Title	Methodology	Study focus	Finding	Challenges	Tools & Technology use
1.Concept Drift Detection (Rodrigo F. de Mello)	Survey + algorithmic techniques	Detecting distribution shift in streams	Adaptive drift detectors maintain model accuracy as data evolves	Timely detection vs false alarms; labeling scarcity	Drift detection tests; incremental learners; sliding windows
2.Real-Time Clickstream Anomaly Detection with Amazon Kinesis (AWS, 2016)	Reference architecture & case study	Online anomaly detection on user events	Managed streaming + serverless enables scalable, low-ops pipelines	Throughput bursts; cost vs latency; model retraining	Amazon Kinesis, Lambda, DynamoDB, CloudWatch
3.Real-Time Time-Series Anomaly Detection for Streaming Applications (AWS, 2023)	System blueprint	Time-series modeling at ingestion	Feature engineering + ML on streams improves precision/recall	Handling seasonality; multi-metric correlation	Kinesis Data Analytics, SageMaker, Glue
4.Streaming Analytics: Top 5 Use Cases, Challenges, and Best	Industry whitepaper	Business use-cases & patterns	Clear wins in fraud, IoT, personalization with managed stacks	Data quality; exactly-once semantics; ops complexity	Apache Kafka, Apache Flink/KSQL, managed Kafka

Practices (Instaclustr)					
5.Anomaly Detection in Azure Stream Analytics (Microsoft Learn, 2024)	Product docs + tutorials	Built-in anomaly operators on streams	Ready-to-use “Spike”/“ChangePoint” UDFs accelerate delivery	Parameter tuning; cold-start; scale-out	Azure Stream Analytics, Cosmos DB, Power BI
6.Faster, Scalable IoT Anomaly Detection of Real-Time Data Streams (Ayla Networks, 2022)	Engineering blog + case	Edge/IoT telemetry monitoring	Feature thinning + streaming scoring reduces lag	Noisy sensors; constrained devices; bandwidth	MQTT/IoT hubs, stream processors, alerting
7.Real-Time Anomaly Detection for Streaming Analytics (arXiv, 2016)	Research preprint	Algorithms for online anomaly scoring	Sketches + incremental stats enable low-latency detection	High-dimensional data; evolving baselines	Probabilistic sketches; windowed stats
8.Review of Anomaly Detection Algorithms for Data Streams (MDPI, 2023)	Systematic literature review	Taxonomy of streaming AD methods	Hybrid/statistical-learning combos outperform single methods	Concept drift; class imbalance; evaluation protocols	Sliding windows, ADWIN, Hoeffding trees

9.Real-Time Data & Analytics for Financial Services (Kinetica)	Vendor whitepaper	Streaming analytics in finance	GPU-accelerated DBs lower latency for risk/fraud	Governance; latency SLAs; cost	Kinetica DB, Kafka, connectors
10.How Real-Time Anomaly Detection Works in Streaming Analytics (Accion Labs, 2023)	Engineering explainer	Pipeline stages for AD	Data enrichment + feature stores are critical for accuracy	Feature freshness; model deployment	Kafka, Flink/Spark, feature store
11.How to Detect Anomalies Using Streaming Analytics & AI (YouTube, 2020)	Tutorial/demo	Practical AD workflow	Simple baselines + thresholds can work with good monitoring	Visualization; alert fatigue	Python notebooks, streaming dashboards
12.What Is Streaming Analytics? (Twilio Segment)	Intro guide	Concepts & benefits	Event pipelines unify product/marketing analytics	Identity resolution; schema drift	Segment pipelines, warehouses, Kafka
13.What Is Streaming Analytics: Data Streaming & Stream Processing	Technical overview	Architectures & tools	Lambda/Kappa patterns frame design trade-offs	Exactly-once; late/out-of-order data	Kafka, Spark/Flink, CDC tools

