The Bioscan
AN INTERNATIONAL QUARTERLY JOURNAL OF LIFE SCIENCES
20(4): 1334-1347, 2025 www.thebioscan.com

Active Fix Lite: A Time-Efficient Auto Encoder Based Non-FIFO Sliding Window Aggregation Approach for Real-Time Analytics

C. Kalyani [1], Dr. M. Safish Mary[2]
[1]Research Scholar, (Reg. No.20214542292030), PG & Research Department of Computer Science, St. Xavier's college (Autonomous), Palayamkottai, Manonmaniam Sundaranar University, Tirunelveli. India, Email: kals.sona@gmail.com
[2]PG & Research Department of Computer Science, St. Xavier's College (Autonomous), Palayamkottai, Manonmaniam Sundaranar University, Tirunelveli. India, Email: marysafish@gmail.com

DOI: 10.63001/tbs.2025.v20.i04.pp1334-1347

KEYWORDS
Sliding window aggregation, Non-FIFO streams, Auto encoder, Dimensionality reduction, Check points

Received on:

18-09-2025

Accepted on:

14-11-2025

Published on:

17-12-2025

ABSTRACT
In the fast dynamic digital world, the progress of stream processing technology experiences a dramatic shift in business operations transforming from a delayed periodic analysis to a continuous real-time insight on the fly. The ability to process this "data inmotion "has had a significant impact on all the real-time applications. Stream processing marks its foot print in almost all the application use cases like real-time fraud detection, prediction maintenance, enhanced customer experience, dynamic pricing and so on.Instream processing, First-In First-Out (FIFO) and non-FIFO streams are the two basic paradigms for the data to be processed challengingly based on the arrival order. Statistical stream aggregation is a crucial process in business analytics that summarizes and calculates high volume data using Sliding Window Aggregation (SWAG) techniques. In stream processing, there is a frequent possibility of occurrence of non-FIFO streams due to network issues. This research work introduces a time-efficient non-FIFO SWAG, Active Fix Lite which is an invariant of Active Fix technique that handles non-FIFO streams for a statistical stream aggregation process utilizing auto encoders as a dimensionality reduction tool to reduce the computational load that will be beneficial for streaming applications.

1. Introduction

The recent role of stream processing has evolved from a special tool for real time applicationstoafoundationalcomponentofdynamicdatadrivenarchitectures. Nowadays it's no longer just a matter of handling big data, but also enabling a new class of applications that can efficientlyreact to events onthe fly inthe dynamic streaming platforms. All real time applications rely on statistical stream aggregation [1] to bridge a gap betweenthe real time operational data with long term strategic goals to meet the competitive edge. Streaming aggregation is a process of computing summary statistics continuously over a group of unbounded data streams. Unlike batch aggregation that is computed over a finitedata set, stream aggregation updates incrementally on the dynamic data and produces resultonthe flyas new dataarrives or old data expires on a defined scope generallya window. As the streams are infinite, Aggregations are performed ona finite set ofdata using a windowing concept [2-8]. Among various windowing strategies, this work utilizes sliding window strategywhich is the popular technique used in real time applications. Streaming aggregation techniques utilizes aggregation and sliding window approach to extract valuable insightsfrom the data streams. This combined procedure is typically stated as Aggregate Continuous Query (ACQ)[9]. In the context of stream aggregation techniques, the right solution to extract abstract knowledge from the raw data for analytical purposes is Sliding Window Aggregation (SWAG). For a statistical stream aggregation process, the Stream processing system deals the data streams differently based on the arrival order. Data streams that are processed in the same sequence order from the source to thestream processing engine are termed as In order streams And that change in sequence are termed as out of order streams. Outoforderness in streamaggregation techniques is a significant challenge to be dealt onthe fly to maintain accurate real time analytics for making proactive decisions. A common solution to deal with out of order streams is by the use of watermarks, a time based control mechanism. Watermarks are used as a time-stamped signal that triggers the system when to close the window and produce the result. It helps to balance latency and accuracy of the system.

All stream processing platforms face a crucial challenge of maintaining a resource efficient, scalable and reliable system. To make proactive decisions instantaneously as the dataarrives,timecomplexity plays avitalroleanddirectlycausesanimpact onthe accuracy, resource usage and latencyofthe system. The objective ofthis work is to develop a time efficient SWAG technique to deal with the non-FIFO streams in an optimised manner. This work utilizes Active Fix, a non-FIFO SWAG technique developed bykalyaniand safish mary [10] to handle out-of-order streams using a control point based indexing. Active FixLite technique is an enhanced and an invariant ofActive Fix. It is developed by reducing the dimensionalityofthe input datainto avectorrepresentationusing AE and thencomputing the stream aggregation with Active Fix approach. Thus the proposed Active Fix Lite technique reduces the computational load as the reduced vector representation is alone incrementally updated instead of entire window elements during a window slide. The performance of this technique is measured using the time complexity metric and the experimental results are proved in an optimized manner.

