panels.

Table 8: Accuracy and NMAE on six representative datasets, with two selected from each difficulty level. Results are reported analogously to Table 1; NMAE values above 10,000 are denoted by ∞.

	Medical Appointments		Beijing Air Quality		NASA SMAP/MSL	
Method	Accuracy	NMAE	Accuracy	NMAE	Accuracy	NMAE
Transformer	**0.96 ± 0.00**	0.30 ± 0.00	0.76 ± 0.02	**0.68 ± 0.01**	0.76 ± 0.01	1.04 ± 0.01
GRU	**0.96 ± 0.00**	0.30 ± 0.01	**0.80 ± 0.02**	**0.68 ± 0.01**	**0.79 ± 0.00**	**0.99 ± 0.00**
Neural Hawkes	**0.96 ± 0.00**	0.31 ± 0.01	0.78 ± 0.01	0.72 ± 0.02	0.79 ± 0.00	1.04 ± 0.05
Trans. Hawkes	**0.96 ± 0.00**	0.33 ± 0.03	0.74 ± 0.02	0.77 ± 0.06	0.26 ± 0.06	1.32 ± 0.19
IFTPP	**0.96 ± 0.00**	∞	0.76 ± 0.01	0.79 ± 0.02	0.78 ± 0.01	1.51 ± 0.55
FEP-Local	**0.96 ± 0.00**	**0.28 ± 0.00**	0.72 ± 0.02	0.71 ± 0.01	0.50 ± 0.22	1.02 ± 0.01
FM-v2	**0.96 ± 0.00**	—	0.36 ± 0.00	—	0.40 ± 0.00	—
FIM-PP	0.01 ± 0.00	0.36 ± 0.00	0.30 ± 0.00	0.88 ± 0.00	0.39 ± 0.00	2.19 ± 0.00
FIM-PP (FT)	0.47 ± 0.00	0.34 ± 0.00	0.43 ± 0.00	0.91 ± 0.00	0.75 ± 0.01	1.52 ± 0.04
FEP (Quantile)	**0.96 ± 0.00**	**0.27 ± 0.00**	**0.80 ± 0.00**	0.63 ± 0.00	**0.78 ± 0.00**	**0.98 ± 0.00**
FEP (Robust log)	**0.96 ± 0.00**	0.28 ± 0.00	0.77 ± 0.00	**0.61 ± 0.00**	0.77 ± 0.00	1.02 ± 0.00

	USGS Water		India Air Quality		US Drought Monitor	
Method	Accuracy	NMAE	Accuracy	NMAE	Accuracy	NMAE
Transformer	0.81 ± 0.01	0.02 ± 0.00	**0.72 ± 0.00**	0.09 ± 0.12	**0.74 ± 0.00**	0.82 ± 0.01
GRU	**0.82 ± 0.00**	**0.00 ± 0.00**	**0.72 ± 0.00**	**0.03 ± 0.00**	0.73 ± 0.01	**0.81 ± 0.01**
Neural Hawkes	**0.82 ± 0.00**	0.50 ± 0.19	**0.72 ± 0.00**	0.83 ± 0.08	0.73 ± 0.01	0.88 ± 0.05
Trans. Hawkes	**0.82 ± 0.01**	0.09 ± 0.05	**0.72 ± 0.00**	0.30 ± 0.21	0.73 ± 0.01	0.82 ± 0.10
IFTPP	0.70 ± 0.11	0.01 ± 0.00	0.71 ± 0.01	0.04 ± 0.01	**0.74 ± 0.00**	0.88 ± 0.01
FEP-Local	**0.82 ± 0.01**	**0.00 ± 0.00**	**0.72 ± 0.00**	**0.03 ± 0.00**	0.70 ± 0.01	0.84 ± 0.01
FM-v2	0.47 ± 0.00	—	0.58 ± 0.00	—	0.37 ± 0.00	—
FIM-PP	0.12 ± 0.00	0.99 ± 0.00	0.59 ± 0.01	0.66 ± 0.00	0.37 ± 0.00	1.09 ± 0.00
FIM-PP (FT)	0.49 ± 0.00	0.99 ± 0.00	0.69 ± 0.02	0.62 ± 0.03	0.32 ± 0.02	1.14 ± 0.00
FEP (Quantile)	**0.84 ± 0.00**	**0.00 ± 0.00**	**0.71 ± 0.00**	**0.02 ± 0.00**	**0.74 ± 0.00**	**0.82 ± 0.00**
FEP (Robust log)	**0.84 ± 0.00**	0.01 ± 0.00	**0.71 ± 0.00**	0.03 ± 0.00	0.71 ± 0.00	**0.82 ± 0.00**

