Optimizing Feature Extraction for On-device Model Inference with User Behavior Sequences

Chen Gong, Zhenzhe Zheng, Yiliu Chen, Sheng Wang, Fan Wu, Guihai Chen

Shanghai Jiao Tong University & **ByteDance Client AI**

2026-05-12

ACM/IEEE SenSys 2026

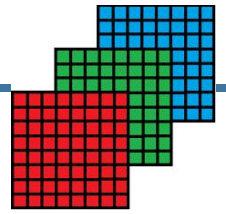
The Rise of Mobile ML

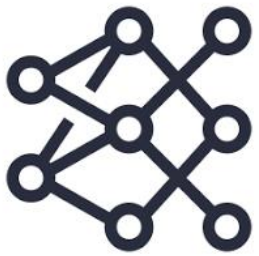
ML models are deeply integrated into modern mobile apps

Dynamic vs. Static Features

Traditional CV/NLP models: static input features

Image Pixels





Token Embeddings

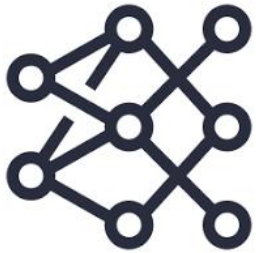
Token String	Token ID	Embedded Vector
"The"	101	[0.5, 0.8, ..., 0.65]
"cat"	2598	[0.1, 0.3, ..., -0.82]
"sat"	4450	[0.41, -0.53, ..., 0.8]
"."	119	[0.17, 0.75, ..., 0.2]

On-device ML models: dynamic user features

T-2: Search

T-1: Watch

T: Favorite

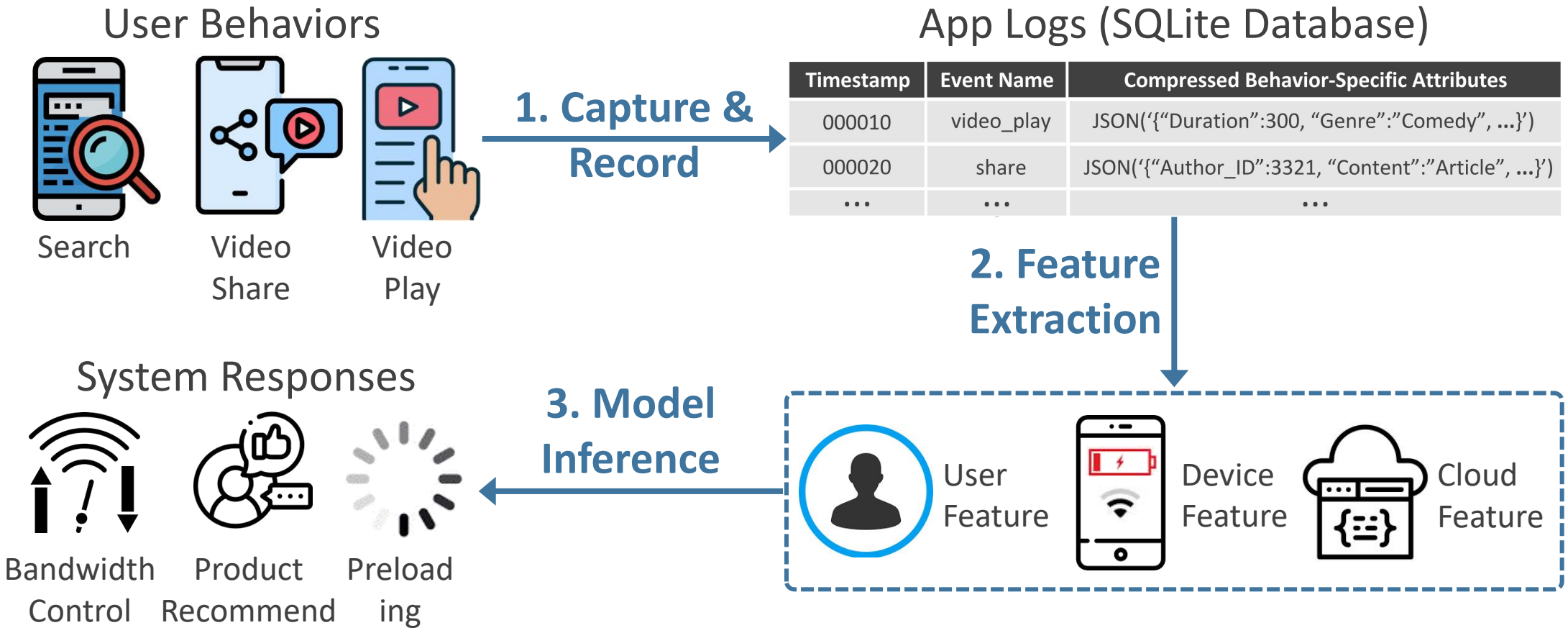


T+1: ?

