Retrieval-Augmented In-Context Foundation Model for Predictive Process Monitoring

Alessandro Berti * and Wil M.P. van der Aalst

Posted Date: 9 July 2026

doi: 10.20944/preprints202607.0705.v1

Keywords: predictive process monitoring; foundation models; process mining; retrieval-augmented in-context learning

 doi:10.20944/preprints202607.0705.v1

Article

Retrieval-Augmented In-Context Foundation Model for Predictive Process Monitoring

Alessandro Berti [1,*] and Wil M.P. van der Aalst [1,2]

[1] Process and Data Science Group, RWTH Aachen University, Ahornstrasse 55, Aachen, 52074, NRW, Germany
[2] Aelonis SE, Theresienstraße 6, Munich, 80333, Germany
* Correspondence: a.berti@pads.rwth-aachen.de

Abstract

Event logs record enterprise service processes as sequences of timestamped activities. Predictive process monitoring typically trains a separate model per event log, requiring retraining when processes, activity vocabularies, or time scales change. We study an event-log-native foundation model pretrained once on heterogeneous logs and adapted to an unseen log with a small labeled support set, without fitting a separate model for that log. The model represents prefixes with a mixture-of-experts transformer and predicts next activity and remaining time with a support-conditioned prototypical head whose label space is defined by the support context. To align training with deployment, we incorporate retrieval-centric objectives that shape the representation for nearest-neighbor support selection and we provide confidence estimates for both classification and regression. We benchmark this no-fine-tuning setting on five public logs against classical CPU-friendly baselines and a general-purpose in-context tabular predictor, down to 0.5% of training cases. Results show that one pretrained predictor can be competitive, but performance depends on retrieving suitable supports; in several settings, simple kNN on learned embeddings matches the full head. We also find that activity-time dependence is informative in this benchmark, and that confidence scores support useful performance stratification and expert-level representation analysis.

Keywords: predictive process monitoring; foundation models; process mining; retrieval-augmented in-context learning

1. Introduction

Enterprise service systems (e.g., ERP, CRM, ITSM) record work as *event logs*: each case (order, ticket, application) becomes a time-ordered sequence of timestamped activities. This data is the basis of process mining [1] and of *predictive process monitoring*, which maps a running case prefix to targets such as the next activity and the remaining time until completion.

Despite many proposed predictors, most deployments still require a separate supervised model per log. In practice, this approach is costly: activity vocabularies and naming conventions differ across systems, time scales vary by orders of magnitude, and processes evolve, triggering retraining and revalidation. Moreover, many settings are *low-data* (new services, rare variants, newly instrumented processes), where task-specific training is unstable or infeasible.

Our aim is not to replace every log-specific predictor. Strong sequence models [2,3] remain useful when enough labeled data, training time, and validation effort are available for one log. We study a complementary setting: train once, then apply to a new log with only a small labeled support set. This makes two questions central: whether prefix representations transfer across logs, and whether retrieved supports cover the local behavior needed for the query.

This paper studies retrieval-augmented prediction with an event-log foundation model, *FM-v2*[1], and an assessment protocol that mirrors deployment on unseen logs. At prediction time, we build a

[1] Significantly updated version compared to *FM-v1* [4]. The updates are explained in Section 4.8.

 doi:10.20944/preprints202607.0705.v1

pool of labeled prefixes from the target log. This pool is formed only from the available labeled cases (the support set), while queries come from held-out (or future) cases that are excluded from the pool. For each query prefix, FM-v2 encodes the prefix, retrieves its k nearest neighbors from this pool, and predicts conditioned on the retrieved supports. For next-activity prediction, FM-v2 does not carry a fixed output vocabulary from pretraining. Candidate next activities are the local activity IDs found in the retrieved target-log supports; therefore, the classification head scores only the activity IDs present among the retrieved supports. If the true next activity is not retrieved, it cannot be predicted. For remaining-time prediction, the retrieved prefixes act as local anchors in the target log, avoiding per-log retraining. Figure 1 sketches the retrieval-augmented inference pipeline of FM-v2 on an unseen log.

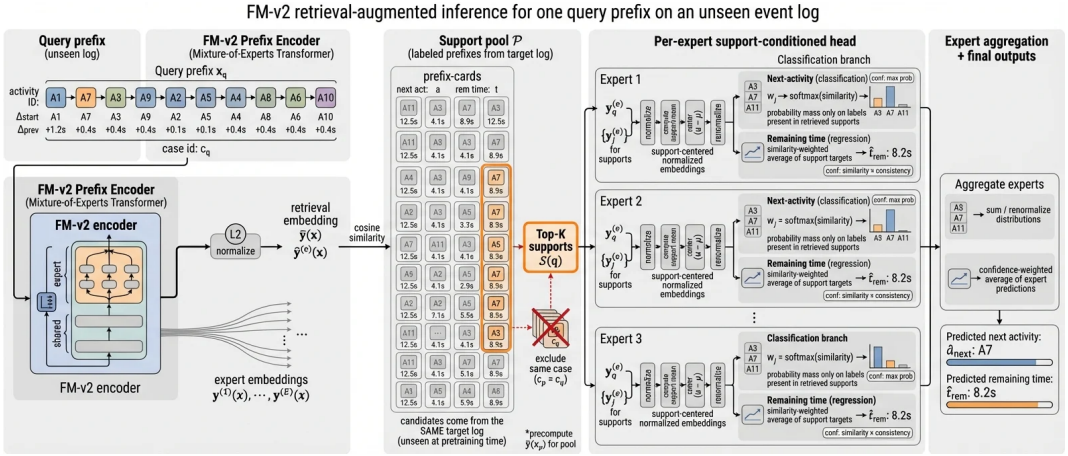


Figure 1. FM-v2 inference for one query prefix on an unseen log. The query is encoded and matched by cosine similarity against a pool of labeled prefixes from the same target log (excluding the same case), yielding a top-K support set. For each expert, normalized and support-centered embeddings drive a support-conditioned head: classification assigns probability mass only to labels present in the retrieved supports, while regression predicts remaining time as a similarity-weighted average of support targets. Expert outputs are aggregated to produce final predictions and confidence scores.

To align training with deployment, FM-v2 is trained with retrieval inside the training batch. In each update, the current batch serves as the candidate pool: each query prefix retrieves supports by nearest-neighbor search in the current embedding space and is trained using the same support-conditioned head used at inference. Throughout training and evaluation, retrieval excludes candidates from the same case as the query to prevent leakage. We assess FM-v2 as a no-fine-tuning transfer predictor on five public event logs, with CPU-friendly baselines and an in-context tabular predictor across data regimes down to 0.5% of training cases. We then analyze when transfer works, how confidence relates to performance, and what is captured by expert representations.

The goals of the assessment are operationalized by the following research questions:

- **RQ1:** In a no-fine-tuning, support-conditioned setting, is an event-log foundation model competitive with the selected CPU-friendly and in-context baselines on next-activity and remaining-time prediction, especially with little data?
- **RQ2:** Which event-log characteristics can predict when the pretrained model will outperform or underperform baselines on a previously unseen log?
- **RQ3:** Is the confidence score returned by the model correlated with predictive performance (accuracy for classification, MAE for regression), and does it enable bucket-wise performance stratification?
- **RQ4:** Are representations learned by individual experts competitive with handcrafted representation baselines, and how do experts differ in terms of within-expert and between-expert structure?

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces preliminaries and notation. Section 4 presents our retrieval-augmented foundation model

 doi:10.20944/preprints202607.0705.v1

FM-v2. Section 5 evaluates the approach empirically. Section 6 discusses the findings. Section 7 concludes the paper.

2. Related Work

We position FM-v2 within predictive process monitoring, pretrained event-log models, and support-conditioned prediction. These areas share a tension: strong PPM models are often trained for one log, while practical use on a new log may need predictions with little labeled data and no new model training. FM-v2 studies this second setting.

2.1. Predictive Process Monitoring

Predictive process monitoring uses prefixes of running cases to forecast process behavior. Common targets include case outcome, next activity, future events, and remaining time. The Process Mining Handbook introduces these tasks and their role in process mining [5]. Surveys and benchmarks show that results depend on prefix construction, labels, temporal splitting, encoding, and learner choice [6,7].

Many PPM approaches turn a prefix into fixed tabular features and train a classifier or regressor for one log. Typical encodings are index-based, aggregation-based, or hybrid, combined with tree ensembles, support vector machines, nearest-neighbor methods, and related tabular learners [8,9]. These models can be effective and inspectable, but they are usually tied to the activity labels, timing scale, and case distribution of their training log.

Recent work also stresses interpretability, explainability, and task-specific feature design. Mehdiyev et al. survey interpretable and explainable methods for PPM, showing that accuracy alone may be insufficient for operational decisions [10]. Guo et al. propose a feature-informed cascade for remaining-time prediction, showing how process features can support an effective and explainable predictor [11]. FM-v2 is complementary: it reuses a pretrained model on a new log through support examples, while analyzing confidence and representations to make predictions easier to inspect.

Neural sequence models reduce hand-designed prefix encodings by learning from event sequences. LSTM PPM models are central references for next-activity and time prediction, and transformer PPM models add attention over prefix events [2,3]. They are strong when enough labeled data and training time exist for the target log, but they are usually trained or fine-tuned per log. FM-v2 studies a different setting: parameters are learned once, and a new log is handled through labeled support prefixes rather than gradient updates.

2.2. Pretrained Event-Log Models

Pretraining for event logs seeks representations reusable across logs or tasks. This relates to foundation models in language and tabular data, where a model trained once is adapted through prompts or small task information [12,13]. For event logs, in-context foundation models for PPM show that labeled target-log examples can guide prediction on an unseen log without fitting a full log-specific model [4].

FM-v2 follows this transfer idea but targets a specific gap. A reusable encoder is not enough when output labels are local and prediction must be inferred from target-log examples. FM-v2 therefore makes support retrieval part of prediction. It aligns training and inference around query-support comparison, adds support-conditioned heads, and analyzes expert representations that may capture different similarity views.

Transfer across event logs is harder than across datasets with a shared label set. Activity identifiers are local: similar work can have different names, and the same name can mean different things across processes. Per-log models avoid this with a separate output layer for each log. A support-conditioned model keeps reusable prefix representations but lets the target-log supports define candidate next-activity labels. Retrieval quality and support coverage therefore become central.

 doi:10.20944/preprints202607.0705.v1

2.3. In-Context Prediction and Mixture-of-Experts

In-context prediction conditions on a small example set at inference time, rather than only on learned weights [12]. Metric-based few-shot learning offers a simple analogy: a query is compared with labeled examples or prototypes, and prediction uses the label space induced by them [14]. In FM-v2, the support set is a set of labeled target-log prefixes, not a natural-language prompt. Retrieved neighbors serve as evidence for next-activity classification and as local anchors for remaining-time regression.

This design adapts quickly but depends on the support pool. A query can use only behavior represented among its retrieved examples. In next-activity prediction, a missing correct label in the retrieved support set is a hard candidate-set failure. In remaining-time prediction, poor retrieval can select anchors with mismatched timing. Thus support-pool construction, neighbor retrieval, and support-conditioned prediction must be considered together.

