CIKM2025

Fast Outlier Detection in Oblique Subspaces

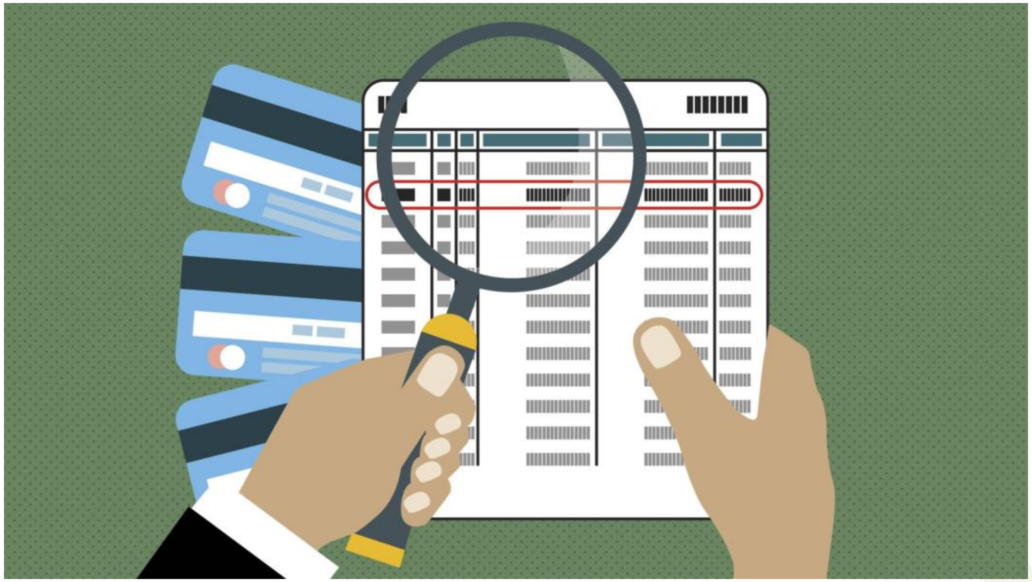
Bowen Li, Charu C. Aggarwal, Peixiang Zhao

Florida State University, IBM T.J. Watson Research Center

CIKM 2025 SEOUL
November 10 - 14

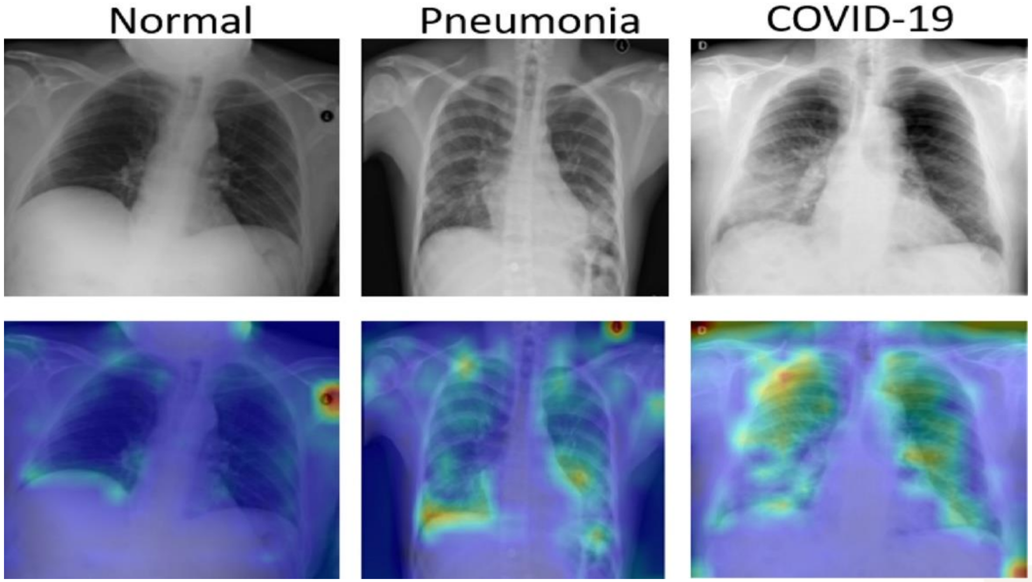
Introduction

- **Outliers**: rare, deviating objects with important insights



Introduction

- **Outliers**: rare, deviating objects with important insights



Motivation & Challenge

- **Outliers**: rare, deviating objects with important insights
- High-dimensional data → curse of dimensionality; irrelevant features mask outliers
- Subspace outlier detection explores lower-dimensional views
- **Limitation**: existing **subspace hashing** methods only work in *axis-parallel subspaces* → fail on **schema-less data** (e.g., time series, graphs)