Evolving User Behaviors

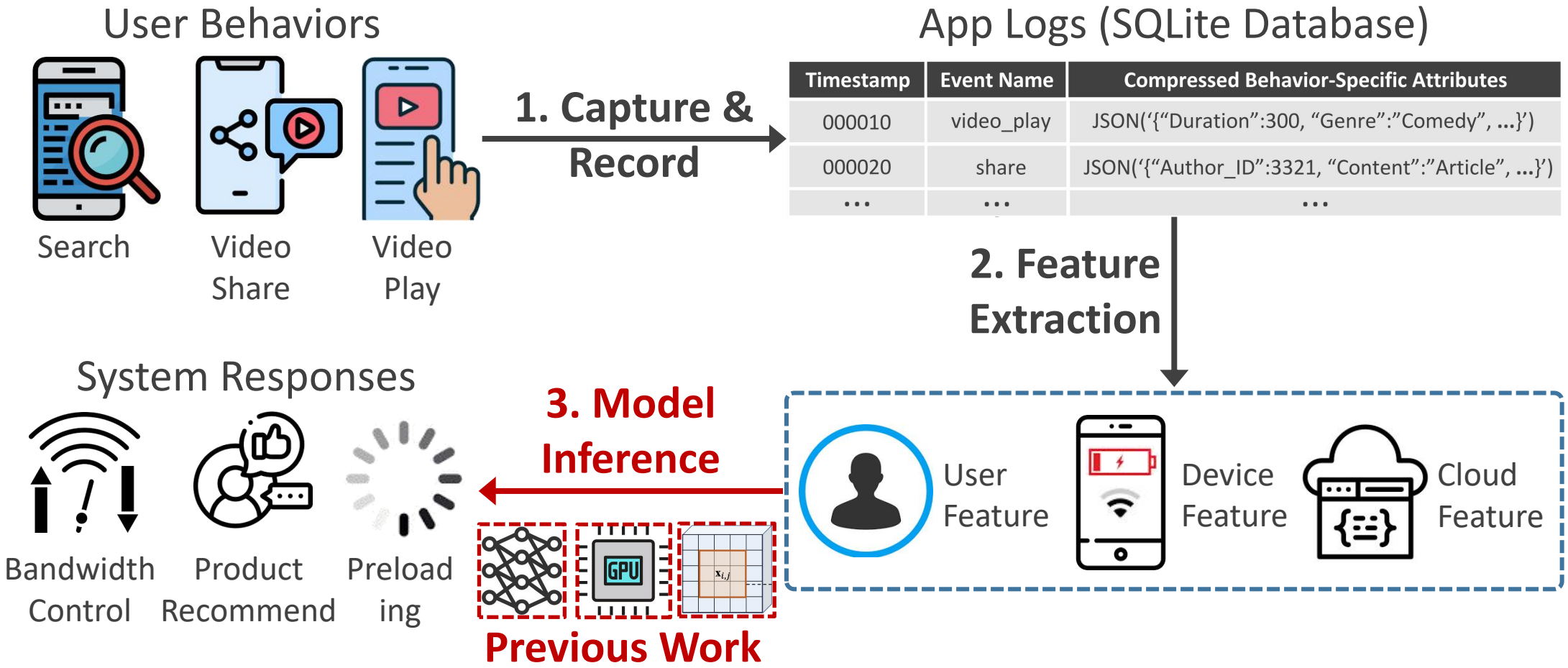
On-Device Workflow

User Behavior → App Logs → Features → App Response



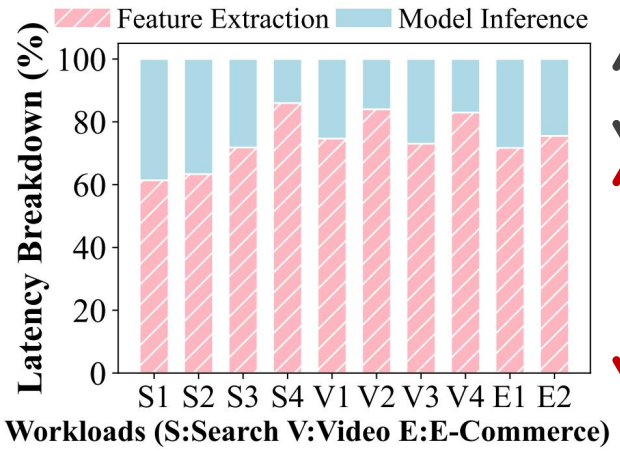
Focus of Prior work

Optimize model inference stage.



Overlooked Bottleneck

Feature extraction accounts for 61% - 86% of execution time



≤ 39%: Inference

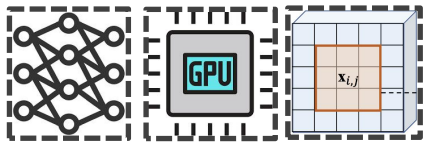
≥ 61%: Feature Extraction

App Logs (SQLite Database)