(AltexSoft, 2020)					
14.What Is Streaming Analytics (Google Cloud)	Product overview	GCP services for streams	Managed cloud services speed time-to-value	Vendor lock-in; quota planning	Pub/Sub, Dataflow (Apache Beam), BigQuery
15.Detect Anomalies in Streaming Data Using Python & ML (Redpanda, 2024)	Code tutorial	Building an AD service with a log broker	Brokers + Python microservices give simple, portable stacks	Back-pressure; serialization overhead	Redpanda (Kafka-API), Python, scikit-learn
16.Real-Time Streaming Analytics with AWS Kinesis (Ascentt)	Solution brief	End-to-end Kinesis analytics	Reference patterns for ingestion → processing → visualization	Partition key design; hot shards	Kinesis Data Streams, KDA, Athena, QuickSight
17.Streaming Analytics: Intro, Tools & Use Cases (Confluent)	Vendor guide	Kafka-centric streaming	Event-driven design improves decoupling & scalability	Ordering, idempotency, schema evolution	Apache Kafka, Schema Registry, ksqlDB
18.Efficient Real-Time Data Stream Anomaly Detection (ReadyTensor, 2025)	Blog/benchmark	Comparing AD approaches	Lightweight models can match deep models for many streams	Reproducible eval; dataset scarcity	Python AD libraries; benchmark harness

19.Exam AWS Certified Data Analytics —Specialty (Q84) (ExamTopics, 2021)	Exam scenario analysis	Best practices for Kinesis analytics	Partitioning, scaling, and consumption patterns are key	Choosing between Streams vs Firehose; consumers	Kinesis, Lambda, EMR, S3
20.What Is Streaming Analytics: Platform for Real-Time & IoT (Software AG)	Product brief	End-to-end streaming platform	CEP + ML supports complex, low-latency use-cases	Rule tuning; integration	Software AG Apama/Streaming Analytics, MQTT

Literature Review :(Rashed)

Title	Methodology	Study focus	Finding	Challenges	Tools	Technology use
1.Who Needs Exact Answers Anyway? The Joy of Approximate Big Data	Sampling techniques(simple random sampling, stratified sampling, and reservoir sampling); probabilistic data structures(histograms, wavelets, and quantile summaries); Python experiments	Demonstrating efficiency and scalability of approximate techniques in big data	~1.46% sampling error; Bloom/Cuckoo etc performance metrics; close cardinality with HyperLogLog	Parameter tuning, accuracy-speed trade-offs, managing false positives	Python, pandas, numpy, bitarray, mmh3, hyperloglog, BCuckooFilter	Python-based implementations, statistical validation, visual comparisons

2.Streams and Sketches	Sketch-based approximate algorithms for one-pass, memory-constrained streams using randomness and concentration bounds	Designing efficient algorithms for streaming tasks: counting, distinct elements, heavy hitters, etc.	Approximations can be accurate, fast, mergeable, and memory-efficient; HyperLogLog is a standout example	One-pass limitations, variance, hash dependency, error-memory tradeoffs, distributed consistency	Sketches (Init/Update/Query), concentration inequalities, pairwise-independent hash families	Algorithmic frameworks: compact sketches, probabilistic analysis, mergeable structures
3.Streaming Model of Computation	Streaming algorithms with polylog memory, O(1) per-item updates, single-pass, approximate solutions	Comparing streaming algorithms with DFAs; computing stream functions (moments, frequency) using sketches & samples	Streaming can simulate DFA, sample uniformly, estimate frequencies/moments with sketches	Severe memory/resource constraints, approximation accuracy, language recognition limits	Reservoir Sampling,Count-Min Sketch, AMS Sketch, sliding windows, histograms, wavelets, hash families	Theoretical algorithmic constructs: randomness, synopses, probabilistic guarantees
4.A Survey of Synopsis Construction in Data Streams	Surveys stream-friendly synopsis structures (sampling, wavelets, sketches, histograms)	Summarization techniques for query estimation, aggregation, joins, mining	Overview of key methods: reservoir/concise sampling, wavelets, Count-Min, sketches, histograms	Comparative evaluation, support for complex data domains, adaptability to evolving streams	Reservoir sampling, concise sampling, wavelet decomposition, Count-Min, sketches, histograms	Algorithmic, probabilistic, one-pass, real-time summarization under constraints
5.Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches	Surveys four synopsis families for approximate query processing: random samples, histograms, wavelets, sketches	Comparative analysis of synopsis methods, their mathematical properties, and use in AQP systems	Sampling is simple & broad; histograms are classic; wavelets good for patterns; sketches excel in streaming	Trade-offs among accuracy, space, dynamism; selection complexity depending on workload/data	Sampling techniques, histogram constructs, wavelet transforms, sketch algorithms	Algorithomic strategies for synopsis construction, AQP integration, scalable streaming/distributed processing
6.Probabilistic data structures for big data analytics: A comprehensive review	Hash-based randomized algorithms to summarize massive data with compact memory	Membership, frequency, cardinality, similarity queries in large/streaming data	PDS achieve high efficiency and low memory use; structures like Quotient Filter add versatility	Balancing size vs. accuracy, ensuring low error, adapting to changing data	Bloom/Cuckoo/Quotient filters; Count-Min Sketch; HyperLogLog variants; LSH/MinHash/SimHash	Real-time, streaming systems, databases, analytics, smart city applications