C Dataset Information

The datasets used for pretraining and testing are collected from a variety of public online repositories, such as 4TU.ResearchData, Kaggle, Zenodo, NYC Open Data, and Data.gov. We process each dataset individually to conform to our event-sequence representation. If a dataset contains more than 100 activity types, we retain the 99 most frequent types and merge the remaining ones into a general

Table 9: Summary of the real-world datasets used for pretraining and evaluation, grouped by domain. Trace length and duration denote the mean number of events and total duration per case, respectively; all values are unweighted averages of per-dataset statistics within each domain. **Environmental monitoring is entirely held out from the pretraining pool**, providing evaluation on a previously unseen domain.

(a) Statistics of the 696 pretraining datasets.

Domain	#	#Cases	#Activities	Trace length	Duration (days)
Commerce, finance, and organizations	102	74 036.17	20.74	172.70	529.61
Healthcare and life sciences	38	8 076.03	20.68	138.02	1 709.82
Digital systems, communication, and media	173	24 235.52	38.31	585.44	739.68
Government and public administration	203	183 071.43	32.61	2 582.07	1 151.45
Industry, infrastructure, and energy	109	16 740.21	15.46	1 716.80	1 638.44
Mobility and transportation	71	38 966.23	28.38	2 747.08	395.72

(b) Statistics of the 50 held-out test datasets.

Domain	#	#Cases	#Activities	Trace length	Duration (days)
Environmental monitoring	**12**	**64 923.75**	**5.42**	**565.63**	**2 170.54**
Commerce, finance, and organizations	11	22 552.36	10.18	191.96	488.05
Healthcare and life sciences	6	5 298.17	25.00	11.49	685.63
Digital systems, communication, and media	6	162 867.17	10.50	4 266.10	166.48
Government and public administration	5	122 600.60	44.40	18.64	117.42
Industry, infrastructure, and energy	5	1 244.60	6.80	522.62	2 162.66
Mobility and transportation	5	408 464.80	42.00	3 709.90	50.13

"other" activity. The test datasets are verified to ensure that they do not overlap with the pretraining data in dataset or process identity, including duplicate or derived versions of the same underlying event log. The training split of each test dataset—used both for training local models and as the context set for foundation-model inference—is capped at 10,000 traces. If the split contains more than 10,000 traces, we randomly subsample 10,000.

Table 9 summarizes the pretraining and test datasets by domain. Notably, environmental monitoring is completely absent from the pretraining pool, enabling evaluation on a previously unseen domain. Figure 5 further compares the distributions of inter-arrival time, trace duration, number of activities, and trace length between the pretraining pool and the easy, moderate, and hard test-dataset groups. The test datasets exhibit noticeable distribution shifts relative to the pretraining data, particularly in inter-arrival times and activity counts for the hard group. Together, these differences make the test collection a challenging benchmark for evaluating cross-system generalization of foundation models.

Finally, Table 10 provides the full catalog of the 50 test datasets, including a brief description of each dataset and preprocessing details where applicable.

Table 10: Details of the 50 real-world test datasets. Trace length and duration (in days) are averaged over all traces within each dataset.

Dataset and source	#Cases	#Act.	Length	Duration
Environmental monitoring				
Beijing Air Quality (Chen, 2017)	12	5	4 387	1 218
Hourly PM2.5 observations from 12 Beijing stations. Values are discretized into six AQI categories and only transitions between categories are retained.				
EPA AQS Ozone (EPA AQS)	1 225	6	186.1	205.6
Hourly ozone measurements from US EPA monitoring sites. Concentrations are mapped to six AQI categories and only changes in category are emitted as events.				