Timestamp	Event Name	Compressed Behavior-Specific Attributes
000010	video_play	JSON('{"Duration":300, "Genre":"Comedy", ...}')
000020	share	JSON('{"Author_ID":3321, "Content":"Article", ...}')
...	...	...

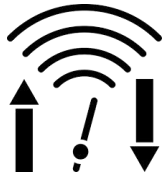
2. Feature Extraction

3. Model Inference



Previous Work

System Responses



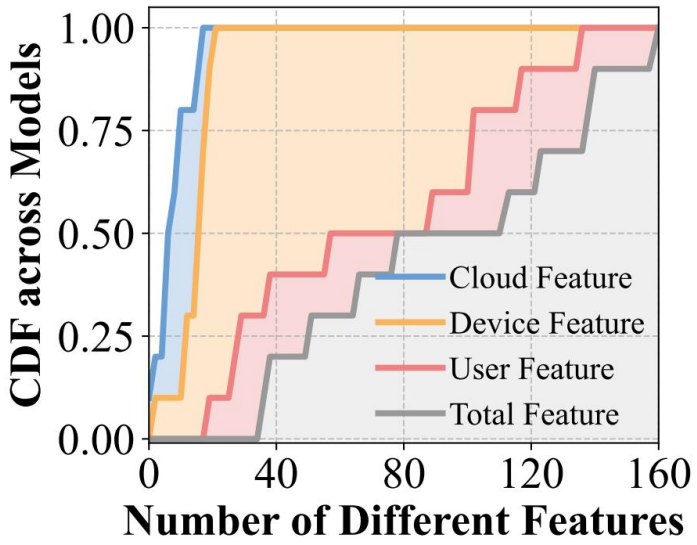
Bandwidth Control

Product Recommend

Preload ing

Why is extraction so slow?