2. Related Works

Rao et al.[11], designed a novel framework for anomaly detection in data streaming that meet the two key challenges with incremental learning of concept drift adaptation. The modelupdatesthe learning in adaptive environments over time. Also remain effective during concept drift.it is anauto encoder based built in drift detection mechanismthat adapt to data distribution changes without the assistance of labeled data.

Kim et al. [12], proposed a multi module unsupervised outlier detection Module particularly focusing on Indoor Air Quality (IAQ). It is a hybrid deep learning method that combinesLSTM(LongShortTermMemory)Encoderwithensemblingmethod. LSTM encoder learns to reconstruct time series data using reconstruction error concept and extract the latent features. A one class SVM is trained on these features to generate a second layer. Ensemble method combines theoutputsofthe two layer detectorsand provides a stable reliable detection rule with a single model.

Ahmed et al. [13], present a hybrid Convolutional auto encoder based model for real time anomaly deduction. The key feature of this work is the implementation of dynamic thresholding to adapt changes in data pattern using statistical methods like mahalanobis distance and moving averages. This model provides a significant improvement in accuracy and alert unusual data moments,which is crucial to preventfinancial loss and systemfailures.

Zervoudakiset al. [14], proposed a hybrid framework with auto encoder and attention mechanism for time series and anomaly detection. In this model, AE extracts local structural patterns (usually in windows) of time series data and the attention mechanism as in transformermodelsisusedtoidentifyandlearnlongtermtemporalrelationshipsand dependencies. This model implements an improved thresholding method based on statistical analysis of reconstruction error.

Hassanet al. [15], proposed an unsupervised multivariate time series anomaly detection framework with attention based ConvLSTM AE with dynamic thresholding. The attention mechanism is used to learn both spatial patterns and temporal dependencies. Andthe dynamic thresholding makes the system adapt to variations in data thus reducing false alarms.

Zhanget al. [16], introduced a hybrid anomaly detection architecture that combines denoising AE with memory module. Instead of depending solely on AE's learned representation, memory explicitly stores and manages normal trends of data from the recent past. Memory also allows the system to adapt to data distribution changes.

Paprockiet al. [17], implemented a deep learning based framework for detecting outliers in data stream patterns. This model works with the three phases namely data pre-processing, Deep Neural Network (DNN) training and detection phases.In the data pre-processing phase, the raw data is cleaned and prepared by normalising the data into a suitable format for learning.TheDNNtraining phasetrainsthepre-processor datausing a multi-layer layerDNNthatlearnssubtitlepatternsandrelationshipsinthedata.Inthedetection phase, the model detects the anomalous data patterns and flags it as an outlier.

Liu et al. [18], proposed a robust deep learning model with the Temporal Convolutional Network (TCN) AE model. TCN Capture long range dependency and data patterns from time series data in mobile devices and analyse for detecting anomalouspatterns. This model continuously updates its knowledge with adaptive learning mechanism making the system vulnerable to malicious data changes

Khan et al. [19], proposed a hybrid framework combining asymmetric Stacked AE (SAE) with one class DNN Classifier. This model offers a high detection rate in IoT security with minimal false positives. It acts as a dimensionality reduction tool and detect using one class DNN Classifier that trains on normal data and learns to differentiate between normaland anomalous data patterns. The work provides a reliable, efficient and high accurate solution for challenges in IoT security protecting against cyber threats.

Kingmaet al. [20], addressa major probleminprobabilistic modelling. They introduce the concept of Variational AE (VAE) Bayesian inference and DNN to efficiently train the complex data distributions. The Auto Encoding Variational Bayes (AEVB) Algorithm is trained to learn the previous of generating model from some underlying hidden (or latent) variables. This algorithm uses a probabilistic encoder and decoder mechanism.

3. Overview of Active Fix Technique

Active Fix is a scalable efficient fragment based bidirectional indexing approach to perform streaming analytics over out-of-order streams using incremental SWAG technique. Similar tothecheckpoint based indexingNon-FIFOSWAGtechnique CPiX implementedby Bou et al.[21], Active Fix also maintains a similar data structure to hold the oldest fragment data in a binary tree and a bidirectional array structure for the remaining fragment data.Unlike existing approaches that eagerly aggregate the intermediate results, CPiX maintainsthe intermediate results in an on-demand manner.

3.1 General Structure and Working of Active Fix Technique

Inspired by the performance of CPiX, Active Fix is developed to handle non-FIFO streams in an optimized manner. The Fig. 1 depicts the general structure of Active Fix to execute a window of elements by ACQ with a window size N and slide size S.

