Cached Summary Embeddings for Memory-Efficient EHR Inference

Rafi Al Attrach* RAFIAA@MIT.EDU
Massachusetts Institute of Technology (MIT), USA; Technical University of Munich (TUM), Germany

Rajna Fani* RAJNAF@MIT.EDU
Massachusetts Institute of Technology (MIT), USA; Technical University of Munich (TUM), Germany

David Restrepo DAVID.RESTREPO@CENTRALESUPELEC.FR
MICS, CentraleSupélec – Université Paris-Saclay, France

Yugang Jia YUGANG@MIT.EDU
Massachusetts Institute of Technology (MIT), USA

Leo Anthony Celi† LCELI@MIT.EDU
Massachusetts Institute of Technology (MIT), USA; Harvard Medical School, USA; Beth Israel Deaconess Medical Center, USA

Peter Schüffler† PETER.SCHUEFFLER@TUM.DE
Institute of Pathology, Technical University of Munich, Germany; Munich Center for Machine Learning (MCML), Germany

Abstract

Transformer-based clinical prediction models face a deployment challenge: processing long patient histories can require memory that exceeds available resources in resource-constrained settings. We propose a deployment architecture that separates expensive historical encoding from lightweight inference. In an offline preprocessing phase, a clinical language model compresses each patient's historical events into a fixed-size vector (768 dimensions, 5 KB per patient). At inference time, the prediction model processes only a short window of recent events, conditioned on the cached summary. Through 252 experiments on a 24-hour in-ICU mortality cohort from MIMIC-IV, we characterize when this architecture provides value. The benefit of cached summaries decays as the recent context window grows: a 6.5% relative AUROC improvement at N=8 recent events ($p < 0.001$) shrinks to a negligible 0.1% at N=256 (not statistically significant). We find that Feature-wise Linear Modulation (FiLM) outperforms token injection for integrating summaries ($p < 0.001$). Our results provide deployment guidance: when hardware constraints limit the recent context to 32 events or fewer, cached summaries recover meaningful predictive signal; when longer sequences are feasible, the caching overhead is not justified.

Data and Code Availability We use the publicly available MIMIC-IV v3.1 dataset (Johnson et al., 2023), processed into the Medical Event Data Standard (MEDS) format (McDermott et al., 2025) using the MIMIC-IV MEDS ETL pipeline (McDermott and Xu, 2025). Our experiments are conducted using the MEDS-Torch framework (Oufattole et al., 2024). Code for reproducing our experiments is publicly available at https://github.com/rafiattrach/cached-summaries-ehr-inference.

Institutional Review Board (IRB) This research was conducted using the publicly available, de-identified MIMIC-IV dataset and did not require additional IRB approval.

1. Introduction

Clinical prediction models built on Transformers (Vaswani et al., 2017) have achieved strong results across EHR-based tasks (Li et al., 2020; Rasmy et al., 2021). However, the quadratic memory complexity of self-attention creates a gap between research and deployment. A model that processes 512 events during

* Shared first author.
† Shared corresponding author.

training may exceed the memory available on deployment hardware, forcing practitioners to truncate patient histories and potentially lose predictive signal.

The Long-Context Challenge in Clinical AI The clinical significance of longitudinal patient history is well-documented. Chronic conditions, prior diagnoses, medication histories, and disease progression patterns often manifest over extended time periods that exceed the practical limits of current Transformer architectures. A patient's response to treatment today may depend critically on events from months or years prior. Yet the memory requirements of self-attention scale quadratically with sequence length: doubling the context window roughly quadruples memory consumption. This creates a fundamental tension between predictive completeness and computational feasibility.

Several research directions address this challenge. Architectural alternatives such as state-space models (Fallahpour et al., 2024) and sparse attention (Beltagy et al., 2020) reduce complexity. BigBird (Zaheer et al., 2020) combines local and global attention patterns. Mamba (Gu and Dao, 2023) achieves linear complexity through selective state spaces. Hardware optimizations like Flash Attention (Dao et al., 2022) improve throughput without changing the fundamental scaling. These approaches modify the model itself, requiring practitioners to adopt new architectures and potentially retrain from scratch.

An Orthogonal Approach: Input-Level Compression We investigate a strategy that modifies the *input representation* rather than the model architecture. Rather than processing the full patient history at inference time, we precompute a compressed summary of historical events using a clinical language model. The inference model then processes only a short "tail" of recent events, conditioned on the cached summary vector. This separation has concrete deployment benefits: the inference model's memory footprint depends only on the recent window length, not the patient's full history. The expensive summarization happens once, offline, and the resulting vectors are reused across all downstream prediction tasks.

The concept of compressing patient histories into dense vectors is established (Choi et al., 2016a,b). What remains unclear is *when* such compression provides value relative to simply processing more recent events. A practitioner choosing between (a) processing 64 recent events with a cached summary and (b) processing 128 recent events without a summary needs empirical guidance. The answer is not obvious: compression necessarily loses information, but recent events may already capture the most predictive signals.

Research Questions We frame this trade-off as a resource allocation problem. Given fixed computational resources, practitioners must choose between high-resolution recent context (the N most recent events processed in full detail) and compressed historical context (a summary of S prior events). Our goal is to characterize how this allocation affects predictive performance across different settings.

This paper provides that guidance through systematic evaluation. We ask three questions:

1. At what recent context lengths do cached historical summaries improve prediction?

2. How should summaries be integrated with recent event representations?

3. What are the computational costs of the caching architecture?

Scope of This Study We focus on the task with the longest event sequences in our MIMIC-IV cohort, 24-hour in-ICU mortality prediction (median 7,351 events per patient), where the recent-context vs. historical-summary trade-off has the largest practical consequence. BioClinical-ModernBERT serves as the offline summarization encoder and a single fixed-size cached vector as the patient representation; these choices are held fixed so the effects of recent-context length, summary size, summary source, and integration method can be isolated. The patterns we report apply within this setting; how they may differ for chronic-disease tasks or alternative summarization encoders is discussed in Section 4.4.