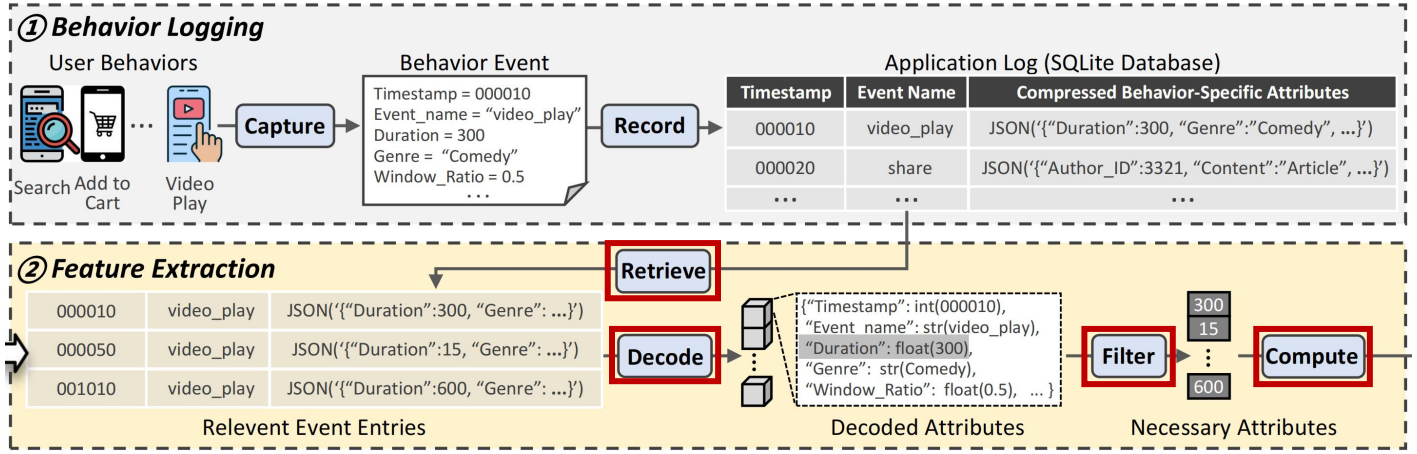
1. High proportion of user features



On average, **74%** of input features are user features.

2. Cumbersome feature extraction

- Retrieve, Decode, Filter, Compute

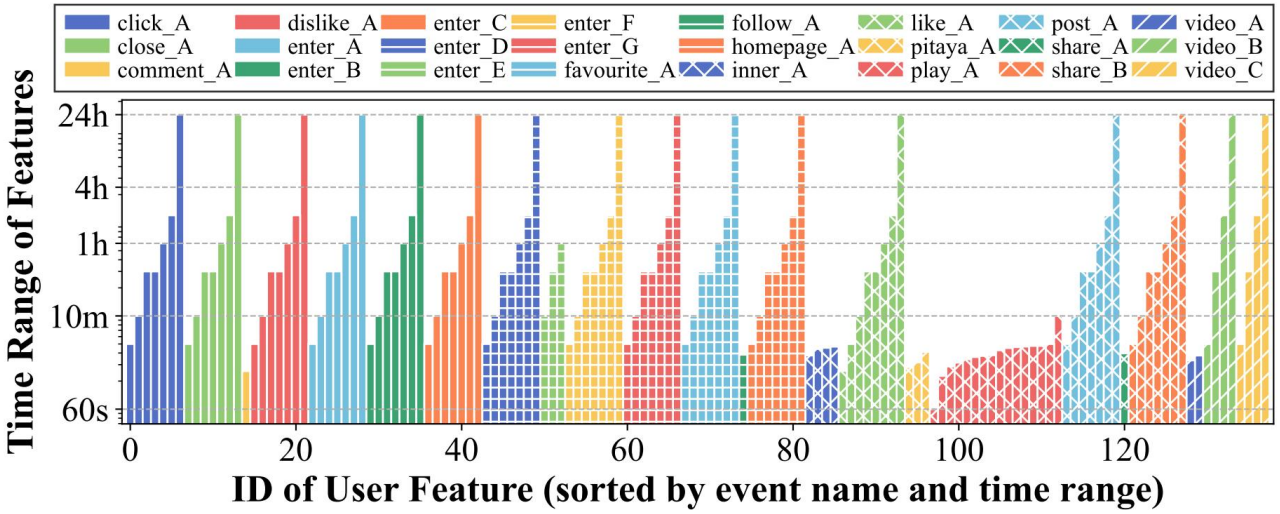


3. Fast inference: ms-level latency

- Model size: MB-level
- Mature optimization

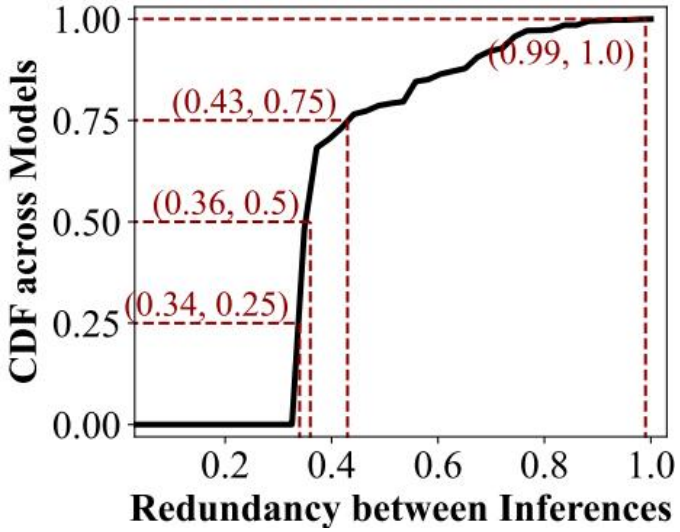
Optimization Opportunity

Key Observation: Many user features are extracted from overlapping user behavior data in app logs.