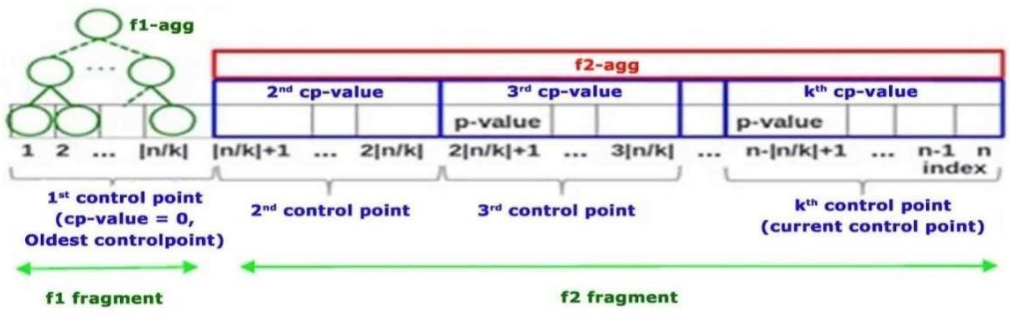

Fig.1 General structure of Active Fix algorithm

- For a window size N and slide size S, n elements from the data source are ingested in to the window and aggregated as p values in 'n' indexes based on the window slide and are represented as p_value. The p values in turnare grouped into k controlpoints. In order to handle the late arrival records, control points are maintained for the window elements in the re-computation of aggregation on the affected fragmentalone.
- The number of control points to accumulate the p values is chosen in such a way where one-third part of p values forms a control point with the tree structure represented as f1 fragment structure. The remaining two-third part of p values formthe control points with an array structure represented as f2 fragment structure asshown in Fig. 1. Thus, the entire fragment structure ofthe window is considered as f1 and f2 structures where f1 represents the binary tree structure maintaining oldest control point and f2 represents the array structure with remaining control points.
- Similar to CPix, Active Fix can use smallest possible control point k if the characteristics of non-FIFO are unknown; otherwise can use largest possible control point.
- The size of all the control points is equal when number of p values is exactly divided by k. If not, the size of the last control point alone differs.
- In the tree structure which denotes f1 structure, the p values represent the leaf nodes, and its aggregated values represent the internal nodes. The root node holds the aggregated value of entire f1 fragment and is denoted as f1-agg.
- The remaining f-values in the f2 fragment are aggregated and maintained as cp-value (one for each control point) as shown in Fig. 1. The cp-values are considered as control points which acts as the control structures to re-compute only the affected fragment that holds the late arrival records. The aggregation of cp-values ismaintained as f2-agg for the entire f2 fragment.

  On execution of ACQ on demand by the user, the aggregation of f1-agg and f2-agg values produce the result of the ACQ.

4. Proposed Active Fix Lite Technique

In the context of SWAG techniques, the improvement in reducing the time complexity relies on the reduction of computational load for aggregation and analytical purposes. In stream aggregation, time management is not only a resource managed feature, but also act as a central pillar of the system that controls the system's ability to betimely, accurate and scalable in real time applications. The proposed technique aims to handle non- FIFO streams in stream aggregation process with high dimensionality data in a time efficient manner. This technique utilizes auto encoders to address the key challenge of dimensionality reduction in stream processing. Auto encoders are atype of neural network which are trained with the reduced vector representation of data in an unsupervised way. It is often used to identify the anomalous patterns of data streams in real time. In this technique auto encoder acts as a dimensionality reduction tool that compress the input data into a low dimensional representation called latent space. In this SWAG technique, the function of latent space is to improve the efficiency by reducing the computational load. Thus this technique makes use of smaller manageable latent space representation of data within a window instead of the entire high dimensional input data for processing.

`Generally without the use of latent space, the time complexity of the SWAG techniqueis often proportionaltoWindow size (N)and thenumberofdata dimensions (D). i.e.O (N.D).By the use of latent space with much smaller data dimension (d << D), the new time complexity would be O (N. d). This drastic reduction makes real time Analytics more feasible in this SWAG technique.

In this work, an invariant of Active Fix Technique named Active Fix Lite is implemented in such away that the SWAG utilizes auto encoders to allow Active Fix technique to perform statistical aggregation based on the latent vector representation of the data points instead ofthe entire window's raw data.This efficient incrementalupdates in Active Fix Lite makes it much faster and more memory efficient than the state of art of solutions.

4.1 Architectural Design of Active Fix Lite

The architectural choice of auto encoder in the stream aggregation technique is crucial for its effectiveness.The architecturaldesign of AE in the streamprocessing systemconsists of two main components an encoder and a decoder. The architectural framework of sliding window aggregator with process flow utilising AE can be depicted using the figure Fig.2.