Our experiments reveal a pattern of diminishing returns. Cached summaries provide substantial benefit when the recent context window is short ($N \leq 32$) but negligible benefit when it is long ($N \geq 128$). We also find that the integration mechanism matters: FiLM conditioning (Perez et al., 2018) significantly outperforms token injection. These findings have direct implications for clinical AI deployment, where memory constraints often dictate model design choices.

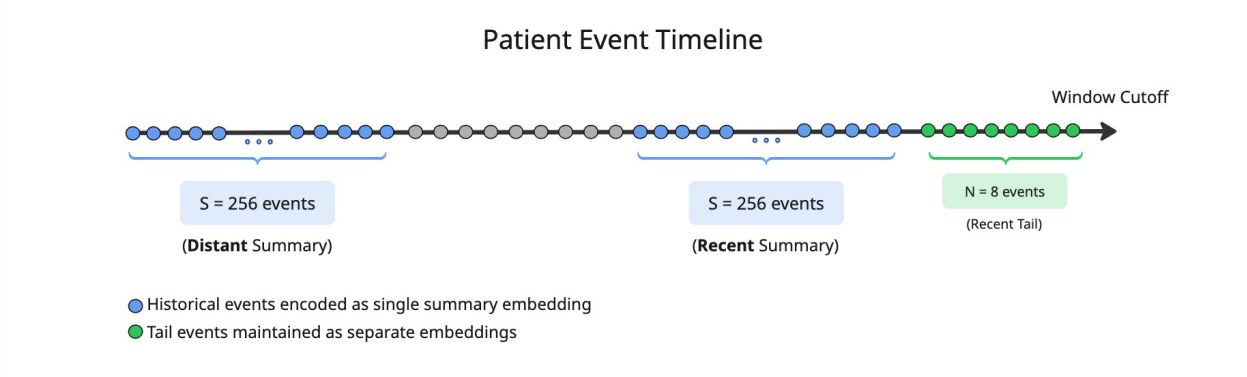

Figure 1: The deployment architecture. Historical events (blue) are encoded offline by a clinical language model into a cached summary vector. At inference, the prediction model processes only the recent tail (green) conditioned on the cached summary. The summary can represent either distant history or the period immediately preceding the tail.

2. Methods

We evaluate the proposed deployment architecture using the MEDS-Torch framework (Oufattole et al., 2024). Technical details appear in Appendix A.

2.1. Data Representation

Each clinical event is represented as a triplet: a medical code, a time delta, and an optional numeric value. Time is stored as a delta (in days) relative to the previous event in the sequence, so timestamps are represented as irregularly spaced gaps rather than absolute times. For example, a glucose measurement in the structured EHR becomes `LAB//220621//mg/dL` with description "Glucose [Mass/volume] in Serum or Plasma (mg/dL)". A concrete triplet instance would be (`LAB//220621//mg/dL`, Δt=0.08 days, 120), meaning a serum glucose value of 120 mg/dL observed 0.08 days after the prior event. This representation captures what happened (the code), when it happened (the time delta), and its magnitude (the numeric value, when applicable).

The MEDS format (McDermott et al., 2025) standardizes this representation across EHR sources, providing a common vocabulary and schema that enables reproducible research. Our cohort has a median of 7,351 events per patient (see Table 1), spanning medications, laboratory results, procedures, diagnoses, and vital signs. This event density reflects the intensive monitoring characteristic of ICU settings.

2.2. Dataset and Prediction Task

We use MIMIC-IV (Johnson et al., 2023), a publicly available critical care database containing de-identified health records from Beth Israel Deaconess Medical Center. The dataset contains 364,627 unique individuals with 546,028 hospitalizations and 94,458 ICU stays spanning 2008-2022, with rich longitudinal data including vital signs, laboratory measurements, medications, procedures, and diagnoses.

We focus on predicting in-ICU mortality within 24 hours, selected because this task has the longest median event sequences in the MEDS-formatted cohort. This characteristic makes it well-suited for evaluating long-context strategies:

1. **Long event sequences**: The median of 7,351 events per patient far exceeds what standard Transformers can process, necessitating truncation or compression.

2. **Clinical relevance**: Early mortality prediction enables proactive clinical intervention and resource allocation.

3. **Historical dependence**: Mortality risk is influenced by both acute physiological changes (captured in recent events) and chronic conditions (potentially requiring historical context).

The cohort contains 35,550 unique patients with valid prediction windows. Table 1 contrasts the median event count for this task with three other common MEDS-formatted MIMIC-IV tasks; in-ICU mortality has roughly an order of magnitude more events per patient than hospital mortality, readmission, or post-discharge mortality, which is what makes it the most useful stress test for long-context strategies in this dataset.

Table 1: Event sequence lengths in MEDS-formatted MIMIC-IV. Selected task in bold.

Task	Patients	Med. Events
ICU mort. (24h)	**35,550**	**7,351**
Hosp. mort. (24h)	164,900	985
Readmission (30d)	145,004	814
Post-disch. mort. (1y)	210,022	541

2.3. Model Architectures

We compare three architectures built on a standard Transformer encoder (architecture details in Appendix A).

Baseline Processes only the N most recent events. Memory scales with N.

Oracle Processes the full sequence of $N+S$ events. Memory scales with total sequence length.

Cached Summary Architecture Processes N recent events plus a precomputed summary vector of S historical events. Memory scales with N only, independent of total history length, because the historical context is compressed into a fixed-size vector before inference.

2.4. Summary Generation

Historical summaries are generated in an offline preprocessing step:

1. Extract the S historical events for each patient window.
2. Convert medical codes to text descriptions (e.g., `PROCEDURE//START//227194` → "Removal of endotracheal tube started").
3. Concatenate descriptions and encode with BioClinical-ModernBERT (Sounack et al., 2025), a clinical language model supporting sequences up to 8,192 tokens.
4. Extract the final hidden state as a 768-dimensional summary vector.
5. Cache to disk (5.35 KB per patient, including metadata).

The preprocessing cost is detailed in Section 3.7 and Appendix B.