Inter-Feature Redundancy

- 134 features extracted from **24** distinct **behavior types** with different time ranges

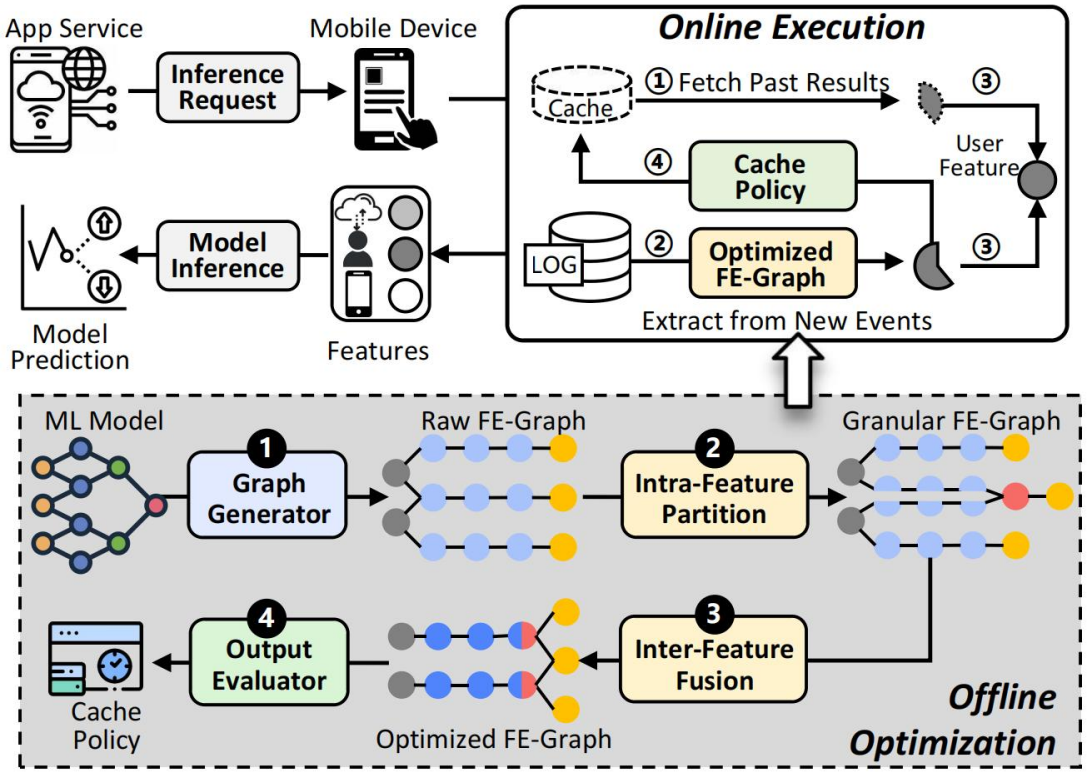


Cross-Inference Redundancy

- 75% online model **inferences** have **≥ 43% overlapping data**

AutoFeature Overview

Accelerate on-device feature extraction without compromising model inference accuracy.



Main Challenges:

#1: How to systematically **identify** redundancy?