Mixture-of-experts models add capacity by combining alternative expert transformations, often through gating or aggregation [15]. In transferable PPM, the motive is not only scale. Logs may emphasize different patterns, such as control-flow similarity, temporal similarity, or local activity context. Expert representations give the model multiple spaces for comparing a query with its supports.

The approach combines reusable prefix encoding, target-log retrieval, support-conditioned classification and regression, confidence estimation, and expert-level comparison. Related work studies many of these ideas separately. FM-v2 combines them for no-retraining PPM. This motivates component ablations for experts, support conditioning, retrieval-oriented training, and pretraining.

3. Preliminaries

Predictive process monitoring predicts running cases, including outcome, remaining time, and future activities. The Process Mining Handbook introduces these tasks and the distinction between model-based and machine-learning approaches [5]. Recent work also stresses explainability and feature design, for example through surveys of interpretable PPM and feature-informed remaining-time prediction [10,11]. We focus on prefix-level next-activity and remaining-time prediction without training a separate model for each target log.

This differs from the usual per-log pipeline. A conventional PPM model is trained or fine-tuned on one log and then used on prefixes from that log. FM-v2 instead uses a pretrained encoder plus a small labeled support set from the target log. The goal is fast example-based adaptation, not dominance over every task-specific LSTM, transformer, or other neural PPM model.

3.1. Logs and Prefixes

An *event log* contains many *cases*. A case records one process execution, such as an order, ticket, or application, as a time-ordered sequence of *events*. Each event has at least an activity name and timestamp. Some logs also include attributes such as resource or cost; when used, missing fields receive neutral values.

At prediction time, the full case is not available. We observe only the part already executed, called a *prefix*. The prefix is the model input and describes the running case through the event sequence seen so far and its timing pattern.

Cases have different lengths. To keep batches manageable, FM-v2 uses a maximum prefix length: longer prefixes keep only the most recent events, while shorter ones are padded only for batching and marked as padding. This fixes input size while preserving the history nearest to the prediction point.

3.2. Prediction Targets

We study two tasks. In *next activity prediction*, the model receives a running prefix and predicts the activity of the next event in the same case. This is classification over the target-log activity labels.

 doi:10.20944/preprints202607.0705.v1

In *remaining time prediction*, the same prefix is used to predict the time left until the case ends. This is regression over a duration. For a completed historical case, the target is the time from the last observed prefix event to the final event.

Both tasks are prefix-level. One case can yield several examples, one per prediction point. For next activity, a prefix is used only if a later event exists. For remaining time, each prefix is labeled with the observed duration until case completion.

3.3. Model Input

FM-v2 receives each prefix as an ordered list of events from the target log. In the reported experiments, the input signal is local to that log and uses event order together with timing values: elapsed time since case start and time since the previous event. Optional numeric values, such as cost, use neutral defaults when absent. A logarithmic transform compresses large numeric values before they enter the model.

The next-activity label space is local to each log. FM-v2 does not require a shared activity dictionary across logs. Unknown output labels are handled through the support-set label space described below.

The model reads each prefix as an ordered sequence of event vectors. Positional information marks event order. A transformer encoder combines a learned summary token with attention over the event tokens to produce the prefix representation used for retrieval, support comparison, and prediction.

3.4. Examples, Query, and Support

The basic prediction unit is a prefix with its target, called an *example*. The example to predict is the *query*. Labeled examples available for comparison form the *support pool*. The model searches this pool and selects similar examples, called the *support set*.

For a new target log, the support pool is built from its available labeled cases. Each eligible prefix is added with its next-activity label or remaining-time value. A query prefix from a held-out or future case is matched to this pool. The pretrained parameters stay fixed; adaptation comes only from selected target-log supports.

The requested support size sets how many neighbors are retrieved. Small values give a narrow local context; larger values improve coverage but may include less similar examples. If fewer valid examples exist, all valid ones are used. If none exists, the support-conditioned head cannot predict the query.

To avoid leakage, a query cannot retrieve prefixes from the same case. This rule applies both with target-log supports and during simulated retrieval in training. The model may use similar historical cases, but not another prefix from the case being predicted.

3.5. Local Activity IDs and Label Space

Activity labels differ across logs: similar steps may have different names, and activity sets vary. Therefore FM-v2 does not use one global output layer for next activity prediction. Each log defines *local activity IDs*, valid only inside that log, as the labels attached to support and query prefixes.

For a new log, activity names are mapped to local IDs before building supports. The support pool and query labels share this map. After prediction, the local ID is mapped back to the target-log activity name. Thus, the target log need not share activity IDs with any previous log.

For next activity prediction, the retrieved support set defines the final label space. The model assigns probability only to activities that appear as next-activity labels among the retrieved supports. If the correct label is absent, it cannot be chosen at that step. This makes support coverage central to classification performance.

For remaining time prediction, the support set is used differently. The model combines the remaining-time values attached to retrieved examples instead of choosing activity labels. Regression

 doi:10.20944/preprints202607.0705.v1

is not limited by label coverage, but it still depends on retrieving examples with relevant remaining times.

4. Approach

FM-v2 is a retrieval-augmented foundation model for event logs. It is pretrained once and applied to a new log through labeled support prefixes from that log. At prediction time, it encodes a query prefix, retrieves related supports, and predicts from them. Its components are the event-log encoder, retrieval representation, support-conditioned head, expert spaces, confidence estimates, and a training objective aligned with retrieval-based inference.

The contrast with a tabular PPM pipeline appears in Figure 2. A tabular model flattens each prefix into fixed features and learns a log-specific classifier or regressor. FM-v2 keeps the prefix as a sequence and fits no new target-log output layer. Retrieved supports provide the local output space and time anchors.

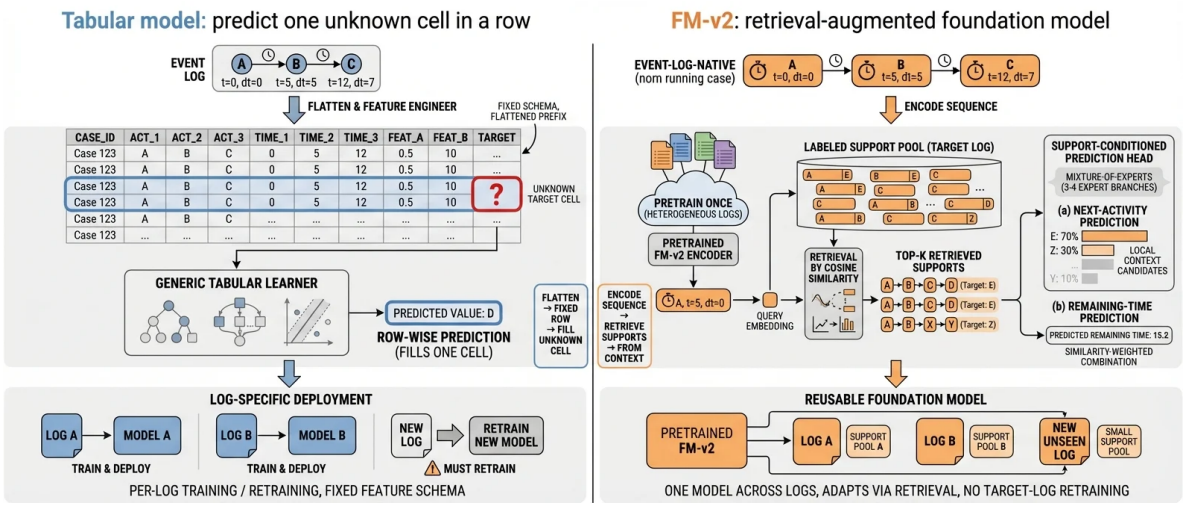


Figure 2. Tabular prediction vs. FM-v2 for predictive process monitoring. Left: a prefix is flattened into a fixed-size feature row for a log-specific tabular model. Right: FM-v2 keeps the sequential prefix form, retrieves similar labeled prefixes from the target log, and predicts without target-log retraining.

4.1. Encoding

FM-v2 maps a prefix to a sequence representation and then to a retrieval representation. In the reported experiments, this step stays local to each log. Timing values are transformed, projected to the model dimension, and processed with the event sequence by a transformer encoder.

TWOThe reported results evaluate the retrieval-trained FM-v2 sequence model with local target-log supports. They should be read as a retrieval-based evaluation of the support-conditioned model described here.

The encoder output has two uses. First, a normalized retrieval representation searches the support pool. Second, each expert produces its own prefix representation for the support-conditioned head after retrieval. At model level, the retrieved support set is shared, while experts compare the query and supports in their own spaces.

On a new log, FM-v2 keeps model parameters fixed. No target-log fine-tuning is used here. The target log affects prediction only through its local label map and available support examples.

4.2. Retrieval on the Target Log

For each target log, FM-v2 builds a support pool of labeled prefixes, computes their normalized representations once, and compares each query representation with them by cosine similarity. Prefixes from the same case are masked, and the top-k valid matches form the support set used by the prediction head.

 doi:10.20944/preprints202607.0705.v1

Top-k controls locality. Too small a value may miss relevant labels or remaining-time values; too large a value may include prefixes from different behaviors. The method therefore depends on support-pool size and the quality of the learned retrieval representation.

Three retrieval failure modes matter. First, same-case exclusion can remove all candidate supports. Second, in next-activity prediction, the true label is unreachable if absent from the retrieved set. Third, in remaining-time prediction, diverse support targets can make the weighted average unstable. These are direct consequences of support-conditioned prediction.

4.3. Next Activity from Support

For next activity prediction, each retrieved support prefix has a known next-activity label. The head normalizes the query and support representations, centers them using the support set, and normalizes them again. It then compares the query with each support and assigns larger weights to closer supports.

The head does not score a global activity list. It groups support weights by the local activity labels in the retrieved set. Each activity score is the total weight of supports with that label, with an optional count adjustment. The predicted activity is the support label with the largest score.

Classification confidence is the probability mass assigned to the selected activity after grouping support weights by label. It is high when most support weight favors one next activity, and low when similar supports point to different activities.

TWOThe implementation turns support similarities into attention weights, sums the weights per retrieved local label, and predicts the label with the largest mass. The confidence score is the largest final label probability.[2]

4.4. Remaining Time from Support

For remaining time prediction, each retrieved support prefix has a known remaining-time value. The same centered and normalized comparison gives weights over supports. The prediction is their weighted average, so closer supports receive more weight.

Regression confidence combines similarity and agreement. It is higher when a retrieved support is close to the query and when support remaining-time values concentrate near the prediction. It is lower for distant or conflicting supports.

TWOThe regression head uses similarity weights over retrieved supports. It predicts a weighted average of their remaining-time values. Confidence is higher when supports are close to the query and agree with each other.[3]

4.5. Experts and Confidence

FM-v2 uses multiple experts. Each expert has its own encoder and support-conditioned head, giving a different representation space for the same prefix. This lets the model compare query and supports in several ways, for example by control-flow similarity or timing patterns.