Code-to-Text Mapping Default MEDS vocabulary descriptions cover roughly 25% of the unique medical codes in our cohort, with the remainder mapped to opaque identifiers (e.g., `LAB//220615//mg/dL`). We extended coverage to 100% by querying the MIMIC-IV MCP Server (Attrach et al., 2025) for human-readable labels drawn from the source MIMIC-IV tables, so that the example above resolves to "Creatinine [Mass/volume] in Serum or Plasma". Without this enhancement, the summarization encoder would receive opaque identifier strings for the majority of events. Examples spanning diagnoses, medications, infusions, procedures, labs, and administrative events appear in Appendix G.

2.5. Summary Integration Methods

We compare two approaches for conditioning the inference model on cached summaries.

Token Injection The summary vector replaces the code embedding at the first position of the recent event sequence. Time and value embeddings at that position are zeroed. The summary influences other tokens only through self-attention.

Feature-wise Linear Modulation (FiLM) The summary vector $\mathbf{z}_s \in \mathbb{R}^{768}$ generates scale and shift parameters that modulate all code embeddings:

$$\gamma = W_\gamma \mathbf{z}_s + b_\gamma \tag{1}$$

$$\beta = W_\beta \mathbf{z}_s + b_\beta \tag{2}$$

$$\mathbf{e}'_{\text{code}} = \gamma \odot \mathbf{e}_{\text{code}} + \beta \tag{3}$$

where $W_\gamma, W_\beta \in \mathbb{R}^{d \times 768}$ are learned projections. This allows historical context to directly influence how recent events are represented before self-attention.

2.6. Experimental Protocol

We conducted a grid search over $N \in \{8, 16, 32, 64, 128, 256\}$ and $S \in \{128, 256, 512\}$. We also compared summaries from **recent** history (immediately preceding the tail) versus **distant** history (earliest events in the record), as shown in Figure 1. Each configuration was run with three random seeds.

We report AUROC as the primary metric, consistent with prior work. We confirmed that AUPRC tracked AUROC trends across configurations. Statistical significance is assessed using paired Wilcoxon signed-rank tests. For our two primary cross-configuration claims (FiLM vs. token injection and recent vs. distant summary source), we apply a Bonferroni correction at $\alpha = 0.01$; both findings remain significant after correction. Per-comparison effect sizes (Cohen's d) and full statistical details appear in Appendix D.

Training Hyperparameters We use Adam with learning rate 10^{-3}, batch size 64, and an early-stopping patience of 3 epochs. These defaults were selected after a sweep of alternatives: higher learning rates (10^{-2}) caused training instability, lower rates (10^{-4}) converged more slowly without improving final test performance, and alternative batch sizes (32, 128) and patience values (5) had minor effects on the test metrics. Full sensitivity results appear in Appendix H.

3. Results

We conducted 252 experiments spanning 6 values of N, 3 values of S, 2 integration methods, 2 summary sources, and 3 random seeds. This systematic evaluation enables robust conclusions about when cached summaries provide value.

3.1. When Do Cached Summaries Help?

The benefit of cached summaries depends critically on the recent context length. Figure 2 shows that relative AUROC improvement decays from 6.5% at N=8 ($p < 0.001$) to 0.1% at N=256 (not statistically significant). This pattern holds across all summary sizes tested and represents the central finding of our evaluation.

The intuition behind this pattern is straightforward: when the recent context window is short, the model lacks access to events that may be predictive.

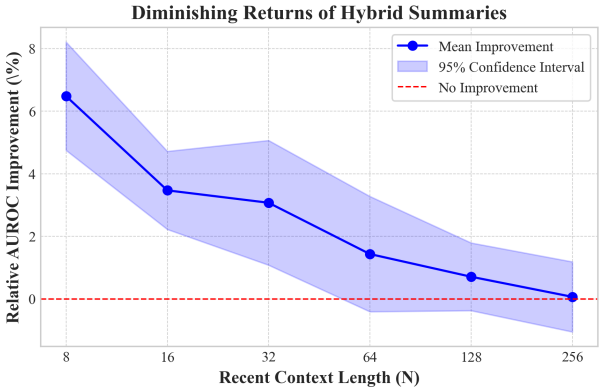

Figure 2: Relative AUROC improvement of the cached summary architecture (FiLM, S=256) over the baseline. Improvement is substantial at small N but negligible at large N. Error bars show 95% confidence intervals from three seeds.

The compressed summary provides a mechanism to recover some of this lost signal. As the recent window grows, the model increasingly has direct access to predictive events, and the marginal value of compressed historical information diminishes.

Statistical Significance Thresholds The improvement at N=8 is highly significant ($p < 0.001$), and remains significant through N=64 ($p < 0.05$). At N=128 and N=256, the improvements (0.5% and 0.1% respectively) are not statistically significant, suggesting that at these context lengths, cached summaries do not provide reliable benefit. Table 2 reports the per-N relative improvements together with their significance levels.

3.2. Interaction Between Summary Size and Context Length

Figure 3 shows the interaction between summary size and recent context length. Larger summaries provide meaningful gains only when recent context is limited. At N=8, increasing S from 128 to 512 provides visible performance gains. At N=256, all three summary sizes perform nearly identically, confirming that larger summaries are not beneficial when recent context is already sufficient.

This finding has practical implications for deployment. Larger summary sizes require more prepro-

Table 2: Relative AUROC improvement (%) of the cached summary architecture (FiLM, S=256) over the baseline at each recent context length N. Statistical significance from paired Wilcoxon tests over three seeds.

N	Rel. Improvement	Significance
8	6.5%	$p < 0.001$
16	4.2%	$p < 0.01$
32	2.1%	$p < 0.05$
64	1.0%	$p < 0.05$
128	0.5%	n.s.
256	0.1%	n.s.

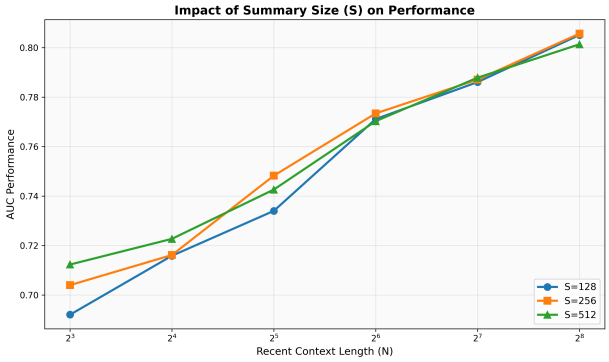

Figure 3: Impact of summary size (S) across recent context lengths (N). The benefit of larger summaries diminishes as recent context increases.

cessing time (encoding more historical events) and slightly more storage. When recent context is constrained, this investment is justified. When $N \geq 128$ is feasible, practitioners should skip the summary infrastructure entirely.