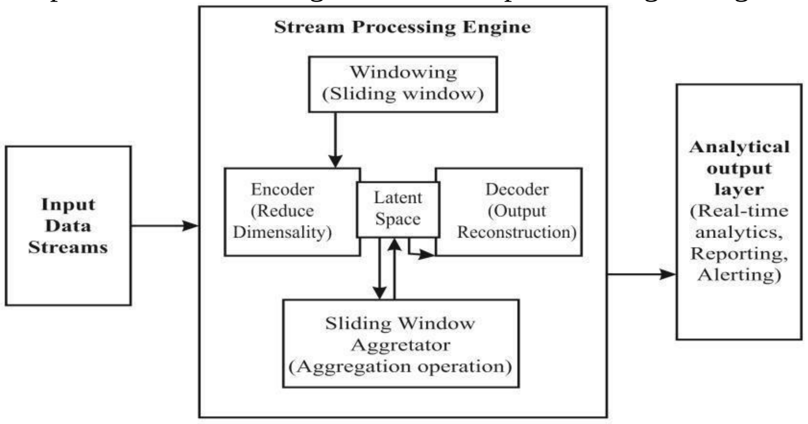
Stream Processing Engine
Windowing (Sliding window)
Input Data Streams
Encoder (Reduce Dimensality)
Latent Space
Decoder (Output Reconstruction)
Sliding Window Aggretator (Aggregation operation)
Analytical output layer (Real-time analytics, Reporting, Alerting)

Fig.2Architectural design of an auto encoder based non-FIFO SWAG technique Active FixLite

Input layer

In the context of statistical stream aggregation technique, relevant input data extracted from the data sources is fed to the stream processing engine for analytical purposes.

Stream processing engine

The Stream Processing Engine (SPE) acts as a central nervous system of the stream aggregation technique. The input data ingested into the SPE is achunk of data from the stream which is fed to the time series window using a sliding window strategy for the computation of statistical aggregation process.

Autoencoder

Auto encoders are the type of neural networks designed to learn anomalous data patterns froma reduced formof data. Theycontain two main components namely an encoder and a decoder.

Encoder: Data from each window is sent to the encoder toreduce the raw input into a low dimensional vector representation called latent space representation.

Latent space: Latent space is a part of the encoder layer which acts as the interface between the encoder and the decoder. It is the dimensionality reduced output of the encoder layer represented as a bottle neck ofa low dimensional vector presentation of original data. It is the most crucial part of the approach where efficiency is gained by performing statistical aggregation on this compact meaningful representation instead of the entire window data.

Sliding Window Aggregator:

The SPE performs the data stream process in this work using a sliding window aggregator for computing the statistical aggregation. This work utilizes the Active Fix model as a sliding window aggregator technique. When the window slides, Active Fix aggregator performs its computation with incremental updates only on the latent vector representation between the evicted and the new data. Thus it avoids recalculating all the high dimensional data in the window by reducing the computational load drastically and improves time complexity of the approach. The aggregated latent vector representation is fed to the latent space for subsequent processing.

Decoder: The aggregated latent vector representation from the latent space is sent to the decoderforreconstructionofthe original highdimensionaldata. The reconstructedoutput data is the final output of the auto encoder.

Output layer:

The time complexity of the statistical aggregation process done by the aggregator is fed to the output layer for reporting, alerting or analytical purposes. Also the reconstructed final output of the AE can be compared with that of original data and re construction error can be found. A lower error wouldindicate that the model is a good representation of latentspace.

4.2 ActiveFixLite MainAlgorithm

```
##LoadCSVdata

Readquantity,Unit_Price

Intializewindow_size,slide_size,nindex,ckpoint,threshold ##

□ Tree Structure Generation

├─Foriinrange(nindex):

│    ├─Createwindow:[start_time,end_time]

│    ├─Assigngroup_id=floor(i/ckpoint)

│    └─Storenode:[start,end,group_id,ckpoint_id,hr_start,hr_end]

 └─Result:Treeofwindowswithgroup/ckpoint IDs

##InitializeAutoencoder Model
```

```
├─InputLayer (2D:Quantity&UnitPrice)
├─HiddenLayers:Encoder →Bottleneck →Decoder
├─Optimizer:Adam, Loss:MSELoss
 └─Device:UseCUDAifavailable
##MainLoopOver EachTimeStep
  Foreachrecord(Quantity,UnitPrice,Time):
  ├─Appendrawvaluesto`raw`, `raw1`,`times`
  ├─SearchTreetofindwindow forcurrenttime:
              ├─Iffound→Append[Quantity, Price]tothatwindow
              │     └─Ifnotfound→⍰Delayorrebuildtree(fallbacklogic)
 ├─ifthe windowhas≥20records:
 ├─Converttotensor →ForwardpassthroughAutoencoder
 ├─Compute ReconstructionLoss=MSE(input,output)
├─ IfLoss >threshold:
    ├─AnomalyDetected
   ├─Aggregationcompute(ActiveFix)
    ├─Label= 1(Anomaly)
    │     └─AppendFixedValueto`fixed`
└─Else:
├─NoAnomaly
├─Appendoriginalvalueto`fixed`
└─Label= 0
└─RetrainAutoencoder(fewepochs) usingcurrentwindow
##Post-ProcessingasReport Generationwithsmalland largecheckpoints ##
        keeping slide timing constant
              ├─PlotvaryingwindowtimingsVstimecomplexityovertime ##
        keeping window timing constant
              ├─Plotvarying slidetimingsVstimecomplexityover time
```