At inference time, expert outputs are combined. For next activity, the model aggregates expert label scores and chooses the largest combined score; confidence is the largest normalized label mass. For remaining time, expert estimates are weighted by expert confidence, and final confidence summarizes average expert confidence.

---

[2] TWOImplementation detail for classification: after normalization and support centering, the query vector z_q and support vectors z_j use $a_j = \mathrm{softmax}_j(\tau_c z_q^\top z_j)$ with $\tau_c = 5.0$, clamped to $[1, 20]$. For each local class c in support set S, $m_c = \sum_{j \in S: y_j = c} a_j$. The class logit is $\ell_c = \log(\max(m_c, 10^{-8})) + \beta \log n_c$, where n_c is the number of supports with class c and β is a learned count prior initialized at 0. Final probabilities are $p_c = \mathrm{softmax}_c(\ell_c)$. Prediction is $\arg\max_c p_c$, and confidence is $\max_c p_c$.

[3] TWOImplementation detail for remaining time: $w_j = \mathrm{softmax}_j(\tau_r z_q^\top z_j)$, where τ_r starts at 5.0 and is clamped to $[1, 100]$. The prediction is $\hat{t} = \sum_j w_j t_j$. Support disagreement is $\sigma_q = \sqrt{\sum_j w_j (t_j - \hat{t})^2 + 10^{-8}}$. The consensus term is $c_{cons} = 1/(1 + \sigma_q)$, and the similarity term is $c_{sim} = \mathrm{clip}((\max_j z_q^\top z_j + 1)/2, 0, 1)$. Regression confidence is $\mathrm{clip}(c_{cons} c_{sim}, 0, 1)$.

 doi:10.20944/preprints202607.0705.v1

TWOAt model level, FM-v2 combines the experts after each expert has produced its own prediction and confidence. For classification, the class scores are aggregated and normalized. For regression, the final estimate gives more weight to experts with higher confidence.[4]

This separation supports analysis. Direct nearest-neighbor prediction on learned embeddings tests the representation itself. The support-conditioned head tests whether a learned head improves over this non-parametric use of the same retrieved examples. Expert analyses show whether experts organize prefixes similarly or differently.

4.6. Training Loop

FM-v2 is trained with the same retrieval structure used at inference. In each step, a batch of labeled prefixes is sampled. Each prefix can be a query, and other prefixes in the batch are candidate supports, with same-case prefixes excluded. The query retrieves supports from the batch representation, and the loss is computed from that support set.

For next activity prediction, batches are sampled so that labels have prefixes from more than one case when possible. A query retrieves same-label examples from other cases plus negative examples, including hard neighbors and broader samples. The support-conditioned head predicts among labels in this training support, using cross-entropy on the query label.

For remaining time prediction, each query retrieves nearby prefixes from other cases in the batch. The head predicts a weighted average of their remaining-time values, and a robust error loss compares it with the observed remaining time. Training therefore focuses on support-based interpolation, not a fixed per-log regression layer.

The objective also shapes retrieval. For classification, an auxiliary contrastive term pulls same-label prefixes from different cases together and pushes other labels apart. For regression, a related term pulls prefixes together when remaining-time values are close. A neighbor-mass term can reward high same-label nearest-neighbor probability, while variance and covariance regularization discourage collapse.

These losses both reward correct predictions and make the embedding space better for nearest-neighbor support selection. The model is optimized for the same sequence used on unseen logs: encode the query, retrieve supports, and predict from them.

TWOThe reported run uses retrieval-style training with fixed loss and optimizer settings. Exact values are kept in a footnote for reproducibility.[5]

4.7. Applying FM-v2 to a New Log

Applying FM-v2 to a new log has four steps. Map activity names to local IDs, build the support pool from available labeled prefixes, and predict each query from its top-k valid supports. No global activity-ID alignment across logs is needed.

This procedure makes local activity IDs compatible with a pretrained model, because output labels are interpreted only through the target-log support set. The cost is that next-activity prediction can choose only retrieved labels. Classification needs good support coverage, and both tasks need good neighborhood quality.

---

[4] TWOImplementation detail for expert aggregation: let $p_e(c)$ be the class probabilities from expert e. Classification sums these vectors, applies L^1 normalization, and uses the largest normalized value as confidence. For regression, let $\hat{t}_e$ and c_e be expert e's prediction and confidence. The final prediction is $\hat{t} = \sum_e c_e \hat{t}_e / \max(\sum_e c_e, 10^{-8})$, and final confidence is the mean expert confidence.

[5] TWOImplementation detail for training: batches contain 128 prefixes. During training, k is increased from 12 to 32 in the first 8 epochs and then fixed at 32. Classification uses cross-entropy with label smoothing 0.05; regression uses Huber loss. The total loss is the mean prediction loss plus auxiliary terms. After the 10-epoch ramp, auxiliary weights are $0.25\mathcal{L}_{con}$, $0.10\mathcal{L}_{knn}$ for classification, and $0.03\mathcal{L}_{var} + 0.03\mathcal{L}_{cov}$. The contrastive temperature is 0.10. Classification uses two positive supports when possible, and regression uses the two nearest targets as positives in the target-neighborhood contrastive term. The hard-negative pool factor is 8, and the random-negative fraction drops from 0.60 to 0.15 in the first 8 epochs. Training uses 4 experts, 45 epochs, 300 episodes per epoch, AdamW with learning rate 10^{-4} and weight decay 0.01, gradient clipping at 1.0, and cosine learning-rate decay to 10^{-6}.

 doi:10.20944/preprints202607.0705.v1

4.8. What Changes from FM-v1 to FM-v2

FM-v2 extends *FM-v1* [4] by making retrieval central in inference and training. Like FM-v1, it is pretrained once and applied to unseen logs through labeled target-log prefixes. The main change is that each prediction uses an explicit top-k support set retrieved from the target log.

At inference, each query prefix is matched to labeled target-log prefixes, and prediction uses the retrieved set. During training, retrieval also occurs inside the batch and uses the same support-conditioned head. The objective therefore learns both the task and a representation suited to nearest-neighbor retrieval.

The intended improvement is not just a larger model, but a closer match between training and use on a new log. FM-v2 should be read as a retrieval-augmented transfer model for low-data, no-retraining settings, not as a claim that per-log sequence models are obsolete.

4.9. Scope and Failure Modes

The design has clear limits. It uses information in the encoded prefix and does not yet model rich case payloads, documents, process constraints, or external context in a general way. It does not fine-tune on the target log here. It also relies on retrieved supports, so missing labels, weak coverage, or poorly aligned neighborhoods directly affect prediction.

The method is most appropriate when fast deployment, limited labeled data, or avoiding per-log retraining matters. A dedicated LSTM or transformer PPM model may still be preferable for a large target log with enough labels and training time. FM-v2 tests how far retrieval-augmented transfer can go under these constraints.

5. Assessment

The assessment evaluates a fixed pretrained FM-v2 on target logs using labeled target-log prefixes as support. The parameters are not adjusted for these logs; the support set defines local activity labels and timing anchors at prediction time. The goal is to study when no-retraining prediction is useful, not to replace strong log-specific models. The foundation model[6], experimental results[7], and snapshot[8] are public.

FM-v2 is pretrained on synthetic event logs generated before target-log assessment. The generator samples process structures with varied control-flow patterns, plays them out into valid traces, assigns stochastic timing to simulated activities, and produces timestamped cases. Empty or invalid simulations are discarded, and remaining traces are exported as XES logs for pretraining. The public evaluation logs below are not used in pretraining. Figure 3 summarizes the process.

[6] https://github.com/fit-alessandro-berti/events-transf
[7] https://github.com/fit-alessandro-berti/experiments-fm
[8] https://drive.google.com/file/d/1S34YJXsP0ZhRFaqicdlKwp0CckMvM-1K/view?usp=sharing

 doi:10.20944/preprints202607.0705.v1

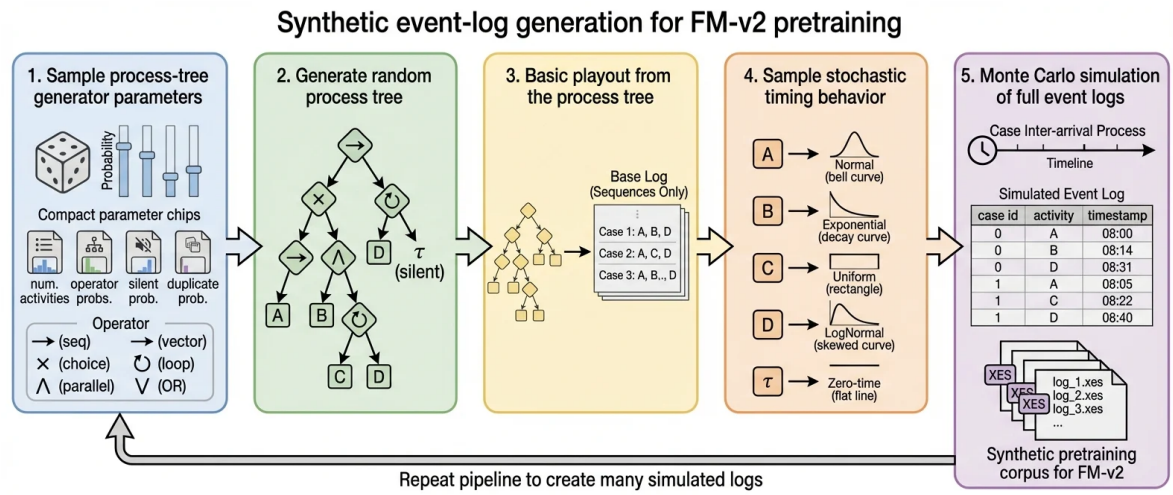


Figure 3. High-level pipeline used to generate the synthetic event logs for FM-v2 pretraining. For each run, generator parameters are sampled to create a random process tree. A base log is produced by playout and stochastic timing models are assigned to transitions. Monte Carlo simulation then generates timestamped traces, which are exported as synthetic XES event logs and aggregated into the pretraining corpus.

TWOThe synthetic pretraining corpus consists of the successful XES files in `logs/out`, generated by `perform_simulation.py`. Each run samples a synthetic process tree, plays out cases, assigns stochastic timing, and stores the resulting log. The exact generator settings are kept in a footnote.[9]

TWOThe snapshot contains 971 successful synthetic logs with 48,550 cases and 1,684,821 events. Trace lengths and timing values are diverse, so pretraining includes many short traces and some long traces. These statistics describe pretraining only and are not used to tune the generator to the five public target logs.[10]

5.1. Experimental Setup

We describe the evaluation protocol by separating pre-assessment logs from public target logs used during assessment. The setup covers target-log splits, support-pool construction, local activity labels, and comparison with retrained baselines.

5.1.1. Event Logs

We evaluate on five public target logs often used in PPM: *CoSeLoG Receipt* [16], *Road Traffic Fine Management* (10,000-case subset) [17], *Sepsis* [18], *Italian Help Desk* [19], and *Hospital Billing* [20]. These logs are used only for target-log assessment and never to fit FM-v2 parameters in these experiments.