3.3. Absolute Performance Comparison

Figure 4 shows absolute AUROC values for the baseline, cached summary (FiLM), and oracle architectures. The cached summary architecture partially closes the gap between the baseline and the oracle, most effectively at small N. The oracle represents an upper bound: processing the full $N+S$ sequence with full attention.

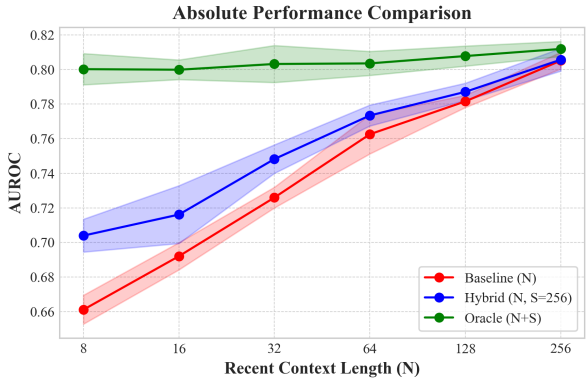

Figure 4: Absolute AUROC comparison. The cached summary architecture (FiLM, S=256) sits between baseline and oracle, with the gap narrowing as N increases.

Notably, the oracle's performance remains relatively stable across N values, indicating that once sufficient context is available, the model extracts consistent predictive signal. The baseline and cached summary architectures converge toward the oracle as N increases, suggesting that recent events carry the majority of predictive information for this task.

Table 3 reports the four primary metrics aggregated across configurations. The cached architecture with FiLM integration sits between baseline and oracle on AUROC, AUPRC, and loss, while accuracy remains close to ceiling for all four architectures because the prediction task is highly imbalanced. The oracle's lower variance across seeds is consistent with having more information to learn from.

Table 3: Performance metrics (mean ± std across seeds, aggregated across configurations). Cached results use recent summaries with S=256.

Architecture	AUROC	AUPRC	Accuracy	Loss
Baseline	0.738 ± 0.051	0.229 ± 0.048	0.914 ± 0.001	0.234 ± 0.045
Cached (Token)	0.745 ± 0.042	0.226 ± 0.047	0.914 ± 0.001	0.231 ± 0.044
Cached (FiLM)	0.754 ± 0.037	0.231 ± 0.044	0.914 ± 0.001	0.228 ± 0.042
Oracle	0.804 ± 0.007	0.267 ± 0.012	0.917 ± 0.001	0.198 ± 0.008

3.4. Integration Method Comparison

A critical design decision for the cached summary architecture is how to integrate the historical summary with recent event representations. We compare two mechanisms: token injection (treating the summary as a sequence element) and FiLM conditioning (using the summary to modulate all event embeddings).

Table 4 summarizes ablation results. FiLM integration significantly outperforms token injection ($p < 0.001$), with a mean AUROC improvement of 0.88 percentage points. This difference is consistent across all N and S configurations tested.

Table 4: Ablation results averaged across all N and S configurations. P-values from paired Wilcoxon tests.

Method	Mean AUROC
Integration Method ($p < 0.001$)	
FiLM	**0.7542**
Token Injection	0.7454
Summary Source ($p < 0.01$)	
Recent History	**0.7542**
Distant History	0.7453

Figure 5 shows the FiLM advantage across context lengths. The performance gap between FiLM and token injection is relatively consistent across N values, suggesting that the modulatory mechanism provides benefit regardless of how much recent context is available.

Table 5 reports Cohen's d for the four primary comparisons, classified as small ($d < 0.2$), medium ($0.2 \leq d < 0.8$), or large ($d \geq 0.8$). Both ablation findings (FiLM vs. token injection, recent vs. distant summary source) show medium effect sizes that are stable across N. The cached-vs-baseline contrast, in contrast, shifts from a large effect at N=8 to a small effect at N=256, quantifying the diminishing-returns pattern.

Table 5: Effect sizes (Cohen's d) for primary comparisons.

Comparison	Cohen's d	Size
FiLM vs. Token (all N)	0.42	Medium
Recent vs. Distant (all N)	0.38	Medium
Cached vs. Baseline (N=8)	1.12	Large
Cached vs. Baseline (N=256)	0.18	Small

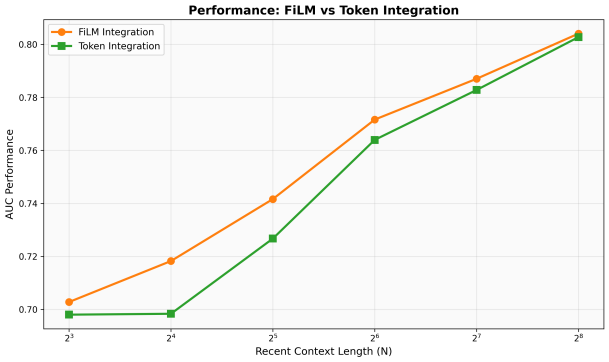

Figure 5: FiLM versus Token integration across recent context lengths. FiLM provides consistent improvement across all N values.

Why FiLM Outperforms Token Injection Token injection treats the summary as an additional sequence element that must compete for attention with recent events. The summary's influence is mediated

entirely through the self-attention mechanism, meaning it can only affect other tokens indirectly. In contrast, FiLM uses the summary to generate scale (γ) and shift (β) parameters that directly modulate the embeddings of all recent events before self-attention. This allows historical context to inform how recent events are *represented*, not just how they are *attended to*.

This distinction aligns with clinical reasoning. A patient's medical history provides context for interpreting current symptoms rather than being an observation itself. A glucose reading of 180 mg/dL has different implications for a diabetic patient than for one without metabolic history. FiLM enables this contextual interpretation by modulating how the glucose event is represented.