7.Data Stream (LUMS CS5312 lecture by Imdad Ullah Khan)	Single-pass, polylog space, constant-time per-item streaming model with probabilistic approximations	Streaming vs. traditional processing; real-world applications (sensors, networks, web, finance)	Approximate summaries enable efficient analytics via structures like Count-Min, Count, and AMS sketches	Unbounded data, memory/time constraints, no random access, real-time demands, adversarial input	Sliding window, histograms, wavelets, Count-Min/Count/ AMS sketches	Streaming-optimized, real-time, scalable, space-efficient, parallelizable tools
8.In the Land of Data Streams where Synopses are Missing	Introduces Condor, a framework integrating synopses as windowed aggregates for scalable stream processing.	Improve real-time analytics by supporting multiple synopses and simplifying development.	Up to 75× faster, scalablehandles diverse synopses efficiently, simplifies implementation.	Diversity of synopses, performance optimization, scalability, user abstraction.	Condor framework with declarative synopsis integration.	Works with Flink, Spark, Storm; supports real-time, scalable, memory-efficient streaming analytics.
9.A Quick Introduction to Data Stream Algorithmics	Introduces streaming algorithms for one-pass, memory-efficient approximations of statistics over large or infinite data streams.	Techniques for processing unbounded data streams: frequency counts, quantiles, heavy hitters, approximate aggregates.	Streaming algorithms enable fast, approximate computations with small memory, supporting real-time analytics.	Handling unbounded, high-speed data; maintaining accuracy with limited memory; balancing error and space efficiency.	Sketches, histograms, sampling, Count-Min Sketch, quantile summaries.	Applicable to network monitoring, web clickstreams, sensor networks, financial data processing.
10.Quantiles and Equi-depth Histograms over Streams	Algorithms for maintaining approximate quantile summaries and equi-depth histograms in a single pass over data streams using limited memory; includes deterministic and probabilistic approaches.	Efficiently compute quantiles and balanced histograms over unbounded data streams for data summarization and analysis.	Quantile summaries and equi-depth histograms can be maintained accurately with small memory; suitable for real-time streaming data analysis.	Operating within limited memory, single-pass constraints, and maintaining accuracy.	Deterministic and probabilistic algorithms for stream summarization.	Real-time analytics in network monitoring, sensor networks, and financial systems.

11.Data Stream Processing	Single-pass, memory-efficient algorithms with low per-record processing time; provides approximate answers with error bounds for real-time stream analysis.	Efficient processing of continuous data streams for real-time insights, trend analysis, and anomaly detection.	Stream processing models handle large-scale data efficiently; approximate algorithms give near-exact answers using less memory and time; suitable for real-time analytics.	Balancing accuracy and resources, scalability for high-volume streams, handling concept drift and evolving data distributions.	Apache Storm, Apache Kafka for stream computation and event streaming.	Real-time analytics for IoT, financial fraud detection, network monitoring, and recommendation systems.
12.Sampling for Big Data	Uses streaming, graph-based, and approximate sampling techniques to extract representative subsets of data efficiently.	Reduce computational overhead while preserving data representativeness for large-scale analytics.	Sampling reduces time/resources, retains dataset characteristics, and adaptive methods improve accuracy over time.	Ensuring representativeness, scalability for large datasets, and adapting to dynamic/streaming data.	Reservoir sampling, random walk-based sampling, importance sampling.	Applied in social network analysis, recommendation systems, and real-time analytics.
13.Algorithms for Data Streams	Presents single-pass, sublinear-memory algorithms for processing large data streams; includes approximation techniques, probabilistic counting, and sliding-window models.	Efficient computation on massive, continuous data streams where storing all data is infeasible.	Approximate algorithms can achieve high accuracy while using drastically less memory and processing time.	Handling unbounded streams, ensuring accuracy under memory limits, dealing with evolving and bursty data.	Count-Min Sketch, Bloom Filters, Reservoir Sampling, Frequency Moments estimation.	Network traffic monitoring, search engine query analysis, sensor network data aggregation, and web clickstream analysis.