Continued on next page

Dataset and source	#Cases	#Act.	Length	Duration
EPA AQS PM2.5 (EPA AQS)	1 029	6	57.60	290.9
PM2.5 observations from US EPA monitoring sites. Measurements are mapped to six AQI categories and ordered per monitoring station.				
EPA ECHO Enforcement (EPA ECHO)	107 695	6	14.90	4 248
EPA ECHO facility histories covering inspections, detected violations, enforcement actions, penalties, and return to compliance. Source tables are joined into one lifecycle per facility.				
EPA ICIS-Air (EPA ECHO)	138 133	4	11.20	2 889
Clean Air Act compliance histories from EPA ICIS-AIR. Inspections, violation determinations, high-priority violations, and resolutions are combined and ordered per facility.				
EPA NPDES (EPA ECHO)	211 120	5	9.70	2 055
NPDES water-compliance histories. Inspection, violation, informal/formal enforcement, and resolution records are combined into chronological facility traces.				
EPA RCRA (EPA ECHO)	266 298	3	6.60	1 610
RCRA hazardous-waste compliance histories. Facility evaluations, violation determinations, and enforcement actions are merged and ordered by date.				
India Air Quality (Kaggle)	52 226	6	22.40	0.912
Hourly Indian air-quality measurements grouped by station and day. PM2.5 is discretized into six AQI categories; each station-day forms one case.				
NOAA Coral Reef Watch (NOAA CRW)	1 080	10	14.10	355.6
NOAA Coral Reef Watch thermal-stress histories. Product observations are represented as ten bleaching or thermal-stress states and ordered for each monitored region-period case.				
US Drought Monitor (Svoboda et al., 2002)	51	6	113.3	7 602
Weekly drought conditions for US states. Area percentages are converted to the most severe drought category exceeding a coverage threshold, and only category changes are retained.				
USGS Streamflow (USGS NWIS)	16	4	433.1	4 039
Daily discharge histories for 16 major river gauges. Per-station quantiles define low, normal, high, and flood regimes, and only regime transitions are retained.				
USGS Water (USGS Water Data)	200	4	1 532	1 534
Daily USGS water-discharge histories for 200 gauges. Measurements are mapped to four flow regimes using station-specific thresholds and ordered per gauge.				
Commerce, finance, and organizations				
Amazon Reviews (McAuley & Leskovec, 2013)	21 504	5	9.60	371.1
Amazon food-review histories, with one case per reviewer and rating values as activities. Reviews are ordered by their original timestamps; simultaneous reviews are retained.				
Berka Banking (Berka & Sochorova, 1999)	4 500	5	212.5	1 118
Bank-account transaction histories; the benchmark exposes operation, transaction type, and purpose prediction tasks, while the displayed statistics use the operation task.				
DeFi Protocol Fees (DefiLlama)	2 100	7	444.7	498.2
Daily fee histories for decentralized-finance protocols. Percentage changes are discretized into seven categories ranging from fee collapse to fee surge.				
DeFi Protocol TVL (DefiLlama)	1 108	7	795.2	804.7
Daily total-value-locked histories for DeFi protocols. Relative TVL changes are binned into seven states from crash through surge.				
DEX Trading Volume (DefiLlama)	1 050	7	606.3	658.1
Daily trading-volume histories for decentralized exchanges. Relative changes are discretized into seven activity levels from volume crash through volume surge.				
Employee HR (Kaggle)	2 771	2	1.40	193.6
Employee lifecycles containing hire and, where observed, exit events. Raw employment dates are converted into chronologically ordered two-state traces per employee.				

Continued on next page

Dataset and source	#Cases	#Act.	Length	Duration
Hotel Booking Demand (Antonio et al., 2019) Hotel reservation lifecycles. Each booking is converted into reservation, cancellation or check-in, and check-out events using the supplied booking and arrival dates.	92 930	4	2.40	68.80
LeadForge Lead Scoring (Kaggle) CRM engagement histories for sales leads. Calls, emails, meetings, and demonstrations are retained as four ordered activity types per lead.	4 560	4	4.30	12.90
NYC Inspections (NYC Open Data) NYC restaurant inspection histories, with one case per establishment. Inspection and reinspection descriptions are normalized into 37 ordered activity types.	25 379	37	9.80	554.2
Olist Brazilian E-Commerce (Kaggle) Brazilian marketplace order lifecycles. Relational order, approval, shipping, and delivery timestamps are joined and converted into four ordered events per order.	88 824	4	3.90	12.60
Open Food Facts (Open Food Facts) Open Food Facts product-addition histories grouped by brand. Raw product categories are consolidated into 30 food-category activities and ordered by creation time.	3 350	30	21.50	1 077
Healthcare and life sciences				
ClinicalTrials.gov (ClinicalTrials.gov) Clinical-trial record lifecycles from registration through study completion and result submission. Available milestone dates are converted to eight ordered event types per trial.	9 425	8	6.80	1 898
FDA Drug Enforcement (openFDA) FDA drug-recall histories grouped by recalling firm. Recall classifications are represented as Class I, II, or III activities and ordered by report date.	842	3	17.30	536.8
FDA FAERS (openFDA) FDA adverse-event reports grouped by suspect drug. Reactions are restricted to the 50 most frequent MedDRA preferred terms and ordered by report date.	2 042	50	19.80	552.0
FitLife (Kaggle) Participant fitness histories containing ten workout types. Sessions are ordered by their recorded dates;	12 000	10	18.40	1 110
Medical Appointments (Kaggle) Patient appointment lifecycles. Scheduling records are converted into scheduled, check-in, completed, and no-show events and ordered per patient.	6 830	4	2.90	16.80
MIMIC-II (Saeed et al., 2011) Temporal-point-process benchmark with 75 diagnosis/event types. We retain its published preprocessing and scaled event times rather than rebuilding traces from raw MIMIC tables.	650	75	3.72	0.0000201
Digital systems, communication, and media				
ATP Tennis Tournaments (Tennis Abstract) ATP player–tournament journeys. Match records are converted to round/outcome events (for example, quarter-final won or lost) ordered within each tournament journey.	811 988	32	1.90	0
CIC-IDS2017 (Sharafaldin et al., 2018) Network-security traffic from CIC-IDS2017. Grouped into host-level temporal traces and six attack/activity classes; the resulting traces are segmented into training examples.	3	6	25 421	1.40
Enron Email (CALO Enron) A case-centric view of the Enron email corpus, with one case per email address. Message actions are mapped to six email activities such as default, response, forwarding, and reminder.	1 262	6	14.40	128.1
MovieLens 25M (Harper & Konstan, 2015) MovieLens user rating histories. Each user is a case and half-star ratings from 0.5 to 5 are treated as ten categorical activities ordered by rating timestamp.	159 025	10	152.5	216.8
Stack Exchange Questions (Stack Exchange) Stack Exchange question lifecycles. Question, answer, comment, acceptance, closure, and related actions are combined into seven ordered event types per question.	933	7	5.00	651.5