3.5. Summary Source: Recent vs Distant History

We also compared summaries from two temporal sources:

- **Recent**: The S events immediately preceding the N-event tail.
- **Distant**: The first S events in the patient's record.

Recent summaries significantly outperform distant summaries ($p < 0.01$). This finding suggests that for acute care prediction, events closer in time to the prediction window carry more predictive signal than formative events from early in the patient's history. The recent summary captures the patient's trajectory leading into the current state, while the distant summary may contain events that are no longer clinically relevant.

3.6. When Cached Summaries Do Not Help

Three patterns explain when the cached summary contributes little signal beyond the baseline.

Patients with short total histories When a patient has fewer than $N + S$ recorded events, the summary covers events the recent window already contains. The cached architecture and the baseline see effectively the same information, leaving no margin for the summary to contribute.

Acute events dominate the tail For cases where mortality may be driven primarily by acute deterioration in the recent window (cardiac arrest, massive hemorrhage), the predictive signal could be concentrated in the tail itself. Historical context likely plays a smaller role in these cases and the summary's contribution may be limited.

Misaligned summary content If the summary covers a hospitalization for a condition unrelated to the prediction target, the compressed representation may inject noise rather than signal. Condition-aware or task-aware summarization, which we did not evaluate, could mitigate this.

3.7. Computational Cost Analysis

Table 6 compares the three architectures. The cached summary architecture maintains inference memory comparable to the baseline while improving AUROC.

Table 6: Computational comparison (averaged across configurations). Preprocessing cost is amortized (see Appendix B).

Arch.	AUROC	Mem.
Baseline	0.738	591 MB
Cached	0.754	594 MB
Oracle	0.804	1,292 MB

Memory Scaling with N Table 7 reports measured GPU memory across recent context lengths. The cached architecture stays within 3 MB of the baseline at every N (one fixed-size summary embedding plus the FiLM modulation overhead). The oracle, by contrast, processes $N+S$ events and consumes approximately $2\times$ the memory of the cached architecture at every operating point. This is the central deployment property: inference memory scales with N alone, not with total history length.

Preprocessing Cost Summary generation requires 22 ms per patient on GPU (20 ms model forward pass, 2 ms tokenization). For our cohort of 35,550 patients, preprocessing takes approximately 72 minutes per $(S, N, \text{variant})$ configuration. Storage is 5.35 KB per patient (768 floats plus metadata), totaling 190 MB for the full cohort. See Appendix B for complete cost breakdown.

Amortization The preprocessing cost is incurred once. In our evaluation (252 experiments reusing cached summaries), the effective per-experiment

Table 7: GPU memory (MB) by recent context length (N). Oracle processes $N + S$ events with S=256; "Oracle Total" shows that total sequence length.

N	Baseline	Cached	Oracle	Oracle Total
8	312	315	1,089	264
16	348	351	1,124	272
32	421	424	1,198	288
64	567	570	1,345	320
128	859	862	1,638	384
256	1,443	1,446	2,614	512

overhead was 0.29 minutes, adding 2.5% to training time. For deployment, summaries are computed once per patient and reused across all prediction tasks.

Training-Time Overhead Training the cached architectures incurs additional time from per-batch summary loading and FiLM modulation; detailed timings appear in Appendix C. This is a one-time training-side cost and does not affect deployment-time inference, which is dominated by the recent-window length and matches the baseline's footprint (Table 6).

4. Discussion

Our results characterize a fundamental trade-off in clinical AI deployment: the balance between historical completeness and computational feasibility. The clear pattern of diminishing returns provides actionable guidance for practitioners facing this trade-off.

4.1. Practical Guidelines

Our results quantify the trade-off between high-resolution recent context (the N most recent events) and compressed historical context (a summary of S prior events):

- **When inference hardware limits $N \leq 32$**: Cached summaries provide 2–6% relative AUROC improvement. The preprocessing cost (72 min per cohort configuration) and storage (190 MB) are justified by the meaningful performance gains.
- **When $N \geq 128$ is feasible**: Cached summaries add less than 1% improvement (not statistically significant at N=256). The preprocessing overhead is not justified; resources are better spent increasing N directly.
- **Integration choice**: Use FiLM over token injection. The FiLM approach allows historical context to modulate how recent events are interpreted, rather than competing with them for attention.

4.2. Implications for Model Design

The superiority of FiLM over token injection has broader implications for clinical AI architecture design. Token injection forces historical context to compete with recent events for attention, treating the summary as just another sequence element. FiLM instead allows historical context to *condition* how recent events are interpreted, a fundamentally different computational role.

This distinction maps to clinical reasoning. When a physician reviews a patient's current lab results, they do so in the context of the patient's history. The history does not compete for attention with current observations; rather, it provides interpretive context. A creatinine level of 2.0 mg/dL has different implications depending on whether the patient has chronic kidney disease. FiLM captures this contextual interpretation by modulating the representation of current events based on historical context.

Where the Summary Helps Most The effect-size pattern in Table 5 is consistent with the model using the cached vector as conditioning rather than as competing sequence content. FiLM's advantage over token injection is stable across all N values (Cohen's $d = 0.42$), whereas the cached-vs-baseline advantage shrinks sharply with N (large effect at N=8 with $d = 1.12$, small at N=256 with $d = 0.18$). The asymmetry has a mechanistic reading. FiLM (Section 2.4) modulates the representations of the recent tokens themselves before self-attention, so its benefit is available at every N because there are always tokens to modulate; the magnitude of the modulation does not depend on tail length. Token injection contributes the summary as one additional element among N recent tokens, and its relative influence under self-attention falls as N grows. Whether the summary helps at all then depends on whether the tail already carries the predictive signal: the failure patterns in Section 3.6 (short total histories, acute events in the tail, misaligned summary content) are

the cases where there is little remaining signal for any integration mechanism to recover.