5. Performance Evaluation

The effectiveness of the proposed Active Fix Lite algorithm is evaluated experimentally based on control points that act as a check point mechanism. In streaming applications dealing with stream aggregation, check points are snapshots of currentprocessing state taken at regular intervals.Check points play a crucial role allowing the systemto function ina fault tolerant, accurate, consistent and scalable manner. Thus, even in case of system failure, it can recover the process from the last good state. It also ensures results to be in an accurate and consistent manner.

In this research work, control points are used to handle the late arrival data for improving the efficiency of the system. The computational time of the data stream process is evaluated

Usingsmallestcontrolpointas

$$cp=\sqrt{n/\ln(10)} \text{ ....Eqn.1} \qquad \text{and}$$

Usinglargestcontrolpointas

$$cp=n/\sqrt{n/\ln(10)} \text{ ....Eqn.2}$$

All stream aggregation techniques generally rely on the time based metric, the Time Complexity (TC) which is used to measure the time requirement of the computational resourceofa technique to process each new data (usually each new window ofelements). TC is used as a crucial metric to measure the efficiencyof a technique. A low TC is essential for the stream aggregation techniques in order to keep up with the continuous arrival of data. When the processing time for each data exceeds the arrival rate, itleads to processingbacklog resulting in increased latency and data loss. When the volume or velocity of data streams increase, TC causes a significant impact on the performance of the processingsystem. As the size of the inputin streaming application isn'tfixed, the time complexity of the aggregation technique is generally measured for each window of elements instead of the entire stream.

Based On the size ofthe input, TC is often expressed using Big O notation describing the worst case running time of a technique. Generally it is defined as a function of input.

In this work, TC is Measured as a function in terms of input data, usually eachnew window ofelements.TC is measured For varying window and slide timings based onthe two scenario i.e., for small and large control points.

In case of small control points, TC is measured with the following function represented as Eqn.3

$$F(s) = (\lceil \tfrac{n}{k} \rceil) + 3 * P \quad \text{.......... Eqn.3}$$

Similarly incase of large control points, the TC is measured with the following function represented as Eqn.4

$$F(l) = (P+1) * \log(|\tfrac{n}{k}|) \text{...........Eqn.4}$$

Where

n is the number of partitions of a window element
K is the number of control points and
P denotes f1 and f2 partitions of the control points
Fig.1shows the parameters n, k, pin detail in section3.

6. Experimental Results and Discussions

The research work utilizes Kaggle datasets which are open-source public data collections generally used for research, training, prediction and so on. These datasets are a kind of synthetic data sets that are of great use for data science practices with machine learning models. Data is extracted from On-line Retail store data from kaggle for this work. Outofnearly33,000 transactionaldata, 901 datatransactions are considered for experimental evaluation.

6.1 ScalabilityperformancewithSmallControlpoints

ThescalabilityperformanceofActiveFixLiteapproachisevaluatedagainstActive Fix working using small control points. Results are taken for the following cases:

Case1:By increasingthewindowtimingintervalfora constantslide timing interval

In case of smaller control points, for a constant slide time interval of 3000 seconds with varying window timing intervals 18000s, 20000s, 22000 and so on, the time complexity is measured and depicted using the following Table 1 and figure Fig. 3.

Table1ScalabilityperformanceofActiveFixVs.ActiveFix Lite with small control points for Slide timing of 3000 s

Window Timings (s)	TimeComplexity (ins)	
	ActiveFix	ActiveFix Lite
18000	6	6
20000	6	6
22000	7.5	7.01
24000	11.09	7.82
26000	11.09	7.82
30000	14.07	9.15
32000	14.07	9.15
34000	17.84	10.24