For each target log, activity names are mapped only within that log. No global activity dictionary is shared. Retrieved support prefixes carry the local next-activity labels available to the FM-v2 head, so the target-log support set induces the next-activity output space.

We restrict all logs to activity labels and timestamps for a fair comparison across datasets and models. Public logs vary in extra attributes, and some methods may benefit from them more than others, mixing method quality with payload richness. The restriction also matches the current FM-v2

[9] TWOGenerator detail: each run samples a PTAndLogGenerator process tree. The minimum activity parameter is triangular on $[6, 14]$ with mode 9; the maximum is triangular on $[min + 5, min + 25]$ with mode $min + 12$. Operator probabilities for sequence, choice, parallel, loop, and or follow Dirichlet$(3, 3, 2, 2, 0.5)$. Silent and duplicate probabilities follow $Beta(2, 8)$ and $Beta(1, 15)$. Per-transition timing is normal with probability 0.40, exponential 0.30, uniform 0.20, and lognormal 0.10. Ranges are $\mu \in [1, 12]$ and $\sigma \in [0.2, 3.5]$ for normal, $loc \in [0, 2]$ and $scale \in [0.5, 6]$ for exponential, $loc \in [0, 3]$ and $scale \in [0.5, 8]$ for uniform, and $s \in [0.3, 1.2]$, $loc \in [0, 1.5]$, and $scale \in [0.5, 6]$ for lognormal. Transition weights follow $Gamma(2, 1)$.

[10] TWOCorpus detail: each synthetic log has 50 cases. The overall activity vocabulary has 33 generated strings. Per log, distinct activities range from 6 to 31, with median 15 and mean 15.36. Trace length has median 10, mean 34.70, interquartile range 4 to 26, 95th percentile 150, and maximum 3,616. Median trace duration is 0.0122 hours, about 44 seconds, with interquartile range 0.00369 to 0.0364 hours. Median inter-event gap is 0.00118 hours, about 4.25 seconds, with interquartile range 0.000654 to 0.00203 hours.

 doi:10.20944/preprints202607.0705.v1

input design, based on control-flow and time. It remains a limitation, since case and event attributes are often useful.

Experimental Datasets and Low-Data Regimes.

For each public target log, complete cases are split case-wise into 80% available cases and 20% held-out test cases. Held-out cases are never used for support construction or baseline training. For low-data settings, we sample training subsets from available cases without replacement for fractions $\rho \in (0, 1]$. Classical baselines train a log-specific model on selected cases. FM-v2 receives no gradient updates; selected cases only provide labeled prefixes for retrieval. All methods are evaluated on the full held-out test set. Thus, pretraining and target logs remain separate, and the within-log split controls only the amount of labeled target-log support.

5.1.2. ML Models

We compare FM-v2 with TabPFN and CPU-retrainable baselines across many log fractions. Because the benchmark spans several logs and fractions, we choose methods suited to repeated evaluation. TabPFN [13,21] is included as a pretrained in-context tabular baseline.

The CPU baselines are common in PPM and remaining-time studies and toolkits [7–9,22]: Random Forest for classification and regression, k-Nearest Neighbors, Support Vector Machines and Support Vector Regression, Gaussian Naive Bayes for classification, Linear Regression for remaining-time prediction, XGBoost [23] and LightGBM [24].

We do not include LSTM or transformer PPM models in this repeated benchmark. They are important task-specific baselines, but usually require training or fine-tuning a separate neural model for each target log and fraction. This differs from the fixed-parameter, support-conditioned setting studied here. The comparison is CPU-friendly and no-retraining oriented, not a claim of superiority over all sequence models.

5.1.3. Retrieval-Augmented Evaluation Protocol

FM-v2 is evaluated on held-out test cases with the same retrieval setup used at deployment. For each fraction ρ, every task-specific prefix from selected available cases enters the support pool, and every task-specific prefix from held-out cases becomes a query. Each query retrieves the top-k nearest support prefixes using normalized FM embeddings. Since support and test cases are disjoint, retrieval cannot return a prefix from the query case.

The training fraction changes support-pool coverage because larger fractions add prefixes from more target-log cases. Top-k controls the local context seen by the head. If the correct next activity is absent from retrieved supports, the classifier cannot select it because it scores only retrieved labels; these cases count as errors. For remaining time, retrieved prefixes act as local time anchors, so poor retrieval or weak coverage can increase MAE.

TWOWe evaluate several support-pool fractions and retrieval sizes. Overview tables show selected rows, while the script records the full sweep.[11]

We report two FM-based predictors. `proto_head` uses the retrieved support set in the FM head: for classification it predicts among retrieved labels, and for regression it uses retrieved target values. `foundation_knn` predicts directly on retrieved FM embeddings, with similarity-weighted voting for classification and similarity-weighted averaging for regression. This separates the learned support-conditioned head from a non-parametric use of the same representation and support pool. Figure 4 contrasts these options.

This is a partial ablation, not a full component study. Both FM predictors use the same pretrained encoder and target-log support pool. The comparison isolates the prediction rule, but not pretraining, the retrieval-oriented objective, or the expert mechanism.

[11] TWOThe support-pool fractions are 0.5%, 1%, 3%, 5%, 10%, 20%, 50%, and 100% of available target-log cases. The evaluated FM-v2 retrieval values are $k \in \{1, 5, 10, 20, 50, 100, 200\}$.

 doi:10.20944/preprints202607.0705.v1

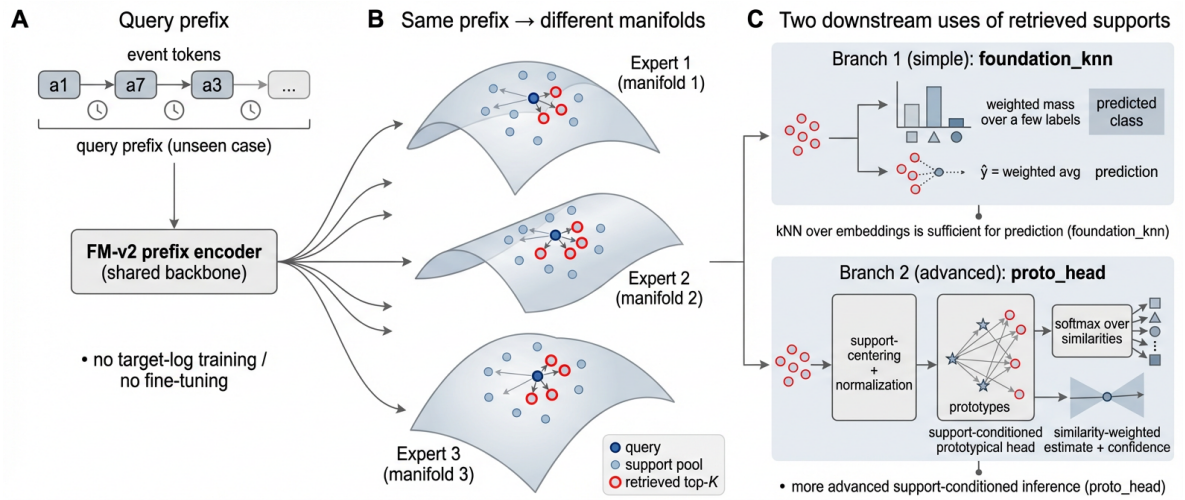


Figure 4. FM-v2 on an unseen log. A query prefix is mapped into expert-specific embedding spaces. In each space, `foundation_knn` predicts from nearest neighbors in the support pool. `proto_head` uses the retrieved support set for the final prediction.

5.1.4. Confidence Evaluation

For RQ3, we use the confidence value returned with each FM-v2 prediction. In next-activity prediction, confidence is high when most retrieved support weight favors the selected activity. In remaining-time prediction, it is high when supports are close to the query and agree on remaining time. These scores are used only for stratified analysis and do not change predictions.

5.1.5. Prefix Feature Representations for Tabular Baselines

The tabular baselines need a fixed-size vector for each prefix, so we use the following representations.

- **FM-v2 Embeddings.** We encode each prefix with the fixed FM-v2 encoder, use the retrieval embedding as its feature vector, and normalize it before neighbor-based analysis and prediction.
- **Handcrafted Prefix Features.** We also build a manual prefix feature vector. It records activities, directly-follows relations, and last-event time information: elapsed time since case start and time since the previous event.

Unless stated otherwise, classical baselines use handcrafted prefix features. FM-v2 embeddings are used for FM+kNN and representation analyses.

5.2. Results

We now present the empirical assessment of FM-v2 structured around the four research questions.

5.2.1. RQ1: Competitiveness

RQ1 asks whether pretrained FM-v2 is competitive with the evaluated baselines on unseen logs across training fractions for next-activity and remaining-time prediction. The comparison focuses on no-retraining and on the included repeated-evaluation baselines, not all dedicated per-log sequence models.

Metrics. For next activity, we report accuracy on test prefixes with a next event. For remaining time, we report MAE and R^2 on all test prefixes. We also report the gap to the best evaluated method for each log and fraction; a smaller gap means closer performance to the top method in this benchmark.

Next-Activity Prediction. Figure 5 and Figure 6 show that FM+kNN reaches the highest accuracy on *billing* at every training fraction. FM itself is competitive on *helpdesk* and reaches the top result in some settings. On *receipt* and most *roadtraffic* settings, TabPFN performs best. On *sepsis*, the best method changes across fractions. Overall, TabPFN has the smallest average gap to the best result, while the FM variants remain competitive on selected logs.