4.3. Deployment Scenarios

The cached summary architecture is most appropriate for specific deployment scenarios:

Edge Devices and Resource-Constrained Environments Hospitals deploying AI at the point of care may face strict memory constraints. A clinical workstation or embedded device may not have the GPU memory to process full patient histories. The cached summary architecture enables deployment on such devices: the expensive summarization happens once (potentially in the cloud), and the lightweight inference model processes only recent events plus a cached vector. The per-N footprints in Table 7 translate directly to deployment thresholds: the cached architecture at N=32 uses 424 MB and so fits within 512 MB of available GPU memory while the oracle at the same setting (1,198 MB) does not, and at N=128 the cached architecture (862 MB) still fits within 1 GB while the oracle (1,638 MB) does not.

High-Throughput Serving Clinical decision support systems serving many concurrent requests benefit from predictable, low-latency inference. Processing 32 tokens plus a cached summary is faster and more memory-efficient than processing 512 tokens, enabling higher throughput with the same hardware.

Multi-Task Systems When multiple prediction tasks (mortality, readmission, length of stay) share the same patient cohort, the preprocessing investment is amortized across all tasks. Summaries are computed once per patient and reused, making the per-task overhead negligible.

Operational Considerations Caching introduces operational overhead beyond the standard model lifecycle: a per-patient embedding store and a preprocessing job that must be re-run when the underlying summarization encoder is updated. These costs are concentrated offline and amortize across downstream tasks (Section 3.7), but should be planned for in deployments without existing vector-store infrastructure.

4.4. Limitations and Future Directions

Our evaluation focuses on 24-hour ICU mortality prediction, a task with long event sequences that makes it well-suited for stress-testing long-context strategies. The diminishing returns pattern we observe may vary for tasks where distant history carries more predictive weight, such as chronic disease progression.

Limitations Beyond the single-task scope, three further axes of our evaluation are held fixed and bear on generality. The MIMIC-IV cohort reflects the patient population and documentation practices of a single academic medical center, so our quantitative thresholds may not transfer cleanly to other institutions or EHR systems. BioClinical-ModernBERT was selected for its clinical pretraining and 8,192-token context support, but other clinical encoders (such as ClinicalBERT or GatorTron) could produce summaries with different characteristics, in particular different relative gains across N. The summarizer is also frozen at inference time; end-to-end training of the summary encoder could improve summary quality, at the cost of the offline amortization that makes the deployment architecture economical.

Beyond a Single Fixed-Size Vector Our cached representation is a single 768-dimensional vector per patient, chosen for deployment simplicity and for clean isolation of the summary-source and summary-size variables in our experimental design. For very long histories (patients with 10,000+ events) and for tasks where multi-scale structure (chronic conditions versus acute episodes) carries independent signal, a hierarchical or multi-vector design may preserve more clinically distinct content at moderate-to-large N, where our single-vector design shows little benefit. A natural variant would compute one summary per hospitalization or per fixed time window, optionally combined with a single global summary; this would shift the diminishing-returns threshold rightward at the cost of additional preprocessing and a more complex cache schema. Multi-vector caches would also enable integration mechanisms beyond FiLM, such as cross-attention over the cached sequence, which benefits from having multiple entries to attend across. FiLM was a deliberate fit for the single-vector regime: with only one cached entry, cross-attention reduces to a fixed projection of that vector and loses the per-query selection that motivates it.

Adaptation, Reuse, and Interpretability A natural extension of the experimental sweep is adaptive summary generation, where summary length varies per patient based on history richness, reducing preprocessing for short histories while support-

ing more detail for complex cases. A second direction is cross-task summary reuse: our preprocessing pipeline produces task-agnostic summaries, but it remains open whether summaries optimized for one task (mortality) transfer cleanly to others (readmission, length of stay), which would multiply the amortization benefit. Finally, the cached vector is currently dense and uninterpretable; structured summaries that retain identifiable clinical concepts could let clinicians inspect what historical content drove a given prediction, which is increasingly relevant for clinical AI in regulated settings.

5. Conclusion

We evaluated a deployment architecture that separates historical encoding (offline, cached) from clinical inference (online, memory-efficient). Across 252 experiments, we find three consistent patterns. First, the value of cached summaries depends on the amount of recent context: gains are substantial at small N and diminish to near zero at large N. Second, FiLM integration outperforms token injection, indicating that historical context is more useful as conditioning signal than as an additional sequence element. Third, summaries drawn from recent history outperform summaries drawn from distant history ($p < 0.01$), suggesting that the most useful compressed context is the interval immediately preceding the high-resolution tail. For deployments where the inference budget caps the recent-event window at a few dozen events, cached summaries provide a practical way to recover additional signal from longitudinal records while keeping inference memory predictable.

Acknowledgments

We thank members of the MIT Laboratory for Computational Physiology for their feedback and discussions during weekly lab meetings throughout this project. We also thank Matthew B. A. McDermott for helpful discussions on the experimental design.

References

Rafi Al Attrach, Pedro Moreira, Rajna Fani, Renato Umeton, and Leo Anthony Celi. Conversational llms simplify secure clinical data access, understanding, and analysis. *arXiv preprint arXiv:2507.01053*, 2025.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In *Machine learning for healthcare conference*, pages 301–318. PMLR, 2016a.

Edward Choi, Andy Schuetz, Walter F Stewart, and Jimeng Sun. Medical concept representation learning from electronic health records and its application on heart failure prediction. *arXiv preprint arXiv:1602.03686*, 2016b.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35:16344–16359, 2022.

Adibvafa Fallahpour, Mahshid Alinoori, Wenqian Ye, Xu Cao, Arash Afkanpour, and Amrit Krishnan. Ehrmamba: Towards generalizable and scalable foundation models for electronic health records. *arXiv preprint arXiv:2405.14567*, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1): 1, 2023.

Yikuan Li, Shishir Rao, José Roberto Ayala Solares, Abdelaali Hassaine, Rema Ramakrishnan, Dexter Canoy, Yajie Zhu, Kazem Rahimi, and Gholamreza Salimi-Khorshidi. Behrt: transformer for electronic health records. *Scientific reports*, 10(1):7155, 2020.

Matthew McDermott and Justin Xu. MIMIC-IV MEDS: An ETL pipeline to extract MIMIC-IV data into the MEDS format, May 2025. URL https://github.com/Medical-Event-Data-Standard/MIMIC_IV_MEDS.