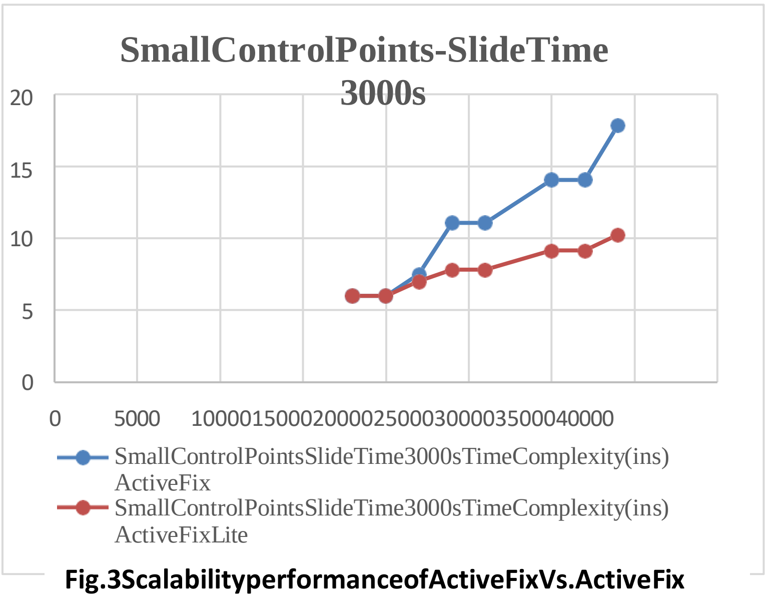

Fig.3ScalabilityperformanceofActiveFixVs.ActiveFix LitewithsmallcontrolpointsforSlidetimingof3000s

Case2: Byincreasing theslide timing intervalfora constantwindowtiminginterval.

In case of smaller control points, keeping the window time interval of 34000 seconds constant for varying slide timing intervals 2000s, 3000s, 4000s and so on, the timecomplexity is measured and depicted using the Table 2 and figure Fig.4.

Table2ScalabilityperformanceofActive Fix vs. Active Fix Lite with small control points for slide timing of 3000 s

Slide Timings (in s)	TimeComplexity (ins)	
	Active Fix	ActiveFixLite
2000	26.56	21.098
3000	17.84	10.242
4000	13.59	7.82
5000	6	6

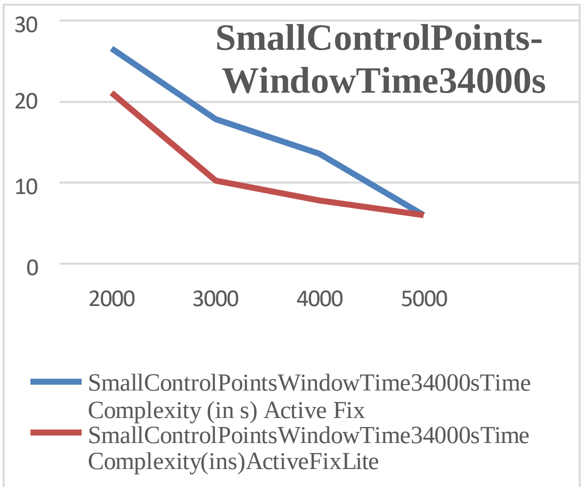

Fig.4ScalabilityperformanceofActiveFixvs.ActiveFix Litewithsmallcontrolpointsforwindowtiming of 3000s with varying slide timings

From the observations of Table 1, Table 2 and Fig.3, Fig.4, it is clear that Active Fix Lite outperforms Active Fix in computational time for varying window timings when small control points are chosen.

6.2 Scalability performance with Large Control points

The scalability performance of Active FixLite approach is evaluated against Active Fix working using large control points. Results are taken for the following cases

Case1: Byincreasingthewindowtimingintervalfora constantslidetiminginterval.

In case of larger control points, keeping the slide time interval of 3000 seconds for varying window timing intervals 18000s, 20000s, 22000s and so on, the time complexity is measured and depicted using the Table 3 and figure Fig.5

Table 3 scalability performance of Active Fix vs.Active Fix Lite with large controlpointsforSlidetimingof3000s

Window Timings (s)	Time Complexity (ins)	
	Active Fix	Active FixLite
8000	7.88	2.027
20000	7.88	2.027
22000	8.72	2.018
24000	9.39	2.820
26000	9.39	2.820
30000	12.90	9.158
32000	12.90	9.158
34000	14.19	4.242

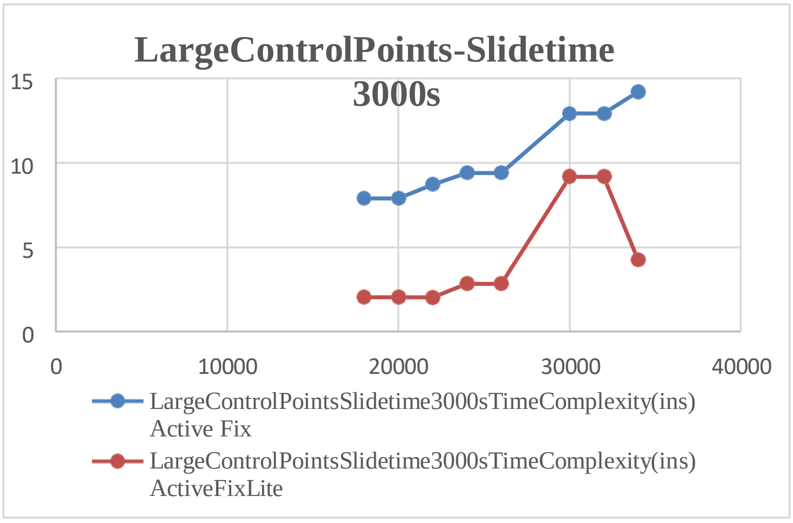

Fig.5ScalabilityperformanceofActiveFixvs.ActiveFixLite with large control points for Slide timing of 3000 s

Case2: By increasing the slide timing interval for a constant window timing interval.