 doi:10.20944/preprints202607.0705.v1

Overall Classification Accuracy

Event Log	%	random forest	knn	svm	gaussian nb	xgboost	lightgbm	tabpfn	proto head	foundation knn	fm gap	fm knn gap
billing	0.5%	0.876	0.780	0.863	0.544	0.881	0.868	*0.896*	0.895	0.910	1.65%	0.00%
	3%	0.895	0.809	0.868	0.526	0.898	0.875	0.905	*0.920*	0.935	1.60%	0.00%
	5%	0.902	0.822	0.875	0.430	0.907	0.881	0.910	*0.915*	0.930	1.61%	0.00%
	10%	0.903	0.831	0.878	0.527	0.912	0.883	0.902	*0.925*	0.940	1.60%	0.00%
	20%	0.909	0.836	0.878	0.526	0.915	0.886	0.903	*0.940*	0.950	1.05%	0.00%
	100%	0.914	0.851	0.883	0.525	0.924	0.157	0.904	*0.945*	0.950	0.53%	0.00%
helpdesk	0.5%	0.726	0.599	0.741	0.642	0.736	*0.743*	0.760	0.725	0.685	4.60%	9.86%
	3%	0.772	0.698	0.784	0.470	0.766	0.743	0.811	*0.800*	0.740	1.36%	8.76%
	5%	0.767	0.696	0.786	0.353	0.768	0.744	*0.809*	0.810	0.740	0.00%	8.64%
	10%	0.764	0.710	0.790	0.349	0.768	0.735	*0.810*	0.810	0.760	0.00%	6.17%
	20%	0.762	0.720	0.797	0.192	0.791	0.743	0.816	*0.805*	0.745	1.31%	8.67%
	100%	0.777	0.744	0.806	0.348	0.809	0.715	*0.820*	0.840	0.780	0.00%	7.14%
receipt	0.5%	0.813	0.305	0.753	0.286	*0.814*	0.335	0.816	0.680	0.650	16.63%	20.31%
	3%	*0.811*	0.439	0.808	0.256	0.800	0.776	0.841	0.800	0.770	4.89%	8.46%
	5%	0.797	0.448	*0.807*	0.241	0.771	0.745	0.827	0.785	0.775	5.09%	6.30%
	10%	0.799	0.438	0.806	0.095	0.782	0.759	0.838	0.805	*0.810*	3.93%	3.34%
	20%	0.788	0.467	0.798	0.027	0.774	0.762	0.836	0.805	*0.810*	3.72%	3.13%
	100%	0.815	0.505	0.815	0.071	0.817	0.805	0.860	*0.850*	0.830	1.16%	3.48%
roadtraffic_10000	0.5%	*0.875*	0.818	0.819	0.755	0.842	0.843	0.908	0.850	0.835	6.44%	8.09%
	3%	*0.932*	0.892	0.838	0.772	0.917	0.914	0.936	0.855	0.850	8.66%	9.20%
	5%	*0.932*	0.892	0.842	0.768	0.926	0.925	0.936	0.880	0.870	5.94%	7.00%
	10%	*0.937*	0.904	0.844	0.775	0.934	0.928	0.942	0.885	0.885	6.05%	6.05%
	20%	*0.936*	0.908	0.915	0.773	0.933	0.929	0.945	0.895	0.880	5.26%	6.85%
	100%	0.947	0.917	0.930	0.772	*0.947*	0.945	0.945	0.900	0.885	4.95%	6.54%
sepsis	0.5%	0.471	0.339	0.408	0.228	0.554	0.416	0.491	0.446	*0.495*	19.50%	10.62%
	3%	0.579	0.390	0.524	0.255	0.596	0.573	0.609	0.542	*0.605*	10.99%	0.68%
	5%	0.595	0.396	0.531	0.263	*0.613*	0.608	0.607	0.550	0.630	12.76%	0.00%
	10%	0.620	0.409	0.543	0.208	*0.643*	0.637	0.659	0.558	0.640	15.32%	2.89%
	20%	0.623	0.424	0.580	0.228	*0.649*	0.646	0.668	0.572	0.640	14.51%	4.26%
	100%	0.643	0.447	0.612	0.245	0.672	*0.682*	0.701	0.598	0.635	14.66%	9.37%

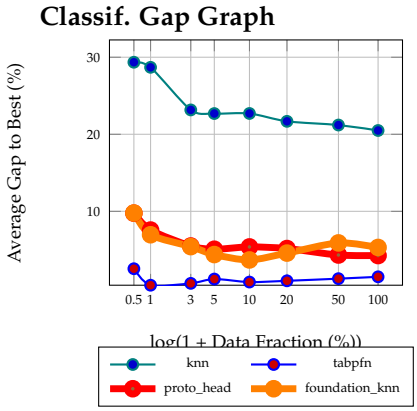


Figure 5. Next-activity prediction: accuracy by log and data fraction (left; values in $[0, 1]$) and mean gap-to-best across logs for selected methods (right; %).

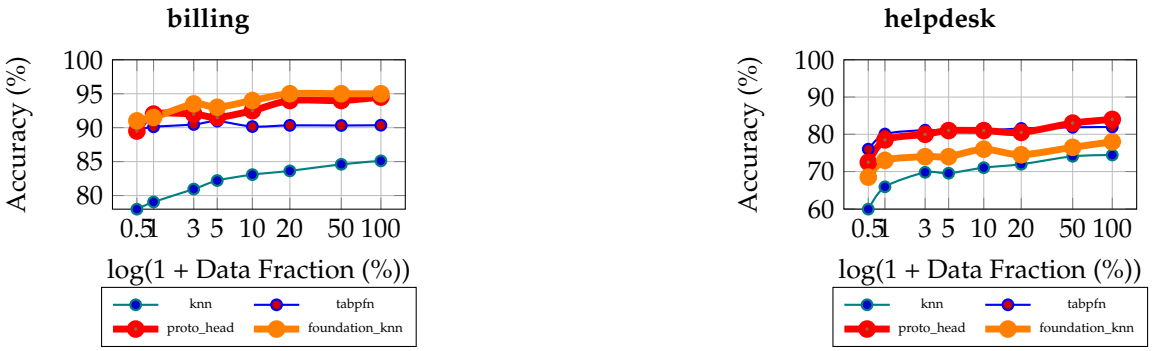


Figure 6. Next-activity accuracy vs. data fraction on *billing* and *helpdesk* (selected methods).

Remaining-Time Prediction. Figure 7 and Figure 8 show that FM+kNN gives the lowest MAE on *billing* across all fractions and often also on *sepsis*. On *receipt*, SVR performs best. On *helpdesk* and most *roadtraffic* settings, TabPFN performs best. The kNN probe on FM embeddings often performs better than the FM prediction head itself, which indicates that the learned representation is useful even when the support-conditioned head is not the top method.

Overall Regression MAE

Event Log	%	linear regression	random forest	knn	svr	xgboost	lightgbm	tabpfn	proto head	foundation knn	fm gap	fm knn gap
billing	0.5%	1121.20	1134.17	1097.94	1163.34	1212.59	1167.42	*1027.95*	1160.78	862.24	34.62%	0.00%
	3%	1095.36	1083.62	1091.73	1163.30	1131.83	1068.50	*990.78*	1194.60	840.10	42.20%	0.00%
	5%	1140.28	1146.92	1163.43	1163.22	1178.85	1099.42	*1023.93*	1163.64	850.30	36.85%	0.00%
	10%	1146.56	1125.97	1132.94	1163.16	1122.39	1070.46	*1022.91*	1120.54	843.26	32.88%	0.00%
	20%	1123.33	1088.50	1129.99	1163.08	1061.39	1033.79	*1001.91*	1157.26	836.87	38.28%	0.00%
	100%	1125.56	1066.67	1138.90	1162.50	1028.49	1029.14	*1011.74*	1099.84	762.32	44.27%	0.00%
helpdesk	0.5%	0.20	*0.13*	0.15	0.37	0.14	0.16	0.13	0.20	0.18	53.51%	43.48%
	3%	0.16	*0.12*	0.14	0.37	0.13	0.13	0.12	0.18	0.18	52.81%	55.57%
	5%	0.16	*0.13*	0.14	0.36	0.14	0.13	0.12	0.18	0.17	60.47%	49.27%
	10%	0.16	*0.13*	0.14	0.35	0.14	0.14	0.12	0.17	0.17	48.76%	45.34%
	20%	0.16	*0.12*	0.13	0.34	0.13	0.13	0.11	0.17	0.17	47.65%	47.30%
	100%	0.16	0.12	0.13	0.26	0.12	*0.12*	0.11	0.17	0.15	46.11%	35.12%
receipt	0.5%	330.62	256.33	198.69	63.65	204.65	252.70	*83.63*	97.11	96.47	52.55%	51.55%
	3%	94.09	97.24	87.44	63.65	94.11	105.09	*70.83*	101.72	100.11	59.81%	57.27%
	5%	135.06	127.94	124.27	63.66	139.18	176.75	*77.19*	99.91	94.89	56.93%	49.05%
	10%	86.62	83.95	83.72	63.65	90.36	97.97	*67.51*	100.90	98.42	58.52%	54.62%
	20%	94.30	98.86	105.27	63.65	107.69	125.53	*68.75*	99.08	94.55	55.66%	48.54%
	100%	95.58	88.34	101.20	63.65	99.51	98.19	*75.13*	94.47	91.25	48.43%	43.37%
roadtraffic_10000	0.5%	5006.76	4517.20	4674.79	6630.91	5000.03	4518.14	4321.48	*4459.56*	4835.69	3.20%	11.90%
	3%	4845.06	4413.49	*4271.01*	6696.12	4700.20	4533.20	4049.38	4477.57	4712.18	10.57%	16.37%
	5%	4784.91	4371.44	4272.63	6630.47	4589.16	4409.42	3963.41	4440.82	4468.74	12.05%	12.75%
	10%	4757.27	4280.74	*4184.32*	6636.04	4560.25	4289.11	3930.25	4456.00	4597.30	13.38%	16.97%
	20%	4798.20	4257.33	4178.20	6610.36	4479.46	*4176.10*	3904.16	4402.37	4556.78	12.76%	16.72%
	100%	4747.07	4091.18	4061.78	6613.61	4072.16	3909.85	*3916.86*	4287.36	4539.75	9.66%	16.11%
sepsis	0.5%	1078.99	855.25	1090.46	824.30	859.35	884.89	846.16	703.20	*779.26*	0.00%	10.82%
	3%	1618.10	1059.61	1086.16	816.71	1020.22	1727.84	978.11	*788.30*	777.50	1.39%	0.00%
	5%	1590.01	1201.28	1125.42	820.67	1302.71	1456.98	1139.81	718.78	*760.54*	0.00%	5.81%
	10%	1231.83	1164.17	1003.45	817.32	1169.97	1157.05	1087.16	*727.68*	723.43	0.59%	0.00%
	20%	1105.65	994.31	980.66	817.08	1098.77	1105.59	930.18	713.35	*736.68*	0.00%	3.27%
	100%	1071.65	1078.93	1020.32	816.21	1104.69	1057.03	965.57	614.94	*732.87*	0.00%	19.18%