Matthew B. A. McDermott, Justin Xu, Teya S. Bergamaschi, Hyewon Jeong, Simon A. Lee, Nassim Oufattole, Patrick Rockenschaub, Kamile

Stankeviciute, Ethan Steinberg, Jimeng Sun, Robin P. van de Water, Michael Wornow, John Wu, and Zhenbang Wu. Meds: Building models and tools in a reproducible health ai ecosystem. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, pages 6243–6244, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3737608. URL https://doi.org/10.1145/3711896.3737608.

Nassim Oufattole, Teya Bergamaschi, Pawel Renc, Aleksia Kolo, Matthew BA McDermott, and Collin Stultz. Meds-torch: An ml pipeline for inductive experiments for ehr medical foundation models. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):86, 2021.

Thomas Sounack, Joshua Davis, Brigitte Durieux, Antoine Chaffin, Tom J Pollard, Eric Lehman, Alistair EW Johnson, Matthew McDermott, Tristan Naumann, and Charlotta Lindvall. Bioclinical modernbert: A state-of-the-art long-context encoder for biomedical and clinical nlp. *arXiv preprint arXiv:2506.10896*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

Appendix A. Technical Details

Hardware Experiments were conducted on AWS `ml.g5.4xlarge` instances with NVIDIA A10G GPUs (24 GB VRAM).

Model Architecture The Transformer encoder uses 2 layers, 2 attention heads, and 128-dimensional token embeddings. This architecture follows MEDS-Torch defaults for the triplet encoder experiments.

Training Configuration All models were trained for up to 10 epochs with early stopping (patience=3). We used the Adam optimizer with learning rate 10^{-3}, batch size 64, and 32-bit precision. We used a patient-level 80/10/10 split (`train`, `tuning`, `held_out`) from the respective cohort.

FiLM Implementation The FiLM layer uses two linear projections $W_\gamma, W_\beta \in \mathbb{R}^{128\times768}$ to map the 768-dimensional summary to 128-dimensional scale (γ) and shift (β) vectors:

$$\gamma = W_\gamma \mathbf{z}_s + b_\gamma \in \mathbb{R}^{128} \tag{4}$$

$$\beta = W_\beta \mathbf{z}_s + b_\beta \in \mathbb{R}^{128} \tag{5}$$

$$\mathbf{e}'_{\text{code}} = \gamma \odot \mathbf{e}_{\text{code}} + \beta \tag{6}$$

These parameters are learned jointly with the rest of the model.

Appendix B. Preprocessing Cost Details

Summarization Model We use BioClinical-ModernBERT-base (Sounack et al., 2025) (hidden size 768, 22 layers, 12 attention heads; 150M parameters) as a long-context, domain-adapted clinical encoder to generate fixed-size cached summaries.

Per-Sample Timing On a single A10G GPU:

- Model forward pass: 20.3 ms (median)
- Tokenization: 1.5 ms (median)
- Total per patient: 21.9 ms (median)
- Cold start overhead: 2.6 seconds (first sample only)

Cohort-Level Average Timing For our cohort of 35,550 patients:

- Wall-clock time per configuration: 72.5 minutes
- Total for all 36 configurations tested: 43.5 GPU-hours

Storage Each cached summary file contains:

- 768-dimensional embedding (float32): 3.0 KB
- Metadata (patient ID, window indices, configuration): 2.35 KB
- Total per patient: 5.35 KB
- Full cohort per configuration: 190 MB

Amortization The 252 training experiments in this paper reused cached summaries generated during preprocessing. Since each configuration's summaries (72.5 minutes to generate) were reused across multiple experiments, the effective preprocessing overhead per training run was $72.5/252 \approx 0.29$ minutes, adding approximately 2.5% to the average training time of 11.6 minutes.

Table 8: Preprocessing cost summary.

Metric	Value
Time per patient	22 ms
Time per cohort configuration	72.5 min
Storage per patient	5.35 KB
Storage per cohort	190 MB
Amortized cost (252 experiments)	0.29 min/exp

Appendix C. Computational Resource Utilization

Table 9 provides a breakdown of GPU memory and training time across architectures, aggregated across all configurations and three random seeds. Memory measurements were taken at peak allocation during forward passes. The cached summary architectures require additional training time due to the overhead of loading and processing the summary embeddings at each batch; this is a one-time training-side cost and does not affect deployment-time inference, where the cached architecture matches the baseline's footprint.

Appendix D. Statistical Analysis Details

D.1. Hypothesis Testing Framework

We use paired Wilcoxon signed-rank tests to compare configurations, as this non-parametric test does not assume normality of performance differences. For each comparison, we pair results by random seed and compute the test statistic over the three seeds.

Given the multiple comparisons in our study, we report both raw p-values and Bonferroni-corrected thresholds where applicable. For our primary claims (FiLM vs token, recent vs distant), we use a significance threshold of $\alpha = 0.01$ after correction.

D.2. Confidence Interval Construction

The 95% confidence intervals in Figure 2 are computed as:

$$\text{CI}_{95\%} = \bar{x} \pm 1.96 \times \frac{s}{\sqrt{n}} \tag{7}$$

where $\bar{x}$ is the sample mean, s is the sample standard deviation, and $n = 3$ is the number of seeds. We acknowledge that with only three seeds, these intervals rely on asymptotic normality assumptions.

Appendix E. Ablation Study Details

E.1. Summary Source: Recent vs Distant

The "recent" summary covers the S events immediately preceding the N tail events. The "distant" summary covers the first S events in the patient's record. Across all configurations, recent summaries significantly outperform distant summaries ($p < 0.01$), with a mean AUROC difference of approximately 0.9 percentage points.

The advantage of recent summaries aligns with clinical intuition: events closer in time to the prediction window are more likely to reflect the patient's current state. The performance gap narrows at larger N because the tail itself captures more recent context.

E.2. Summary Size Sensitivity

We evaluated summary sizes $S \in \{128, 256, 512\}$. S=256 provided the best balance across configurations, though differences between S=256 and S=512 were not statistically significant. This suggests that the summarization model may saturate in its ability to compress additional events, or that events beyond 256 provide marginal additional signal for this prediction task.