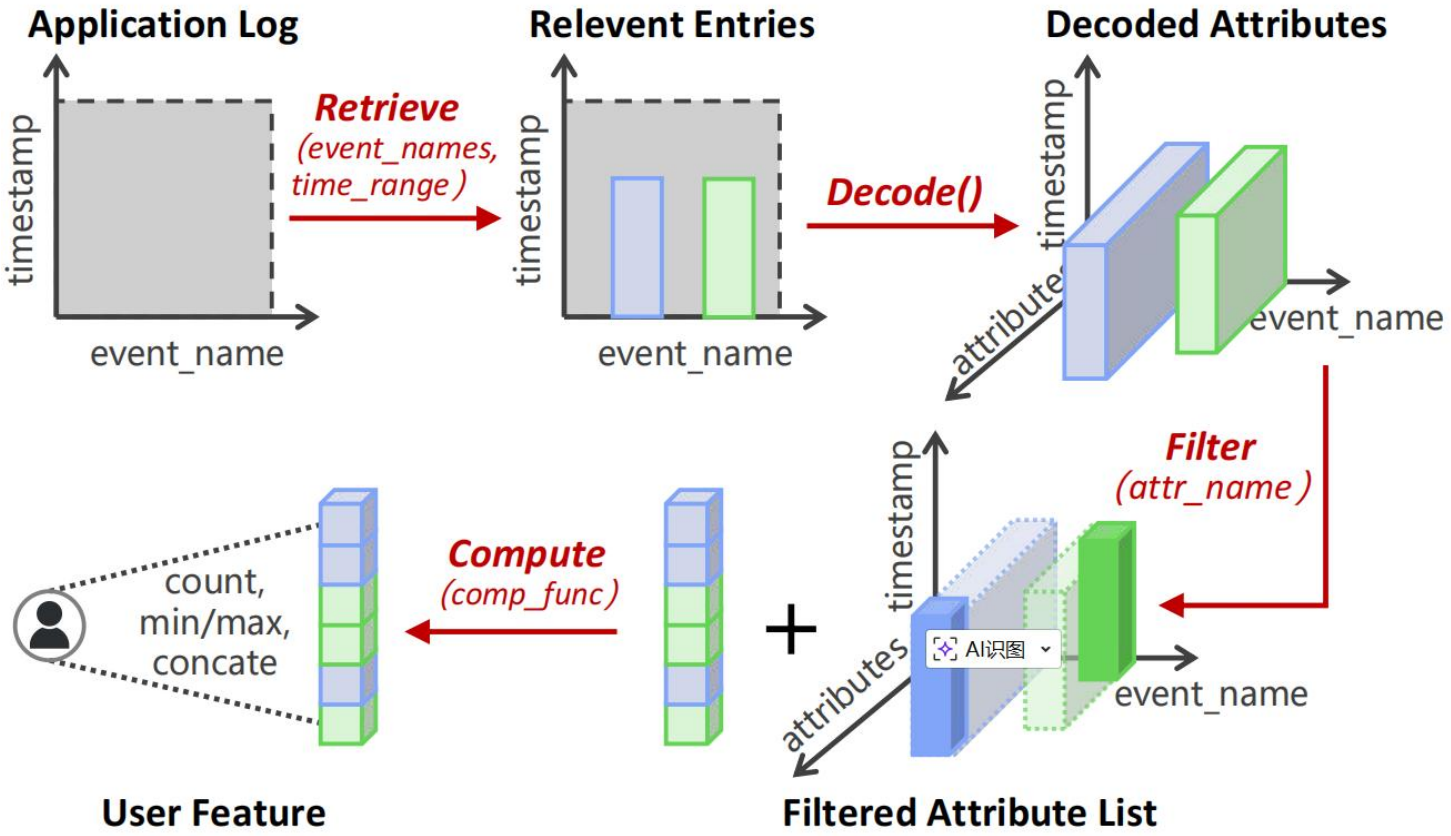
2: How to effectively eliminate **inter-feature** redundancy?

3: How to efficiently minimize **cross-inference** redundancy?

Solution 1 - Graph Generator

How to systematically identify redundancy?