In case of larger control points, keeping the window time interval of 34000 seconds constant for varying slide timing intervals 2000s, 3000s, 4000s and so on, the timecomplexity is measured and depicted using the Table 4 and figure Fig.6.

Table4 scalability performance of Active Fix vs. Active Fix Lite with large control points for window timing of 34000s s

Slide Timings (in s)	**TimeComplexity (in s)**	
	ActiveFix	**Active FixLite**
2000	47.55	14.098
3000	20.84	4.242
4000	16.09	2.820
5000	12	2.027

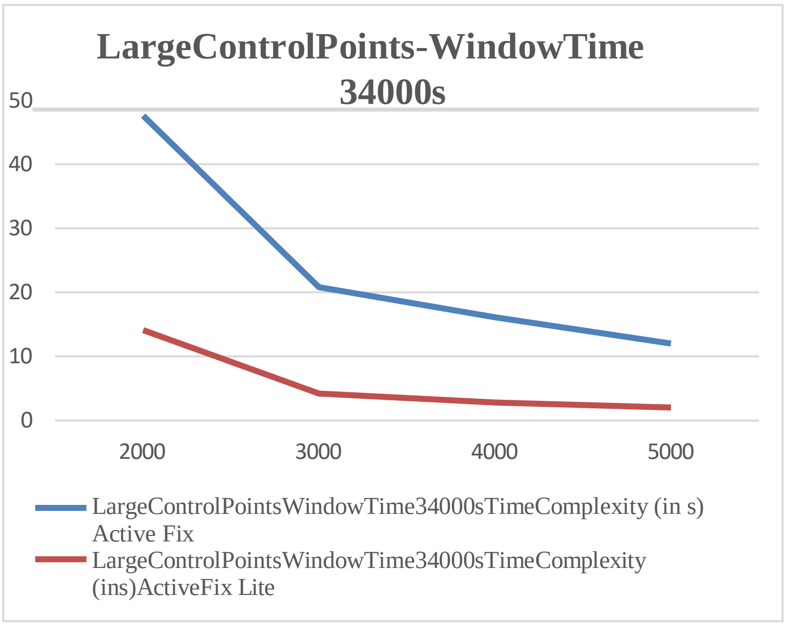

Fig.6ScalabilityperformanceofActiveFixvs.ActiveFixLite with large control points for window timing of 34000s for varying slide timing

From the observations of Table 3, Table 4 and Fig. 5, Fig. 6, it is clear that Active Fix Lite outperforms Active Fix in computational time for varying slide timings when large control points are chosen.

7. Conclusion and Future Enhancement

The incremental SWAG technique Active FixLite isdevelopedto handle non-FIFO streams using a sliding window aggregator approach and auto encoders for a resource efficient stream processing system. Time complexity is the fundamental metric used for measuring the performance of stream aggregation technique. In the context of stream processing, it helps to choose the techniques that are fast and more energy efficient. The performance of Active Fix Lite technique is experimentally tested using small and largecheck points for varying window and slide timings in order to measure the time complexity. The proposed Active Fix Lite technique outperforms Active Fix technique by reducing the computational load and is suitable for applications with adaptive window environments.

The future enhancement for this research work can be concentrated to measure the amount of out of orderness and train the model efficiently using machine learning techniques.

References

1. Hirzel, Martin, et al. ‘Sliding-Window Aggregation Algorithms: Tutorial’. Proceedings of the 11th ACM International Conference on Distributedand Event-Based Systems, ACM, 2017, pp. 11–14. DOI.org (Crossref), https://doi.org/10.1145/3093742.3095107. 1
2. Akidau,Tyler, Robert Bradshaw, et al. ‘The Dataflow Model: A PracticalApproach to BalancingCorrectness, Latency, and Cost inMassive-Scale, Unbounded, out-of-Order Data Processing’. Proceedings of the VLDB Endowment, vol. 8, no. 12, Aug. 2015, pp. 1792–803. DOI.org (Crossref), https://doi.org/10.14778/2824032.2824076.
3. Gedik, Buğra. ‘Generic Windowing Support for Extensible Stream Processing

Systems:AWINDOWINGLIBRARYFOREXTENSIBLESTREAMPROCESSING SYSTEMS'. Software: Practice and Experience, vol. 44, no. 9, Sept. 2014, pp. 1105–28. DOI.org (Crossref), https://doi.org/10.1002/spe.2194.