Continued on next page

Dataset and source	#Cases	#Act.	Length	Duration
TensorFlow GitHub (GitHub)	3 992	2	1.50	1.10
TensorFlow GitHub issue lifecycles. API records are converted into opened and closed events per issue and ordered by their GitHub timestamps.				
Government and public administration				
BPI Challenge 2015 (van Dongen, 2015)	1 097	100	43.00	94.70
Environmental-permit workflows from five Dutch municipalities. We retain the original event ordering and use full activity codes.				
US Congressional Bills (GovInfo)	63 928	9	5.20	44.00
US congressional bill histories. Detailed BILLSTATUS actions are mapped to nine legislative stages, such as introduction, committee, floor, presidential action, and enactment.				
NSF Awards (NSF)	47 140	10	3.20	345.7
National Science Foundation award histories grouped by principal investigator. Funding programs are used as the activity, restricted to ten aggregated program groups in this benchmark.				
NYC 311 (NYC Open Data)	500 000	3	2.30	17.90
NYC 311 service-request lifecycles. Requests are sampled to 500,000 cases and represented by created, in-progress, and closed events inferred from available status timestamps.				
WABO Permits (4TU.ResearchData)	838	100	39.50	84.80
Receipt-phase environmental-permit workflows from a Dutch municipality. Original detailed process codes and timestamps are retained and ordered per permit application.				
Industry, infrastructure, and energy				
Backblaze SMART (Backblaze)	4 334	4	2.20	34.10
Daily hard-drive SMART observations, with one case per drive. We discretize into healthy, minor degradation, major degradation, and failure, and repeated unchanged states are collapsed.				
EIA Electricity Generation (EIA)	308	13	143.0	9 904
Annual US electricity-generation records grouped by state and producer type. Fuel or energy source is used as the event activity and records are ordered chronologically.				
NASA SMAP/MSL (Hundman et al., 2018)	81	6	2 452	21.50
NASA SMAP and Mars Science Laboratory telemetry channels. Continuous telemetry is converted into six nominal/deviation states, producing one ordered trace per channel.				
Pipeline Incidents (PHMSA)	192	7	12.40	854.2
PHMSA hazardous-liquid pipeline incidents grouped by operator. Detailed cause fields are consolidated into seven incident-cause activities and ordered by incident date.				
UCI SECOM (McCann & Johnston, 2008)	1 308	4	3.00	0.00694
Semiconductor manufacturing cases. Each wafer is represented by inspection start, measurement completion, and pass/fail classification events separated by fixed five-minute offsets.				
Mobility and transportation				
BART Ridership (BART)	50	2	15 966	15.60
Hourly BART passenger flows, with one case per station. Origin–destination counts are represented as arrival and departure events at their corresponding station and hour.				
BTS On-Time Flights (BTS)	2 019 872	6	4.00	0.138
Domestic US flight lifecycles. Scheduled records are converted into gate departure, takeoff, landing, gate arrival, cancellation, and diversion events for each flight.				
GeoLife GPS (Microsoft GeoLife)	142	100	1 004	145.1
Microsoft GeoLife user trajectories. GPS points are mapped to roughly 1-km grid cells, consecutive repeats are collapsed, and all but the 100 most visited cells are mapped to OTHER.				
NYC Taxi (NYC TLC)	255	2	1 561	75.60
NYC yellow-taxi zone activity. A sample of trips is converted into pickup and drop-off events and regrouped as chronological traces for each taxi zone.				