Table 9: Computational resource utilization (mean ± std across configurations).

Architecture	GPU Memory	Training Time
Baseline	591 ± 243 MB	3.3 ± 0.7 min
Cached (Token)	593 ± 243 MB	11.4 ± 2.5 min
Cached (FiLM)	594 ± 243 MB	11.6 ± 2.7 min
Oracle	1,292 ± 249 MB	3.6 ± 1.2 min

Appendix F. Code-to-Text Mapping Details

F.1. Mapping Coverage

The MEDS vocabulary contains 7,351 unique medical codes in our MIMIC-IV cohort. The default MEDS code descriptions provided coverage for approximately 25% of codes, with the remainder mapped to generic identifiers.

To address this, we used the MIMIC-IV MCP Server (Attrach et al., 2025) to retrieve human-readable labels directly from the MIMIC-IV database, increasing description coverage to 100%. This ensures that all medical codes have meaningful textual representations for the summarization model. Representative examples are shown in Appendix G.

Appendix G. Enhanced Mapping Examples

To illustrate the scope of the enhanced mapping process, Table 10 shows representative examples across common event categories. In our cohort, the enhanced mapping replaced sparse identifiers for previously unmapped codes and achieved 100% description coverage (compared to approximately 25% coverage under the default mappings).

This representative sample shows how the enhanced mapping converts cryptic identifiers into clinically meaningful descriptions. For example, `INFUSION_START//221794` maps to "Infusion of furosemide (Lasix) started", which provides semantic context that a text encoder can leverage. Procedure codes such as `PROCEDURE//START//227194` map to "Removal of endotracheal tube started", which makes the clinical action explicit. The enhanced mapping covers diagnoses, medications, infusions, procedures, laboratory measurements (including units), and administrative events, so that text-based summaries are grounded in meaningful descriptions throughout the record.

Appendix H. Hyperparameter Sensitivity

H.1. Learning Rate

We evaluated learning rates in $\{10^{-4}, 10^{-3}, 10^{-2}\}$. The default 10^{-3} provided stable convergence across all configurations. Higher rates (10^{-2}) led to training instability in some configurations, while lower rates (10^{-4}) converged more slowly without improving final performance.

H.2. Batch Size

We evaluated batch sizes in $\{32, 64, 128\}$. Batch size 64 was selected as the default, balancing memory efficiency and gradient noise. Larger batches (128) showed marginal improvements (<0.5% AUROC) but required more memory.

H.3. Early Stopping Patience

We used patience=3 epochs based on validation AUROC. Experiments with patience=5 showed no improvement in final test performance but increased training time by 20% on average due to additional epochs before convergence.

Appendix I. Reproducibility Checklist

I.1. Compute Resources

- Hardware: AWS `ml.g5.4xlarge` (NVIDIA A10G, 24GB VRAM).
- Summary precomputation: 36 configurations; approximately 70–75 minutes per configuration on a single A10G; approximately 40–45 instance-hours in total.

- Model training: 252 training runs on a single A10G. Wall-clock time per run varies by model variant and context length, so we report training time per run rather than summarizing with a single aggregate GPU-hour estimate.
- Estimated cost (order of magnitude): on-demand `ml.g5.4xlarge` pricing varies by region and time; at roughly $2/hour, the precomputation cost is on the order of $100.

I.2. Software Versions

- Python: 3.11.11
- PyTorch: 2.7.0
- PyTorch Lightning: 2.5.1
- MEDS-Torch: 0.0.8
- Transformers: 4.55.2
- NumPy: 1.26.4

I.3. Data Availability

- MIMIC-IV v3.1: Available via PhysioNet (credentialed access required)
- MEDS preprocessing scripts and full experimental code: https://github.com/rafiattrach/cached-summaries-ehr-inference
- Pretrained model: BioClinical-ModernBERT-base from HuggingFace

I.4. Random Seeds

All experiments were run with seeds $\{0, 1, 2\}$. Seeds were set using PyTorch Lightning's seed utility, which configures PyTorch, NumPy, and Python's random module for reproducibility.

Table 10: Representative examples from the enhanced mapping showing the transformation from structured codes to clinical descriptions.

Original Code	Enhanced Description
Demographics and Baseline Factors	
GENDER//F	Gender recorded as female
GENDER//M	Gender recorded as male
LAB//226512//kg	Body weight (kg)
LAB//226707//Inch	Body height (Inch)
BMI	Body mass index (BMI)
Diagnoses (ICD-9 and ICD-10)	
DIAGNOSIS//ICD//10//E119	Type 2 diabetes mellitus without complications (ICD-10: E119)
DIAGNOSIS//ICD//10//Z87891	Personal history of nicotine dependence (ICD-10: Z87891)
DIAGNOSIS//ICD//9//4019	Unspecified essential hypertension (ICD-9: 4019)
DIAGNOSIS//ICD//10//I10	Essential (primary) hypertension (ICD-10: I10)
Medications	
MEDICATION//Acetaminophen//Administered	Acetaminophen administered
MEDICATION//START//Furosemide	Furosemide started
MEDICATION//Heparin//Stopped	Heparin stopped
Infusions	
INFUSION_START//221794	Infusion of furosemide (Lasix) started
INFUSION_END//225166	Infusion of potassium chloride ended
INFUSION_START//222168	Infusion of propofol started
Procedures	
PROCEDURE//START//227194	Removal of endotracheal tube started
PROCEDURE//END//225459	Plain chest X-ray ended
PROCEDURE//START//224263	Central venous cannula insertion started
Laboratory Values	
LAB//220621//mg/dL	Glucose [Mass/volume] in Serum or Plasma (mg/dL)
LAB//220615//mg/dL	Creatinine [Mass/volume] in Serum or Plasma (mg/dL)
LAB//220050//mmHg	Systolic blood pressure (mmHg)
Administrative Events	
HOSPITAL_ADMISSION//URGENT//PHYSICIAN REFERRAL	Admitted urgently via physician referral
ICU_ADMISSION//Medical Intensive Care Unit (MICU)	Admitted to Medical Intensive Care Unit (MICU)
TRANSFER_TO//transfer//Medicine	Transferred to Medicine