4. Ali, Mohamed, et al. 'Real-Time Spatio-Temporal Analytics Using Microsoft StreamInsight'. Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2010, pp. 542–43. DOI.org (Crossref), https://doi.org/10.1145/1869790.1869888.
5. Bou, Savong, Toshiyuki Amagasa, et al. 'Filtering XML Streams by XPath and Keywords'. Proceedings of the 16th International Conference on Information Integration and Web-Based Applications & Services, ACM, 2014, pp. 410–19. DOI.org (Crossref), https://doi.org/10.1145/2684200.2684309.
6. Bou,Savong,et al. 'KeywordSearchwithPath-BasedFiltering overXMLStreams'. 2014 IEEE 33rd International Symposium on Reliable Distributed Systems, IEEE, 2014, pp. 337–38. DOI.org (Crossref), https://doi.org/10.1109/SRDS.2014.63.
7. Bou, Savong, Toshiyuki Amagasa, et al. 'Path-Based Keyword Search over XML Streams'. International Journal of Web Information Systems, vol. 11, no. 3, Aug. 2015,pp.347–69.DOI.org(Crossref),https://doi.org/10.1108/IJWIS-04-2015-0013.
8. Bou, Savong, et al. 'Scalable Keyword Search over Relational Data Streams by Aggressive Candidate Network Consolidation'. InformationSystems,vol. 81,Mar. 2019, pp. 117–35. DOI.org (Crossref), https://doi.org/10.1016/j.is.2018.12.004.
9. Arasu,Arvind,et al. 'The CQL Continuous Query Language:Semantic Foundations and Query Execution'. The VLDB Journal, vol. 15, no. 2, June 2006, pp. 121–42. DOI.org (Crossref), https://doi.org/10.1007/s00778-004-0147-z.
10. Kalyani & Safish Mary M. (2024). Active Fix: An Optimal StatisticalAggregation of Out-of-Order Data Streams with Sliding Window using Control Point based Fragment Indexing. *Tamil Nadu Scientific Research Organization*, *15*(87), 85203–85218. https://tnsroindia.org.in/JOURNAL/issue87/IJONS

11. Bou, S., Kitagawa, H., &Amagasa, T. (2023). Cpix: Real-time analytics over out-of- order data streams by incremental sliding-window aggregation. *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, 3759–3760. https://doi.org/10.1109/ICDE55515.2023.00310
12. Li, J., Malialis, K., &Polycarpou, M. M. (2023). Autoencoder-based anomaly detection in streaming data with incremental learning and concept drift adaptation. *2023 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN54540.2023.10191328
13. Park, J., Seo, Y., & Cho, J. (2023). Unsupervised outlier detection for time-series data of indoor air quality using LSTM autoencoder with ensemble method. *Journal of Big Data*, *10*(1), 66. https://doi.org/10.1186/s40537-023-00746-z

14. Maitra, S., Kundu, S., & Shankar, A. (2024). *A real-time anomaly detection using convolutionalautoencoderwith dynamic threshold* (No. arXiv:2404.04311). arXiv. https://doi.org/10.48550/arXiv.2404.04311

15. Najafi, S. A., Asemani, M. H., &Setoodeh, P. (2024). *Attention and auto encoder hybrid model for unsupervised online anomaly detection* (No. arXiv:2401.03322). arXiv. https://doi.org/10.48550/arXiv.2401.03322
16. Hassan, R., &Mohamed, A. (2023). An attention-based ConvLSTM auto encoder

with dynamic thresholding for unsupervised anomaly detection in multivariate time series. *Information*, *4*(2), 135–149. https://www.mdpi.com/2504-4990/4/2/15

17. Bhatia, S., Jain, A., Srivastava, S., Kawaguchi, K., &Hooi, B. (2022). *Memstream: Memory-based streaming anomaly detection* (No. arXiv:2106.03837). arXiv. https://doi.org/10.48550/arXiv.2106.03837
18. Hassan, A. F., Barakat, S., &Rezk, A. (2022). Towards a deep learning-basedoutlier detection approach in the context of streaming data. *Journal of Big Data*, *9*(1), 120. https://doi.org/10.1186/s40537-022-00670-8
19. Owoh, N., Riley, J., Ashawa, M., Hosseinzadeh, S., Philip, A., &Osamor, J. (2024). An adaptive temporal convolutional network auto encoder for malicious data detection in mobile crowd sensing. *Sensors*, *24*(7), 2353. https://doi.org/10.3390/s24072353
20. Ayad, A. G., El-Gayar, M. M., Hikal, N. A., &Sakr, N. A. (2024). Efficient real-time anomaly detection inIoTnetworks using one-class autoencoderanddeepneural network. *Electronics*, *14*(1), 104. https://doi.org/10.3390/electronics14010104
21. Chen, Y., Liu, J., Peng, L., Wu, Y., Xu, Y., & Zhang, Z. (2024). Auto-encoding variationalbayes. *Cambridge Explorations in Arts and Sciences*, *2*(1). https://doi.org/10.61603/ceas.v2i1.33