Mid-Project Report

Adaptive Multi-Resolution Count-Min Sketch for Real-Time Stream Analytics

Saj Patel
Saurav Gupta

Rice University

COMP 580 — Probabilistic Algorithms

Fall 2025

Submitted: November 17, 2025

Abstract

Real-time streaming systems thrive on accurate frequency estimation during critical time intervals. However, traditional Count-Min Sketch (CMS) structures fall short when it comes to sliding-window and time-bounded queries. Many existing methods struggle with per-item deletions, which can lead to inaccuracies, especially at high data rates. Excitingly, recent research has introduced innovative multi-resolution and decay-based sketching techniques that enhance time awareness [1, 2, 4], paving the way for more effective solutions.

We're developing a framework for adaptive, time-bounded frequency tracking in real-time data streams without needing deletions. Our approach includes: (1) a ring-buffer Count-Min Sketch for easy time partitioning, (2) Hokusai's multi-resolution hierarchy for quick queries [1], (3) adaptive resizing and time-decay features inspired by Ada-Sketches and Forward Decay [2, 4], and (4) BurstSketch methods for detecting frequency spikes [3]. This implementation has been developed in Python and tested using synthetic data, with a focus on accuracy, memory efficiency, temporal stability, and responsiveness. Our goal is to deliver a practical solution that eliminates the need for deletion in real-time analytics.

1 Introduction and Problem Description

Real-time stream analytics is crucial for managing frequency-related queries over short periods. It helps identify trends, measure immediate actions, and detect surges in high-velocity data streams. While classical algorithms, such as Count-Min Sketch (CMS), efficiently estimate frequencies, they focus on append-only streams and overlook time constraints. In areas like telemetry and fraud monitoring, users need counts for specific time frames, such as the last 5 minutes. Unfortunately, CMS can't handle these time-sensitive queries effectively because its counters accumulate indefinitely, making it hard to remove outdated data.

To enhance time awareness, removing expired items one by one is a practical approach. However, the Count-Min Sketch (CMS) technique complicates this due to shared counters from hash collisions, potentially leading to inaccuracies when deleting a single key. Accurate deletions require tracking timestamps for each key and additional structures to identify affected buckets, which increases time complexity and memory requirements.

A practical alternative is to use time-partitioned sketches, such as a ring buffer with per-minute CMS instances. Queries sum the relevant sketches while discarding outdated ones. This avoids deletions but struggles with large windows, many merges, and non-stationary workloads. Multi-resolution hierarchies like Hokusai [1] address some of these issues by aggregating sketches at different time scales.

Our project aims to create a deletion-free, multi-resolution sketching framework for real-time frequency estimation. We include techniques from Hokusai's time hierarchies [1], adaptive resizing and time decay from Ada-Sketches and Forward Decay [2, 4], and burst detection from BurstSketch [3]. The goal is to achieve accurate and memory-efficient frequency tracking in dynamic environments.

2 Literature Review

The Count-Min Sketch (CMS) is great for estimating frequencies efficiently, but struggles with time frame-specific queries. Hokusai addresses this by providing a multi-resolution hierarchy of sketches for different time spans [1]. This approach enables time-window queries with simple sketch merges, eliminating the need for per-item deletions. Although there are challenges associated with increased errors and adapting to changing workloads, Hokusai enhances efficient data analysis.

Adaptive methods for managing streaming data include Ada-Sketches, which use a resizing strategy and exponential time weighting to minimize error fluctuations [2]. Forward Decay provides a continuous-time decay model that focuses on recent data without requiring deletions [4]. While these approaches enhance stability, there is still room for improvement in handling multi-resolution queries.

BurstSketch identifies short, high-frequency spikes in real-time data streams [3]. It uses lightweight estimates to track potential bursts and validates them through snapshot comparisons. While not designed for sliding-window analytics, it effectively detects transient events often missed by traditional CMS structures.

Together, these works highlight several directions for time-aware frequency estimation, including multi-resolution sketching, adaptive decay, continuous-time weighting, and burst detection. Our project integrates these ideas into an adaptive, deletion-free framework for real-time stream analytics.

3 Hypotheses

We hypothesize that a multi-resolution Count-Min Sketch structure can support accurate time-bounded queries without requiring per-item deletions [1].

We hypothesize that adaptive resizing and continuous time decay will reduce error variability under changing workload conditions compared to a fixed-size CMS [2, 4].

We hypothesize that integrating a burst detection mechanism will identify short high-frequency spikes more effectively than standard CMS-based estimates [3].