- Model feature extraction workflow as a DAG



Source Node: App Log Data

Target Node: Each Feature

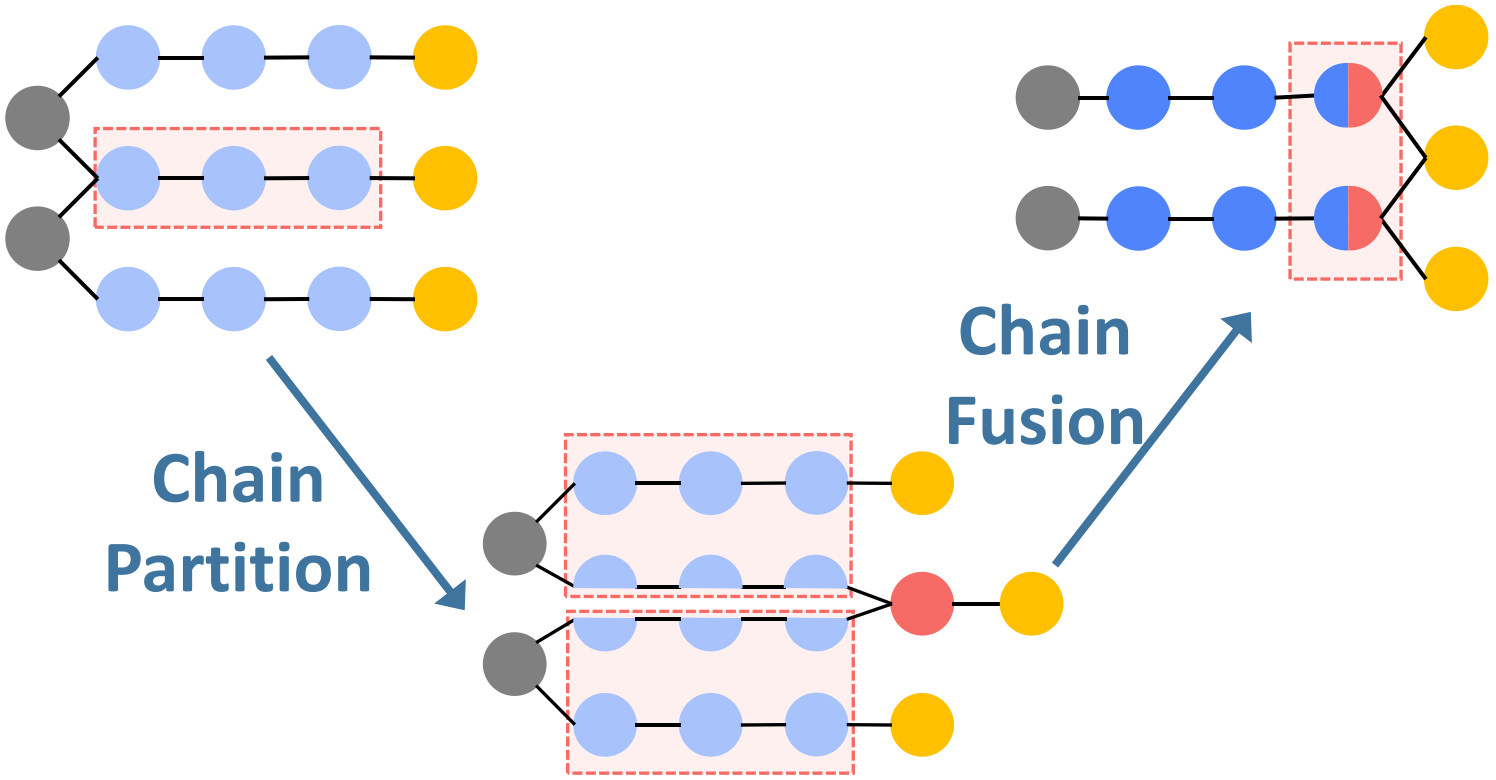
Connection: 4 atomic operation nodes

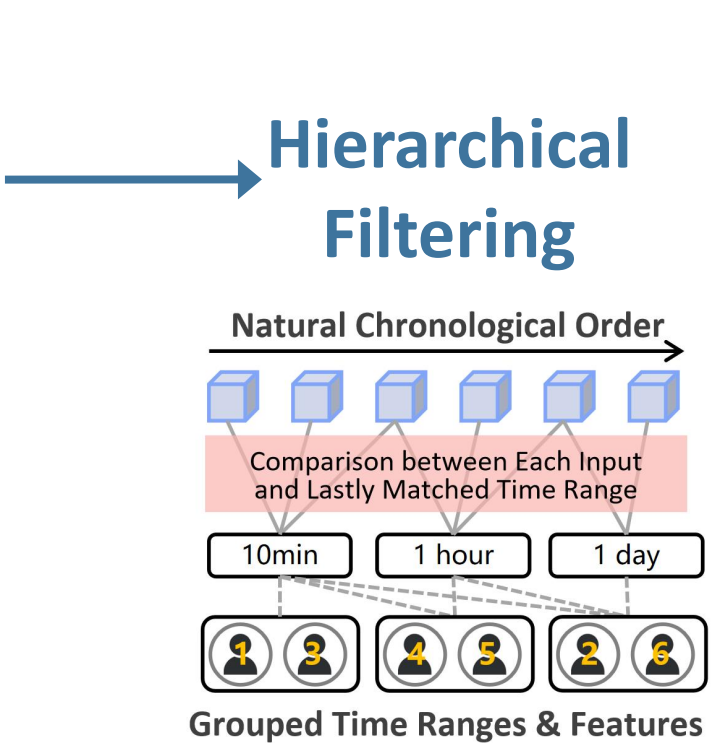
Quantify all redundancy via condition intersection !