Continued on next page

Dataset and source	#Cases	#Act.	Length	Duration
OpenSky Flights (Zenodo)	22 005	100	14.10	14.20
OpenSky aircraft route histories for October 2022. Destination airports form the activities; the 100 most frequent are retained and the remainder are mapped to OTHER.				

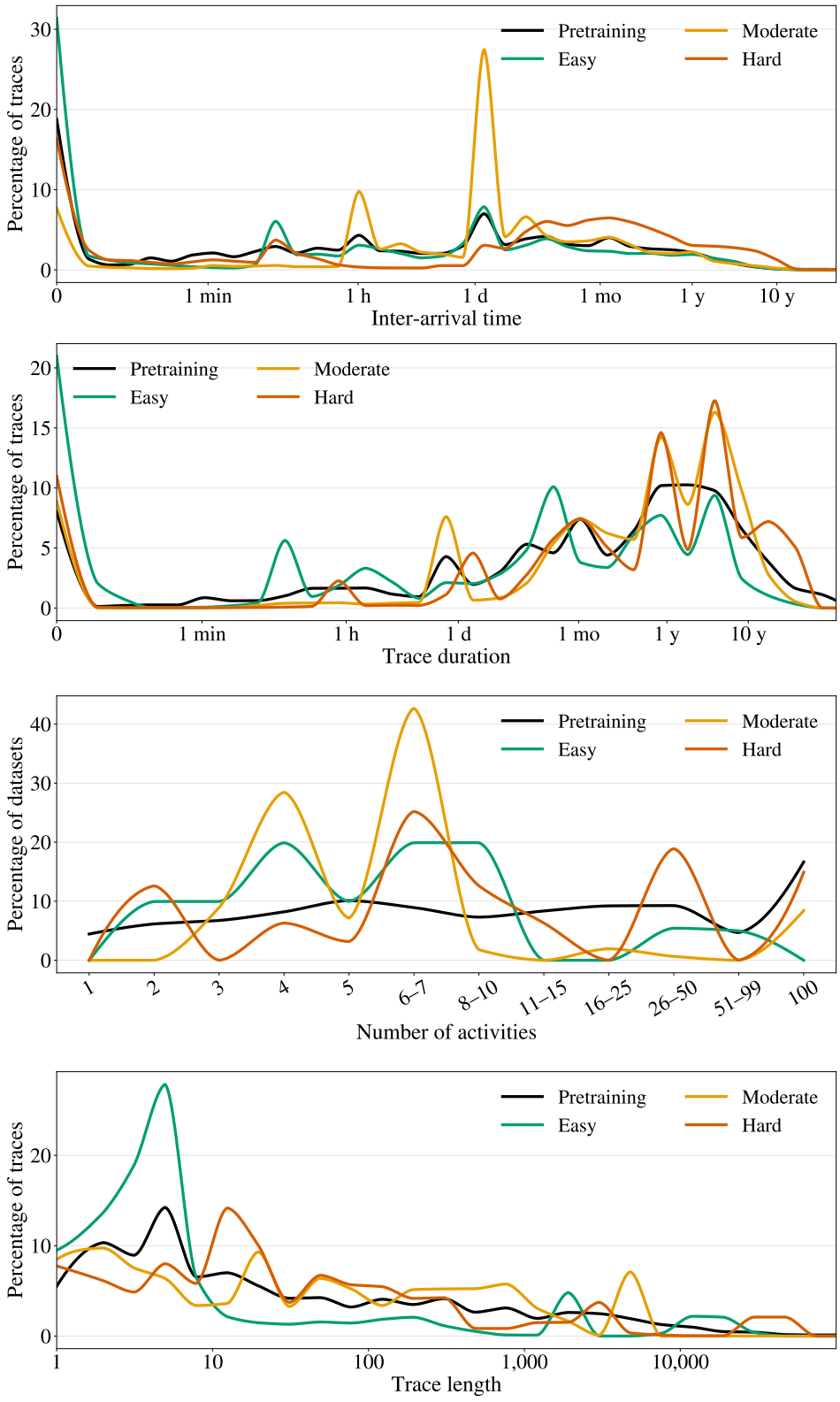


Figure 5: Interarrival times, trace durations, activity counts, and trace lengths per difficulty in the test datasets, and among the pretraining datasets. Interarrival times, trace durations, and trace lengths are per-dataset averages.