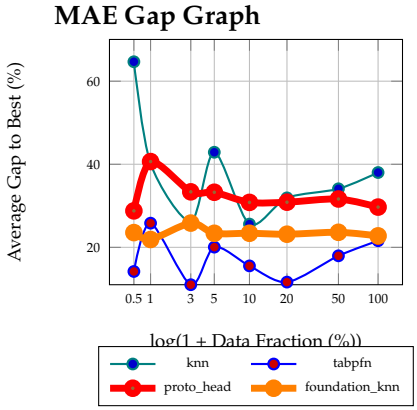
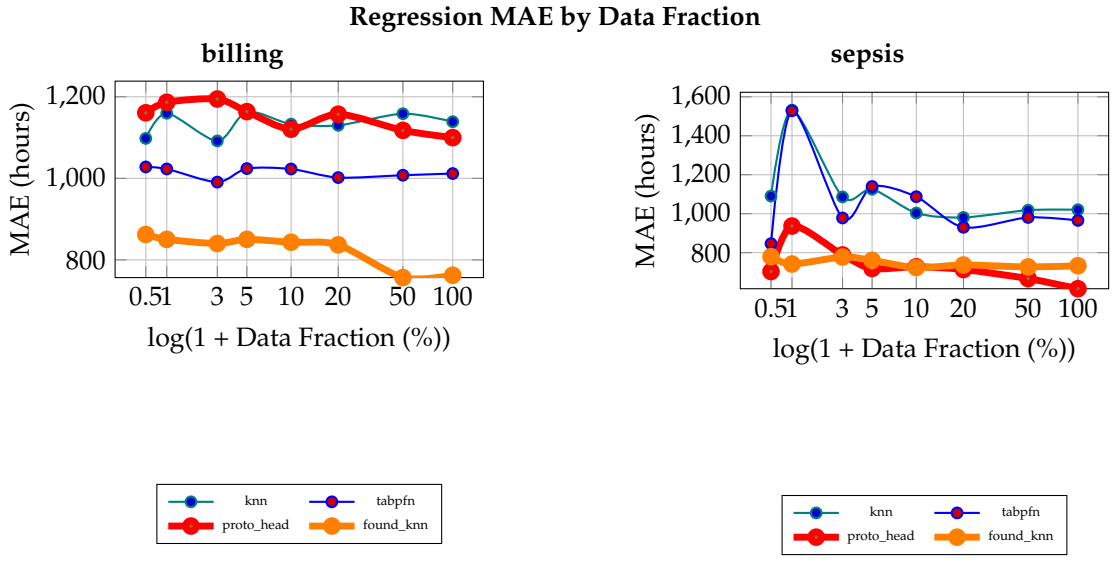


Figure 7. Remaining-time prediction: MAE by log and data fraction (left; hours) and mean gap-to-best across logs for selected methods (right; %).

 doi:10.20944/preprints202607.0705.v1



Regression R^2

Event Log	%	linear regression	random forest	knn	svr	xgboost	lightgbm	tabpfn	proto head	foundation knn
billing	0.5%	0.139	0.054	0.125	-0.216	-0.034	0.063	*0.148*	**0.160**	0.075
	3%	0.153	0.102	0.113	-0.216	0.048	0.135	**0.177**	*0.173*	0.102
	5%	0.158	0.053	0.083	-0.211	0.019	0.165	*0.192*	**0.254**	0.105
	10%	0.120	0.041	0.074	-0.212	0.062	*0.145*	0.141	**0.156**	0.125
	20%	0.158	0.109	0.093	-0.212	0.167	**0.209**	0.150	*0.178*	0.164
	100%	0.164	0.135	0.102	-0.213	0.217	*0.220*	0.165	0.209	**0.304**
helpdesk	0.5%	0.654	*0.804*	0.795	-0.079	0.767	0.750	**0.825**	0.105	0.314
	3%	0.806	0.819	*0.822*	-0.059	0.786	0.813	**0.846**	0.341	0.343
	5%	0.798	0.813	*0.824*	-0.007	0.764	0.809	**0.852**	0.274	0.381
	10%	0.805	0.813	*0.822*	0.046	0.778	0.804	**0.851**	0.347	0.397
	20%	0.809	0.823	*0.829*	0.120	0.807	0.826	**0.857**	0.384	0.404
	100%	0.812	0.833	0.831	0.472	0.835	*0.848*	**0.855**	0.432	0.511
receipt	0.5%	-14.943	-8.620	-3.199	-0.156	-10.067	-3.122	**0.023**	*-0.018*	-0.036
	3%	-0.087	-0.263	-0.028	-0.156	-0.354	-0.202	-0.031	**0.243**	*-0.010*
	5%	-0.447	-1.911	-0.760	-0.156	-4.082	-3.192	0.021	*0.026*	**0.039**
	10%	*-0.009*	-0.024	-0.064	-0.156	-0.156	-0.092	-0.029	-0.018	**-0.002**
	20%	0.022	-0.599	-0.389	-0.156	-0.913	-0.862	*0.047*	0.018	**0.054**
	100%	0.000	-0.158	-0.316	-0.156	-0.314	-0.171	*0.096*	**0.160**	0.046
roadtraffic_10000	0.5%	0.236	0.265	0.263	-0.006	0.070	*0.285*	**0.334**	0.015	0.055
	3%	0.254	0.205	*0.303*	-0.241	0.129	0.222	**0.343**	0.006	0.075
	5%	0.281	0.222	*0.294*	-0.006	0.177	0.267	**0.379**	-0.009	0.145
	10%	0.284	0.264	*0.311*	-0.097	0.195	0.307	**0.376**	0.029	0.138
	20%	0.276	0.268	0.303	-0.001	0.215	*0.326*	**0.378**	0.026	0.140
	100%	0.284	0.312	0.318	-0.030	0.328	**0.387**	*0.373*	0.123	0.153
sepsis	0.5%	-0.451	-0.236	*-0.057*	-0.202	-0.239	-0.239	-0.223	-0.110	**-0.011**
	3%	-1.076	-0.285	-0.095	-0.177	-0.404	-1.275	-0.244	*-0.073*	**-0.028**
	5%	-0.969	-0.699	-0.123	-0.193	-0.868	-0.751	-0.434	*-0.047*	**-0.024**
	10%	-0.348	-0.432	-0.071	-0.180	-0.451	-0.284	-0.285	**0.040**	*-0.030*
	20%	-0.233	-0.177	-0.076	-0.179	-0.322	-0.245	-0.087	**0.043**	*-0.041*
	100%	-0.090	-0.203	-0.068	-0.174	-0.248	-0.139	-0.067	**0.261**	*-0.033*

Figure 8. Remaining-time prediction: MAE vs. data fraction for *billing* and *sepsis* (top, hours) and R^2 across logs and fractions (bottom).

Answer to RQ1. In this benchmark, pretrained FM-v2 is competitive on some logs and tasks without target-log training, especially with kNN over its embeddings. It is strongest on *billing* for both tasks and on *sepsis* regression. Yet performance is log-dependent, and TabPFN or classical baselines are better elsewhere. RQ1 therefore supports conditional competitiveness in the no-retraining setting, not broad dominance.

Comparison with Published High-Performance Baselines

To address RQ1 baseline scope, Table 1 compares 100% support FM-v2 results with a compact set of strong results from other PPM papers on the same logs. It does not replace the repeated evaluation in Figure 5, Figure 7, and Figure 8. It is an external reference point for dedicated sequence, graph, transformer, mixture-of-experts, and retrieval-based methods.

 doi:10.20944/preprints202607.0705.v1

Table 1. Selected comparison with high-performing values from original PPM papers on the same logs. FM-v2 values come from the 100% support setting in Figure 5, Figure 7, and Figure 8. The table reports the better of `proto_head` and `foundation_knn` for each FM-v2 metric. External MAE values reported in days are converted to hours. n.r. means no R^2 was reported in the selected same-log sources.

Log	FM acc.	Reported acc.	FM MAE (h)	Reported MAE (h)	FM R^2	Rep. R^2
CoSeLoG Receipt	0.850	OREO Transformer: 0.919 [25]	91.25	PGTNet: 65.28 [26]	0.160	n.r.
Helpdesk	0.840	MMoTE: 0.91 [27]	0.15	MMoTE: 89.76 [27]	0.511	n.r.
Sepsis	0.635	RAG with phi-4: 0.70 [28]	614.94	PGTNet: 395.52 [26]	0.261	n.r.
Road Traffic Fine	0.900	ProcessTransformer: 0.900 [3]	4287.36	PGTNet: 2724.72 [26]	0.153	n.r.
Hospital Billing	0.950	GAT single: 0.933 [29]	762.32	XGB-RF: 738.31 [30]; PGTNet: 856.32 [26]	0.304	n.r.

External values come from the original papers and use different feature extraction and preprocessing. Several use richer original or derived attributes, while this FM-v2 study mainly uses control-flow and timestamp information. The numbers are therefore not directly comparable, but they indicate where FM-v2 stands relative to strong log-specific methods.

The comparison is mixed but useful. For next activity, FM-v2 is above the selected graph neural result on *Hospital Billing*, close to ProcessTransformer on *Road Traffic Fine Management*, and below the best selected values on *CoSeLoG Receipt*, *Helpdesk*, and *Sepsis*. For remaining time, FM-v2 is close to the selected ensemble on *Hospital Billing* but behind the strongest selected methods on *CoSeLoG Receipt*, *Sepsis*, and *Road Traffic Fine Management*. Helpdesk MAE needs care because preprocessing and target definitions differ across papers.

Overall, Table 1 supports a cautious RQ1 answer. FM-v2 can reach the range of strong methods on some logs without target-log training, especially on *Hospital Billing*. However, specialized log-specific models remain stronger on several tasks. FM-v2 is therefore a competitive no-retraining alternative in selected settings, not a universal replacement for models engineered for one log.

Comparison with FM-v1

Table 2 compares FM-v2 with FM-v1 [4] on the same five target logs. The FM-v1 columns keep the 1, 5, 10, and 20 shot settings. The gain column compares FM-v2 with the best FM-v1 value for each metric, making the comparison conservative for FM-v1.

Table 2. Comparison between FM-v1 and FM-v2 on the five target logs. FM-v1 uses the reported 1, 5, 10, and 20 shot settings. FM-v2 uses the 100% support results from this assessment, taking the better FM-v2 predictor for each metric. MAE is in hours. In the MAE gain column, the percentage is the reduction relative to the best FM-v1 MAE; a negative value means FM-v2 has larger MAE.

Log	FM-v1 acc. 1/5/10/20	FM-v2 acc.	FM-v1 MAE (h) 1/5/10/20	FM-v2 MAE (h)	FM-v1 R^2 1/5/10/20	FM-v2 R^2	Gain vs. best FM-v1
	0.329/0.377		181.11/85.15		-0.174/-0.102		Acc. +0.422 MAE −7.2%
CoSeLoG Receipt	0.393/0.428	**0.850**	119.32/86.79	91.25	-0.058/-0.070	**0.160**	R^2 +0.218
	0.318/0.389		0.33/0.27		-1.101/-0.107		Acc. +0.390 MAE 30.1%
Helpdesk	0.423/0.450	**0.840**	0.22/0.21	**0.15**	0.055/0.161	**0.511**	R^2 +0.350
	0.373/0.450		1291.98/990.24		-1.512/-0.134		Acc. +0.152 MAE 28.3%
Sepsis	0.464/0.483	**0.635**	857.70/900.66	**614.94**	-0.085/-0.118	**0.261**	R^2 +0.346
	0.616/0.671		7437.31/5772.81		-1.130/-0.432		Acc. +0.211 MAE 13.1%
Road Traffic Fine	0.664/0.689	**0.900**	4933.58/5237.55	**4287.36**	-0.114/0.006	**0.153**	R^2 +0.147
	0.420/0.506		2183.09/1346.33		-0.471/-0.166		Acc. +0.411 MAE 30.0%
Hospital Billing	0.535/0.539	**0.950**	1515.60/1089.48	**762.32**	-0.060/0.033	**0.304**	R^2 +0.271

 doi:10.20944/preprints202607.0705.v1

The results show clear gains from FM-v1 to FM-v2. Next-activity accuracy improves on every log, with absolute gains of 0.152 to 0.422 over the best FM-v1 shot setting. R^2 also improves on every log. For remaining time, FM-v2 reduces MAE on four logs; the exception is *CoSeLoG Receipt*, where the best FM-v1 MAE is slightly lower.