Solution 2 - Graph Optimizer

How to effectively eliminate inter-feature redundancy?

- How to fuse? Chain Partition & Fusion
- How to separate? Hierarchical Filtering





Solution 3 - Efficient Caching

How to efficiently minimize inter-inference redundancy?

- Formulate as a knapsack packing problem

Target: Maximize savings

$$\mathcal{P}^* = \arg\max_{\mathcal{P}_i \in \{0,1\}} \sum_{i=1}^{N} \left[\mathcal{P}_i \times U(E_i)\right]$$

$$\text{s.t.,} \sum_{i=1}^{N} \left[\mathcal{P}_i \times C(E_i)\right] \leq M,$$

Constraint: device memory

Greedy Policy:

Based on saving-to-cost ratio

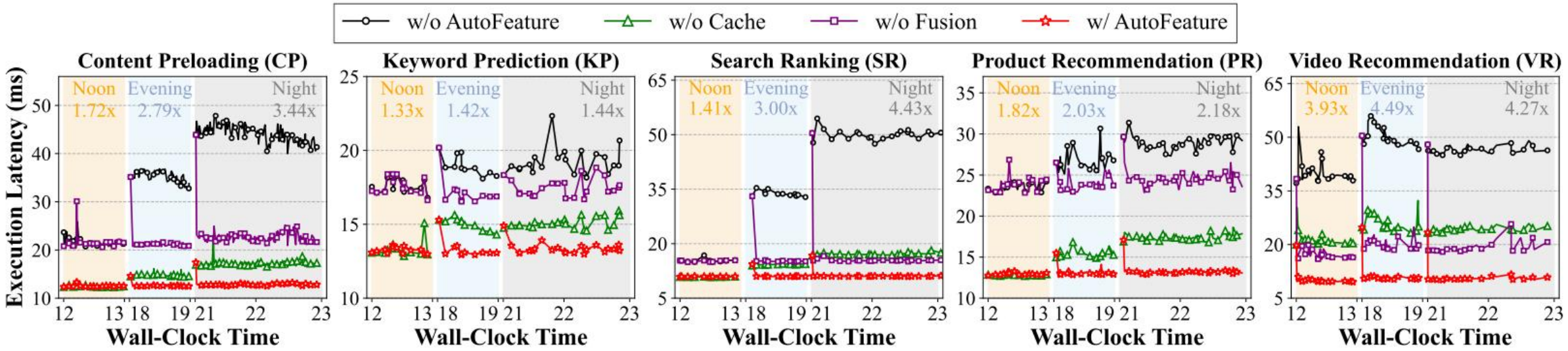
$$\frac{U(E_i)}{C(E_i)} \overset{(a)}{=} \frac{Time\_Overlap(E_i) \times Freq(E_i) \times Cost\_Opt(E_i)}{Time\_Range(E_i) \times Freq(E_i) \times Size(E_i)}$$

$$= \underbrace{\frac{Time\_Overlap(E_i)}{Time\_Range(E_i)}}_{\text{Dynamic Term: O(1) Time}} \times \underbrace{\frac{Cost\_Opt(E_i)}{Size(E_i)}}_{\text{Fixed Term: Precompute}},$$

Evaluation & Results

5 Industrial Services & Real-world Users

- Consistently reduce model execution latency to **≤20 ms**
- **Daytime: 1.33-3.93x** speedup, **Night: 1.44-4.43x** speedup



Content Preloading
1.72-3.44x Speedup

Keyword Prediction
1.33-1.44x Speedup

Search Ranking
1.41-4.53x Speedup

Product Recommend
1.82-2.18x Speedup

Video Recommend
3.93-4.43x Speedup

Coclusion

Problem:

Unveil feature extraction bottleneck in on-device model inference

Solution:

AutoFeature optimizes feature extraction without compromising model inference accuracy

Result:

Remarkable speedups in online evaluation of industrial services.

Thank You for Your Attention !

Chen Gong
gongchen@sjtu.edu.cn