4 Experimental Settings

4.1 Ring Buffer Count-Min Sketch Baseline

We ingest the full month of Reddit May 2019 n-grams, select the top five, bottom five, and an 800-item random pool, and stream them through a 24-slot Count-Min Sketch ring buffer with one-hour slots. We record burst scores every 10 minutes. For ground truth, we track each target with an exact deque-based counter. We compare CMS estimates to exact counts through error distributions and plots. We visualize natural high-frequency phrases, rare phrases, and random phrases that exhibit bursty behavior. We also evaluate CMS parameter sensitivity by varying sketch width

and depth to observe memory-error tradeoffs. Planned next steps include extending the exact vs. CMS comparison across multiple days for stability analysis, exploring adaptive widening for heavy hitters, testing longer window definitions, and benchmarking alternatives such as Count Sketch or Space Saving.

4.2 Hokusai Multi-Resolution Sketch

For the Hokusai model, we construct sketches at time resolutions of 1 hour, 2 hours, 4 hours, and 8 hours [1]. Each level aggregates the corresponding span of the stream and supports time-window queries through merges of a small number of sketches. We use the same Reddit n-gram stream and the same set of tracked targets. We compare estimated window counts to exact per-target deques for multiple window sizes. We evaluate the accuracy impact of sketch folding at higher levels and the query cost relative to the ring-buffer baseline. Planned extensions include testing different resolution hierarchies and quantifying merge-induced error.

4.3 Adaptive Sketch with Time Decay

For the adaptive model, we implement CMS variants that apply exponential decay and allow width adjustments based on observed key frequencies. These techniques are inspired by Ada-Sketches and Forward Decay [2, 4]. We track counts using both decayed estimates and exact sliding-window counts. Planned work includes evaluating decay parameters, testing adaptive widening for heavy hitters, and comparing decay schedules.

4.4 BurstSketch-Inspired Burst Detector

For burst detection, we implement a lightweight two-phase mechanism inspired by BurstSketch [3]. We compare CMS-based burst scores to exact deque-based scores and evaluate detection latency, false positives, and false negatives for top, rare, and random phrase groups. Planned extensions include multi-day testing, alternative scoring formulas, and comparison to simple fixed-threshold baselines.

5 Current Experimental Status

The ring-buffer Count-Min Sketch baseline is fully implemented, using a 24-slot structure with one-hour buckets to process the Reddit May 2019 n-gram stream in chunks. We track the top five items, the bottom five, and maintain an 800-item random pool. Exact sliding-window counts for each target are computed using target-specific deques, serving as ground truth for evaluating Count-Min Sketch estimates. Every 10 minutes, we log burst scores and compare Count-Min Sketch burst signals to exact signals by adjusting sketch width and depth. Several plots, including time-series burst curves and error distributions, have been generated, all produced in Python, ensuring

stable and repeatable visualizations. The current implementation focuses on burst scores, but raw counts and deltas can be added for deeper analysis.

The remaining algorithms, including the Hokusai hierarchy [1], adaptive sketch [2, 4], and BurstSketch-inspired detector [3], are not yet implemented. The end-to-end pipeline is still being expanded to support these models.

6 Conclusion and Future Work

The current stage of the project has established a reliable baseline for time-bounded frequency analysis using a ring-buffer Count-Min Sketch and deque-based ground truth. We have validated the ingestion pipeline, implemented evaluation tools, and created visualizations that distinguish natural, rare, and bursty phrases. These results confirm that the ring-buffer approach offers a strong foundation for advanced models.

Next steps involve implementing the Hokusai hierarchy [1], adaptive decayed sketches [2, 4], and a BurstSketch-inspired detector [3]. We will integrate these models, automate parameter sweeps, extend evaluations across multi-day windows, and compare performance with Count Sketch and SpaceSaving.

References

[1] Matusevych, S., Smola, A. J., and Ahmed, A. (2012). *Hokusai: Sketching Streams in Real Time.* Available at: https://www.auai.org/uai2012/papers/231.pdf

[2] Shrivastava, A., Konig, A. C., and Bilenko, M. (2016). *Time Adaptive Sketches (Ada-Sketches) for Summarizing Data Streams.* Proceedings of SIGMOD 2016. DOI: https://doi.org/10.1145/2882903.2882946

[3] Zhong, Z., Yan, S., Li, Z., Tan, D., Yang, T., and Cui, B. (2021). *BurstSketch: Finding Bursts in Data Streams.* Available at: https://yangtonghome.github.io/uploads/BurstSketch.pdf

[4] Cormode, G., Shkapenyuk, V., Srivastava, D., and Xu, B. (2009). *Forward Decay: A Practical Time Decay Model for Streaming Systems.* Available at: http://dimacs.rutgers.edu/ graham/pubs/papers/fwddecay.pdf