14.Data Stream Algorithms for Vectors (Draft Chapter — "Algorithms for Signals")	Models high-dimensional signals as a vector x ∈ ℝⁿ, updated via a stream of (i□, Δ□) operations. Algorithms aim to approximate functions g(x)—like moments, distinct counts, or heavy hitters—using one-pass, polylogarithmic space, and probabilistic error bounds	Core streaming tasks in vector analysis: computing frequency moments (F□ = Σ	Core streaming tasks in vector analysis: computing frequency moments (F□ = Σ	Processing high-dimensional vectors via streaming updates, while coping with strict memory limits, ensuring single-pass computation, and maintaining approximation accuracy.	Misra–Gries algorithm for heavy hitters - Reservoir sampling for uniform sampling from unknown-length streams - AMS sampling for estimating frequency moments and entropy	Applicable to monitoring real-time high-speed signal contexts such as IP network flow tracking, sensor networks, transaction streams, and high-dimensional data monitoring, where full state retention is infeasible.
15.RTDS Unit-5: Advanced Concepts on Stream Computing	Introduces stream synopses—compact sketches, wavelets, and histograms—designed to summarize massive, continuous data streams efficiently, enabling approximate computation with minimal memory. Methods include Count-Min Sketch, HyperLogLog, wavelet transforms, and histograms (equi-width, equi-depth), plus sampling techniques like reservoir sampling.	Exploring synopsis construction and sampling strategies for stream mining—focusing on summarizing incoming data and enabling approximate analytics in real time.	These techniques allow efficient frequency estimation, distinct-item counting, signal compression, distribution approximation, and uniform sampling—all achieved without storing the full dataset.	Addressing infinite or very large data streams poses issues around limited memory, real-time processing, and maintaining acceptable approximation quality.	Count-Min Sketch for frequency estimation - HyperLogLog for cardinality estimation - Wavelet transforms for data compression - Histograms (equi-width and equi-depth) for distribution summarization - Reservoir Sampling for selecting representative samples from streams	These methods are applied in real-time analytics across domains like network monitoring, sensor data processing, clickstream analysis, anomaly detection, and query estimation over streaming data.

16.Box Queries over Multi-Dimensional Streams	Extends streaming algorithms to support box queries—range queries across multiple dimensions—over online, multi-dimensional data streams. Introduces three algorithms: C-DARQ, DARQ, and MARQ, designed for efficient counting, frequency estimation, heavy-hitter detection, and other statistical tasks.	Performing box queries (multi-attribute range queries) in a streaming context, including support for distributed scenarios where data is collected across multiple sites (e.g., network routers).	The proposed algorithms asymptotically improve space usage and update/query performance compared to prior approaches. They also generalize beyond counting to a broader range of statistical functions.	Processing multi-dimensional data in streaming fashion, achieving sublinear space, ensuring fast updates and queries, supporting distributed aggregation, and managing unknown or evolving dimensional bounds.	Three custom-designed algorithms tailored for box queries in streams: C-DARQ, DARQ, and MARQ	Applicable in distributed real-time analytics—such as network monitoring, spatial analytics, or high-dimensional time-series—where data arrives continuously from multiple sources and aggregated, fast range queries are required.
17.Learning based approaches to achieve better statistics in Big Data	Proposes regression-based techniques to generate statistical summaries (e.g., histograms) in distributed streaming systems. Focuses on balancing error metrics across workloads rather than minimizing only maximum error. Introduces new measures—Q-Regression pair (slope & intercept) and QRegrArea—to optimize overall estimation performance.	Improve statistical summary quality in big-data and decision support systems. Specifically aimed at systems where fast approximations (for query optimization, resource planning, etc.) are needed under constraints of memory and computation in streaming/distributed environments.	Identified that conventional methods minimizing maximum error may not yield best average performance. Regression-based summaries provide more consistent and smoother error distribution, reducing overall estimation error across workloads.	Handling extremely large-scale data volumes where memory, compute, and latency are limited; achieving accurate statistical summaries in highly distributed and streaming scenarios; balancing error across diverse query workloads.	Regression algorithms applied to summary generation; new error metrics (Q-Regression pair, QRegrArea) designed to guide optimization of summary accuracy over entire workload.	Relevant to real-world systems like query optimizers in RDBMS, distributed stream-processing, Industrial IoT, edge analytics, and autonomous systems, where efficient and accurate statistical estimation is critical.