These gains are linked to FM-v2's architectural changes. FM-v2 makes prediction-time retrieval explicit, aligns training with retrieval-based inference, conditions predictions on a top-k target-log support set, and also supports direct nearest-neighbor prediction on learned representations. This strengthens adaptation to a new log without gradient updates. The comparison informs RQ1, but is not a full ablation because the two generations use different evaluation designs.

5.2.2. RQ2: Predictors of Performance

RQ2 studies which event-log properties can indicate in advance whether FM-v2 is likely to perform well on an unseen log.

We focus on the relation between activity patterns and time information. In particular, we examine whether logs with a stronger connection between control-flow and temporal behavior are also the logs where FM-v2 performs better relative to the baselines.

Log-Level Activity and Time Metrics. To study this question, we compute simple dependence measures on the handcrafted prefix features described in subsubsection 5.1.5. These measures capture how strongly activity-related information and time-related information move together within a log. A higher value means that time carries more information about the activity structure, or that activity patterns carry more information about timing.

Results. Table 3 shows negative correlations between activity-time dependence and average gap-to-best for both tasks. The relation is stronger for remaining time than for next activity. Because only five target logs are used, the values are exploratory rather than definitive.

Table 3. Log-level Pearson correlation ($n = 5$ logs) between activity and time dependence metrics and mean gap-to-best for next-activity accuracy and remaining-time MAE (lower gap is better).

Classification Gap to Best (lower is better)

Metric	Pearson r	Logs used
Distance correlation(activity_block, time_block)	-0.750	5
MI(activity_indicators, time_features) mean	-0.463	5
MI(activity_indicators, time_features) prevalence-weighted mean	-0.759	5

MAE Gap to Best (lower is better)

Metric	Pearson r	Logs used
Distance correlation(activity_block, time_block)	-0.968	5
MI(activity_indicators, time_features) mean	-0.930	5
MI(activity_indicators, time_features) prevalence-weighted mean	-0.881	5

In this benchmark, logs with stronger activity-time dependence tend to be those where FM-v2 is closer to the best method, sometimes reaching it. Because the analysis is log-level and uses few public target logs, this is an explanatory pattern, not a stand-alone selection rule.

Answer to RQ2. Activity-time dependence is a candidate indicator of when FM-v2 may transfer well. Here, higher dependence is associated with stronger FM-v2 results, especially for remaining time. When dependence is low, classical tabular baselines are often safer. More logs are needed before treating this as a reliable rule.

5.2.3. RQ3: Confidence and Performance

RQ3 studies whether the confidence scores produced by FM-v2 can separate stronger from weaker predictions on unseen logs, and how this compares to a random-forest baseline.

Bucket-wise results are shown in Figure 9 for classification and Figure 10 for regression.

 doi:10.20944/preprints202607.0705.v1

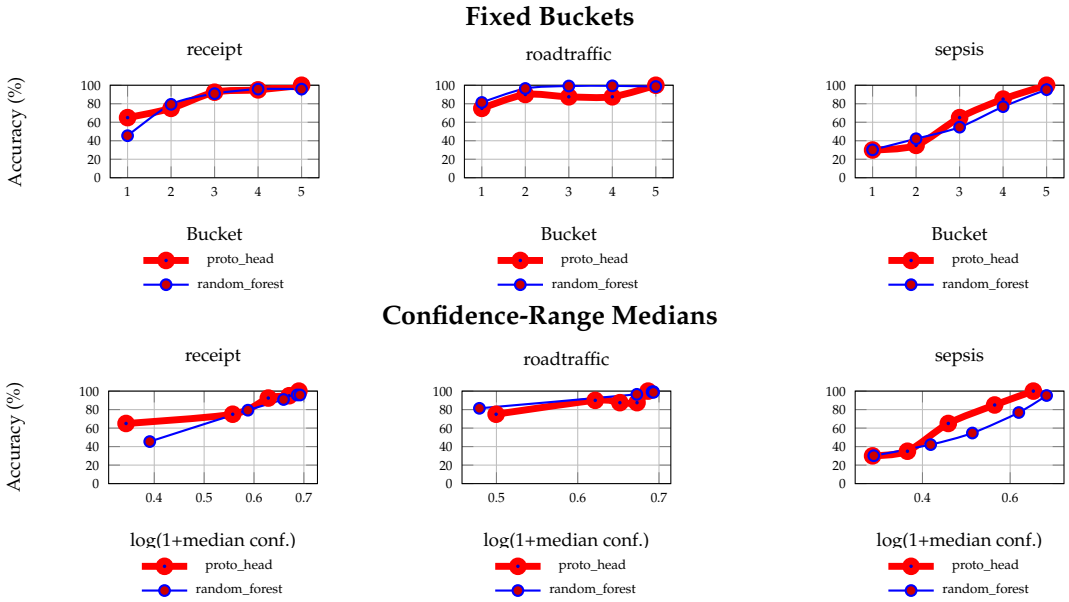


Figure 9. Next-activity accuracy stratified by confidence for FM-v2 (`proto_head`) and Random Forest on *receipt*, *roadtraffic*, and *sepsis*. Top: fixed confidence bins. Bottom: buckets positioned by median confidence.

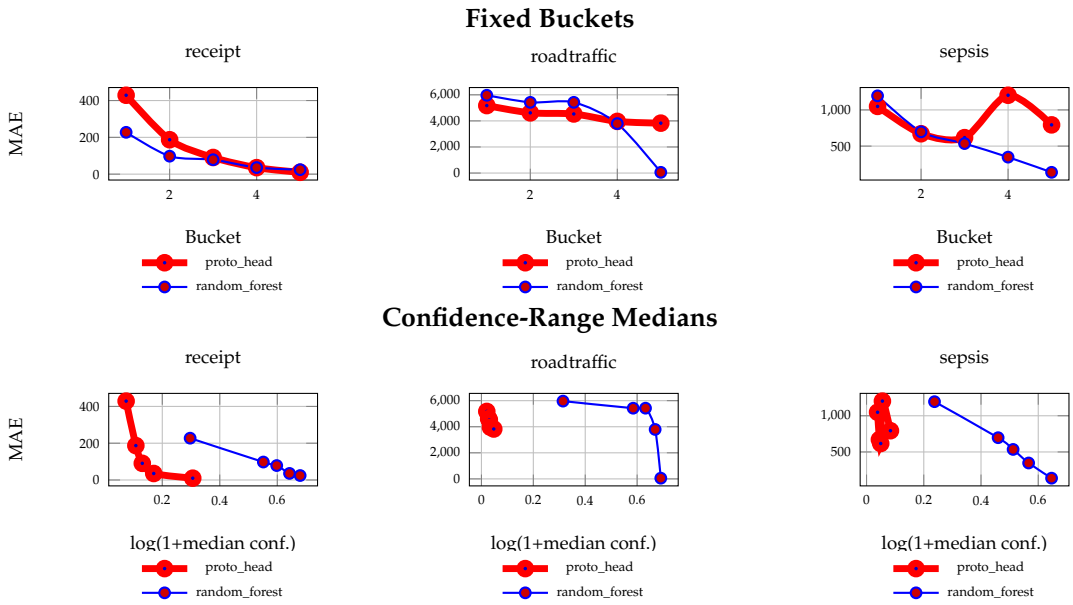


Figure 10. Remaining-time MAE stratified by confidence for FM-v2 (`proto_head`) and Random Forest on *receipt*, *roadtraffic*, and *sepsis*. Top: fixed confidence bins. Bottom: buckets positioned by median confidence (lower is better).

Classification. For next-activity prediction, FM-v2 usually becomes more accurate as confidence increases. This trend is clear on *receipt* and *sepsis*. On *roadtraffic*, the highest-confidence bucket still gives the best accuracy, but the middle buckets are less ordered. The random-forest baseline shows a similar pattern.

Regression. For remaining-time prediction, FM-v2 shows lower MAE at higher confidence on *receipt* and *roadtraffic*. On *sepsis*, the pattern is less stable, since a middle-confidence bucket performs worse than nearby buckets. The random-forest baseline is more ordered on this event log.

Answer to RQ3. FM-v2 confidence helps separate stronger from weaker predictions in next-activity prediction and in remaining-time prediction on *receipt* and *roadtraffic*. The link is weaker for *sepsis* regression. Confidence is useful for stratification, but not a universal calibration guarantee; it should be checked per log and task.

 doi:10.20944/preprints202607.0705.v1

5.2.4. RQ4: Expert Representations

RQ4 asks whether FM-v2 expert representations support retrieval-based prediction and capture different structure. This is a representation analysis, not a full expert-mechanism ablation.

Representation Metrics. We analyze each expert separately by using its output vectors as prefix representations. We then test whether nearby prefixes tend to share the same label, whether these representations support nearest-neighbor prediction, and how similar the expert spaces are to each other.

Within-Expert Structure. Within each expert space, prefixes with the same label tend to group together. This holds in local neighborhoods and in the broader layout of the space, and it is not limited to only the closest neighbors.

Between-Expert Diversity. The experts do not learn the same feature space. Their outputs differ for the same prefix, which means that they organize the data differently. As a result, different experts can retrieve different supports and capture different relations between prefixes.

Answer to RQ4. Individual experts support retrieval and nearest-neighbor prediction without hand-crafted features, while also capturing different structure. This supports aggregating expert outputs in FM-v2, but does not quantify the expert mechanism's contribution to end-to-end accuracy.

6. Discussion

In its current form, FM-v2 is most convincing as a transferable representation learner and no-retraining predictor, not as a complete PPM model for all settings. Across both tasks, `foundation_knn` often matches or exceeds `proto_head`, suggesting that much transferable signal already lies in the learned embedding space. This supports the pretraining strategy but also exposes a weakness: the support-conditioned head does not consistently exploit retrieved context better than simple non-parametric prediction. The results show that reusable event-log representations can support fast adaptation, not that FM-v2 already dominates well-tuned per-log LSTM, transformer, or other neural PPM models.

Retrieval is both FM-v2's strength and bottleneck. It enables adaptation to unseen logs without a fixed global label space or target-log retraining, but makes prediction sensitive to support coverage, top-k, and neighborhood quality. For next activity, the correct label is unreachable if absent from retrieved support. For remaining time, diverse target values in a neighborhood can destabilize the weighted average. Since more support data does not always close the gap to strong baselines, the limitation is not only data amount, but also how well the representation aligns with the target log and retrieves relevant local behavior.

The results support a qualified transfer claim. FM-v2 is competitive on some logs and tasks, but not uniformly strong; TabPFN or classical tabular baselines remain better in several settings. The assessment separates pretraining from real target logs, and the within-target split forms support or baseline-training cases and held-out query cases, matching support-based adaptation. Evidence is still limited by the small number of real logs, the absence of dedicated per-log neural sequence baselines, and the FM-v1 comparison, which uses provided FM-v1 result files rather than a fully matched rerun. FM-v2 should be viewed as a reusable cross-log predictor with promising no-retraining behavior, not a replacement for well-tuned log-specific models. The activity-time dependence analysis is useful, but with five evaluation logs (Table 3) it remains exploratory rather than a robust selection rule.