18.Data Summaries for Scalable, High-Cardinality Analytics	Designs data summarization techniques tailored for high-cardinality aggregation workloads. Employs an end-to-end algorithm design approach, optimizing for accuracy, memory usage, and runtime in distributed, segment-based systems like Druid or Kylin	Supports efficient querying over data with many segments or high-dimensional groupings—a setting where naive mergeable summaries fail to scale. Emphasizes quantile queries, frequency estimation, and kernel density estimation	The thesis introduces three techniques—Moments Sketch for efficient quantile queries, Cooperative Summaries for improved aggregate accuracy, and TKDC for fast kernel density estimation—all outperforming existing methods in real systems.	Balancing scalability with accuracy and efficiency in real-time analytics over millions of segments; overcoming latency and memory bottlenecks inherent in standard mergeable summary techniques	Introduced novel summarization data structures: Moments Sketch, Cooperative Summaries, and TKDC tailored for high-cardinality aggregation contexts	Applicable to high-performance analytical systems like Apache Druid, Kylin, etc., to speed up segment-level queries by precomputing compact summaries that enable fast merging and computation over vast, segmented datasets
19.1.Data Stream Management: A Brave New World	Presents the data stream model for continuous, unbounded, high-volume data; introduces stream management systems (DSMS) with query operators adapted for real-time.	How to manage and process continuous data streams efficiently compared to traditional DBMS.	DSMS can support real-time monitoring and decision-making; approximate queries and synopses make large-scale streaming feasible.	Handling infinite streams, limited storage, query optimization, and timeliness of results.	Stream management systems, approximation methods, synopsis structures	Applications in sensor monitoring, financial tickers, network traffic analysis, and real-time applications.
19.2.Foundations and Basic Stream Synopses	Describes compact data structures (synopses) like sampling, histograms, wavelets, and sketches for one-pass approximations under limited memory.	Designing synopses that summarize streams to enable approximate query answering.	Different synopses suit different tasks: sampling is simple, histograms capture distribution, wavelets detect patterns, sketches excel at frequency/counting.	Achieving accuracy with sublinear memory; adapting to evolving streams	Sampling algorithms, histograms, wavelet transforms, sketches.	Used in real-time analytics, data compression, sensor aggregation, and query optimization.

19.3.Mining Data Streams	Applies incremental and approximate mining algorithms (clustering, classification, frequent items, and association rule mining) adapted to the stream setting.	Discovering knowledge and patterns in continuous, high-speed streams without storing all data.	Streaming mining algorithms enable tasks like frequent pattern detection, clustering, and classification in real time.	Limited memory, concept drift (changing patterns), high arrival rates, and balancing accuracy vs. efficiency.	Stream clustering algorithms, decision trees, frequent pattern mining techniques.	Applications in web clickstream analysis, network intrusion detection, real-time recommendations, and IoT analytics.
20.Synopses for Massive Data: Samples, Histograms, Wavelets, Sketches	Survey-based approach reviewing Approximate Query Processing (AQP) techniques: Samples, Histograms, Wavelets, Sketches, analyzing construction, usage, and trade-offs.	Focuses on compact, lossy data summaries (synopses) to enable fast approximate queries on massive datasets, including streaming data.	1.Each synopsis type has strengths and weaknesses. 2.Trade-offs exist between accuracy, space, and time. 3.Error bounds and incremental updates are critical for reliable results.	1Balancing accuracy with efficiency 2.Handling dynamic/streaming data 3.Controlling approximation errors 4.Choosing the right synopsis for each application	1.Sampling algorithms 2.Histogram construction methods 3.Wavelet transforms 4.Sketching algorithms	1.Efficient data structures 2.Streaming and approximation algorithms 3.Error metrics for evaluating query accuracy