The confidence and expert analyses point the same way. Confidence often ranks predictions well, especially for classification, but it is not a calibrated estimate of correctness or error and is less stable for regression. Expert diversity suggests that the model avoids collapse to one representation space, yet fixed aggregation may blur cases where only some experts fit a log. The `proto_head` versus `foundation_knn` comparison is a useful partial check, but it does not isolate the expert mechanism, support-conditioned head, retrieval-oriented objective, or encoder. These ablations are still needed to explain the observed transfer behavior.

 doi:10.20944/preprints202607.0705.v1

Pretraining data also bounds the conclusions. Synthetic logs expose the model to many controlled process structures and timing patterns before real-log use, but they need not cover enterprise irregularities such as noisy recording, changing behavior, or domain-specific timing. Local activity IDs avoid a shared output vocabulary across logs, but shift difficulty to the target-log support pool. A new activity can be predicted only when represented in retrieved support, a practical trade-off of no-retraining design.

The main limitations are clear. The benchmark uses only five public logs, the input design relies mainly on activity labels and timestamps, payload attributes are not yet central, and comparisons are limited to CPU-friendly baselines plus TabPFN. This keeps the study tractable and aligned with FM-v2 inputs, but limits conclusions about deployment where richer attributes and stronger task-specific models may matter. Overall, the evidence supports retrieval-augmented foundation modeling for PPM, while showing that FM-v2 needs stronger retrieval robustness, better head design, calibrated confidence, broader benchmarks, a tighter comparison to earlier foundation-model variants, and cleaner component ablations.

7. Conclusion

FM-v2 applies retrieval-based in-context prediction to unseen event logs without target-log retraining. Across five logs, performance depends on log and task, and much of the gain comes from the learned representation because kNN on FM embeddings often matches or exceeds the support-conditioned head. Activity-time dependence helps identify when transfer may work, while confidence helps rank predictions but needs log-specific validation. The results make retrieval-augmented foundation models a promising option when fast PPM adaptation matters, but not a universal replacement for task-specific PPM models.

Future work should test FM-v2 on more real logs and under process drift, extend the FM-v1 comparison with matched support protocols, compare with stronger per-log sequence models, and study how support size, top-k, coverage, and missing retrieved labels affect both tasks. Component ablations should isolate the expert mechanism, support-conditioned head, retrieval-oriented objective, and pretraining benefit over random initialization. Further work should align retrieval and prediction more closely during training, add richer payload information, and calibrate confidence for routing or human review. This can clarify when retrieval-based foundation models should be used alone, with kNN on embeddings, or with tabular and neural baselines.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The public event logs are available from the sources cited in Section 5.1.1. Code, results, and a pretrained-model snapshot are available at the links provided in Section 5. **Code Availability**: The code used to implement the method and run the experiments is publicly available[12]

Conflicts of Interest: The authors declare that they have no competing interests.

Funding: Funded by the European Union. This work received funding from the European High Performance Computing Joint Undertaking (JU) and from the German Federal Ministry of Research, Technology and Space (BMFTR), the Ministry of Culture and Science of North Rhine-Westphalia (MKW NRW), and the Hessian Ministry of Science and Research, Arts and Culture (HMWK) under grant agreement No. 101250682.

Author Contributions: FAlessandro Berti implemented the method and the experiments and prepared the first manuscript draft. Wil M.P. van der Aalst contributed to the study design, interpretation of the results, and revision of the manuscript.

References

1. van der Aalst, W. *Process Mining - Data Science in Action, Second Edition*; Springer: New York, USA, 2016. https://doi.org/10.5555/2948762.

[12] https://github.com/fit-alessandro-berti/events-transf and https://github.com/fit-alessandro-berti/experiments-fm.

 doi:10.20944/preprints202607.0705.v1

2. Tax, N.; Verenich, I.; La Rosa, M.; Dumas, M. Predictive Business Process Monitoring with LSTM Neural Networks. In Proceedings of the CAiSE, New York, USA, 2017; Vol. 10253, *Lecture Notes in Computer Science*, pp. 477–492. https://doi.org/10.1007/978-3-319-59536-8_30.
3. Bukhsh, Z.; Saeed, A.; Dijkman, R. ProcessTransformer: Predictive Business Process Monitoring with Transformer Network. *CoRR* **2021**, *abs/2104.00721*. https://doi.org/10.48550/arXiv.2104.00721.
4. Berti, A.; van der Aalst, W. An In-Context Foundation Model for Predictive Process Monitoring on Event Logs. *IEEE Access* **2026**. https://doi.org/10.1109/ACCESS.2026.3658877.
5. Francescomarino, C.D.; Ghidini, C. Predictive Process Monitoring. In *Process Mining Handbook*; Lecture Notes in Business Information Processing, Springer, 2022; pp. 320–346. https://doi.org/10.1007/978-3-031-08848-3_10.
6. Teinemaa, I.; Dumas, M.; La Rosa, M.; Maggi, F. Outcome-Oriented Predictive Process Monitoring: Review and Benchmark. *ACM Trans. Knowl. Discov. Data* **2019**, *13*, 17:1–17:57. https://doi.org/10.1145/3301300.
7. Verenich, I.; Dumas, M.; La Rosa, M.; Maggi, F.; Teinemaa, I. Survey and Cross-benchmark Comparison of Remaining Time Prediction Methods in Business Process Monitoring. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 34:1–34:34. https://doi.org/10.1145/3331449.
8. Tama, B.; Comuzzi, M.; Ko, J. An Empirical Investigation of Different Classifiers, Encoding, and Ensemble Schemes for Next Event Prediction Using Business Process Event Logs. *ACM Trans. Intell. Syst. Technol.* **2020**, *11*, 68:1–68:34. https://doi.org/10.1145/3406541.
9. Rizzi, W.; Di Francescomarino, C.; Ghidini, C.; Maggi, F. Nirdizati: an advanced predictive process monitoring toolkit. *J. Intell. Inf. Syst.* **2025**, *63*, 259–291. https://doi.org/10.1007/S10844-024-00890-9.
10. Mehdiyev, N.; Majlatow, M.; Fettke, P. Interpretable and explainable machine learning methods for predictive process monitoring: a systematic literature review. *Artif. Intell. Rev.* **2025**, *58*, 378. https://doi.org/10.1007/s10462-025-11399-0.
11. Guo, N.; Liu, C.; Li, C.; Zeng, Q.; Ouyang, C.; Liu, Q.; Lu, X. Explainable and Effective Process Remaining Time Prediction Using Feature-Informed Cascade Prediction Model. *IEEE Trans. Serv. Comput.* **2024**, *17*, 949–962. https://doi.org/10.1109/TSC.2024.3353817.
12. Brown, T. Language Models are Few-Shot Learners. In Proceedings of the Advances in Neural Information Processing Systems, 2020, Vol. 33, pp. 1877–1901. https://doi.org/10.5555/3495724.3495883.
13. Hollmann, N.; Müller, S.; Purucker, L.; Krishnakumar, A. Accurate predictions on small data with a tabular foundation model. *Nature* **2025**, *637*, 319–326. https://doi.org/10.1038/s41586-024-08328-6.
14. Snell, J.; Swersky, K.; Zemel, R. Prototypical Networks for Few-shot Learning. In Proceedings of the Advances in Neural Information Processing Systems, 2017, Vol. 30. https://doi.org/10.5555/3294996.3295163.
15. Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; Dean, J. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer, 2017. https://doi.org/10.48550/arXiv.1701.06538.
16. Buijs, J. Receipt phase of an environmental permit application process (WABO), CoSeLoG project, 2014. https://doi.org/10.4121/uuid:a07386a5-7be3-4367-9535-70bc9e77dbe6.
17. de Leoni, M.; Mannhardt, F. Road traffic fine management process, 2015. https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5.
18. Mannhardt, F. Sepsis Cases - event log, 2016. https://doi.org/10.4121/uuid:915d2bfb-7e84-49ad-a286-dc35f063a460.
19. Polato, M. Dataset belonging to the help desk log of an Italian Company, 2017. https://doi.org/10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb.
20. Mannhardt, F. Hospital Billing - Event Log, 2017. https://doi.org/10.4121/uuid:76c46b83-c930-4798-a1c9-4be94dfeb741.
21. Hollmann, N.; Müller, S.; Eggensperger, K.; Hutter, F. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second, 2022. https://doi.org/10.48550/arXiv.2207.01848.
22. Botelho, J.; Choueiri, A.; Júnior, J.; Santos, E. A process mining and machine learning based approach for remaining time prediction of production orders. *Journal of Intelligent Manufacturing* **2026**. https://doi.org/10.1007/s10845-026-02792-9.
23. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794. https://doi.org/10.1145/2939672.2939785.

 doi:10.20944/preprints202607.0705.v1

24. Ke, G.; Meng, Q.; Finley, T.; Wang, T. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Proceedings of the Advances in Neural Information Processing Systems, 2017, Vol. 30. https://doi.org/10.5555/3294996.3295074.
25. Pellicani, A.; Ceci, M. Positional trace encoding for next activity prediction in event logs. *Knowl. Based Syst.* **2025**, *319*, 113544. https://doi.org/10.1016/j.knosys.2025.113544.
26. Elyasi, K.A.; van der Aa, H.; Stuckenschmidt, H. PGTNet: A Process Graph Transformer Network for Remaining Time Prediction of Business Process Instances. In Proceedings of the CAiSE. Springer, 2024, Lecture Notes in Computer Science, pp. 124–140. https://doi.org/10.1007/978-3-031-61057-8_8.
27. Wang, J.; Sui, Y.; Liu, C.; Shen, X.; Li, Z.; Yu, D. Multi-task Learning with Multi-gate Mixture of Transformer-Based Experts for Predictive Process Monitoring in Manufacturing. *Science Progress* **2024**, *107*. https://doi.org/10.1177/00368504241292196.
28. Casciani, A.; Bernardi, M.L.; Cimitile, M.; Marrella, A. Enhancing next activity prediction in process mining with Retrieval-Augmented Generation. *Inf. Syst.* **2026**, *137*, 102642. https://doi.org/10.1016/j.is.2025.102642.
29. Lischka, A.; Rauch, S.; Stritzel, O. Directly Follows Graphs Go Predictive Process Monitoring With Graph Neural Networks. *CoRR* **2025**, *abs/2503.03197*. https://doi.org/10.48550/arXiv.2503.03197.
30. Tian, Y.; Su, Y.; Zhang, R.; Du, Y.; Zhou, N.; Gao, X. Ensemble Prediction of Business Process Remaining Time Based on Random Forest and XGBoost. *Comput. Informatics* **2025**, *44*, 828–852. https://doi.org/10.31577/cai_2025_4_828.