Our Solution: OS-Hash

Contributions: *an ensemble, subspace hashing method in oblique subspaces*

1. Support subspace outlier detection for high-dimensional, schema-less data

2. Introduce the novel notion of oblique subspaces, where subspace outliers could arise

3. OS-Hash is constant-space, linear-time, and highly scalable

4. OS-Hash can be extended to the data stream scenario for subspace outlier detection

Key Idea: Oblique Subspaces

- Define subspaces not by fixed axes but by **pairwise object proximities**
- Works for any data type if a **similarity function** $\mathbf{S_{ij}=s(O_i, O_j)}$ exists
- The **projection** of $\mathbf{O_k}$ on the oblique direction

$$
\begin{aligned}
\mathbf{proj}(O_k) &= (\overrightarrow{X_k} - \overrightarrow{X_i}) \cdot (\overrightarrow{X_j} - \overrightarrow{X_i}) \\
&= \overrightarrow{X_k} \cdot \overrightarrow{X_j} - \overrightarrow{X_k} \cdot \overrightarrow{X_i} - \overrightarrow{X_i} \cdot \overrightarrow{X_j} + \overrightarrow{X_i} \cdot \overrightarrow{X_i} \\
&= s_{kj} - s_{ki} - s_{ij} + s_{ii}.
\end{aligned}
$$

- No need for explicit vector representation

OS-Hash Overview

Sampling	Projection	Hashing	Scoring

- Sample $\mathbf{r}$ pairs of objects at random from dataset O
- Form **r-dimensional oblique** subspaces for O

OS-Hash Overview

Sampling	Projection	Hashing	Scoring

- Select **a sample $S \subseteq O$** of objects at random
- **Project the objects** of S along each of r oblique dimensions
- Min-max normalization
- Convert to a **r-dimensional discrete vector** $\overrightarrow{\mathbf{Y}_{\mathbf{i}}}$

OS-Hash Overview

Sampling	Projection	Hashing	Scoring

- Construct a **Count-Min sketch** $\mathbf{H}$
- Apply each hash function $\mathbf{H_k}$ upon the discrete vector
- **Increment** the count value in the $\mathbf{H_k}(\overrightarrow{\mathbf{Y_i}}) -$ th bucket **by 1**

OS-Hash Overview

Sampling	Projection	Hashing	Scoring

- Transform each object $\mathbf{O_i} \in \mathbf{O}$ into **its r-dimensional discrete vector representation**
- Insert it into the trained Count-Min Sketch $\mathbf{H}$

- **Repeat m times**, once for each base detector of the ensemble

$$\text{OS-Hash}(O_i) = \frac{1}{m}\sum_{j=1}^{m} os_j^i$$

OS-Hash in Stream

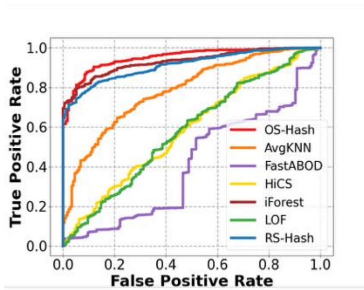
- **Data streams**: a continuous and rapid flow of data objects arrives in real time
  - Processed in one pass
  - A large amount of data is coming quickly and continuously
  - Rapid changes of underlying patterns
- **Sliding Window**: New objects are inserted into the sketch, and old ones automatically removed.
- **Time-decayed model**: Uses an exponential decay rate $\boldsymbol{\lambda}$ to weigh recent objects more.
  - If $\boldsymbol{t}$ objects arrive after object $\mathbf{O_i}$, its weight becomes $2^{-\lambda t}$

Experimental Setup

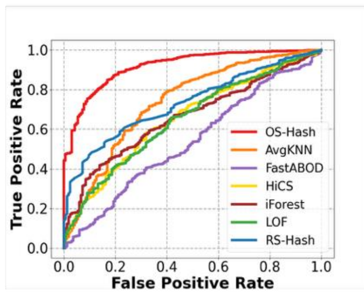
- **Dataset:**
  - Multidimensional
  - Time Series
  - Graphs
  - Streams
- **Baselines:** RS-Hash, AvgKNN, LOF, iForest, COF, LoOP, etc.
- **Metrics:** AUC (accuracy) + runtime or throughput (efficiency)

Results (Static Data)

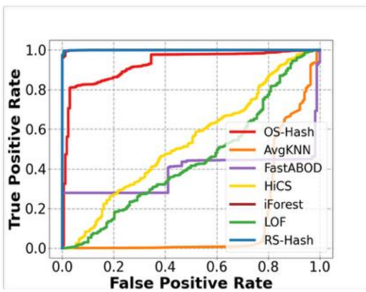
Dataset	OS-Hash	RS-Hash	AvgKNN	LOF	iForest	HiCS	FastABOD
LYMPHOGRAPHY	97.28	**99.92**	97.89	97.41	99.30	95.85	46.36
CARDIO	**94.98**	91.19	78.53	58.00	93.07	58.27	41.16
MUSK	**100.00**	100.00	24.10	39.17	100.00	39.50	48.78
WAVEFORM	**91.23**	72.97	73.83	65.03	66.20	65.23	53.68
KDDCUP99	92.91	**99.96**	14.35	44.69	99.94	52.19	38.27



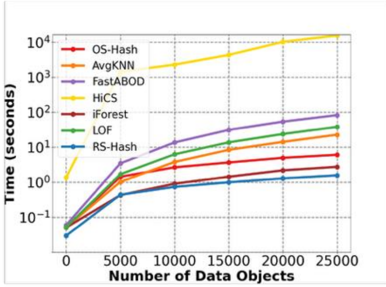
CARDIO



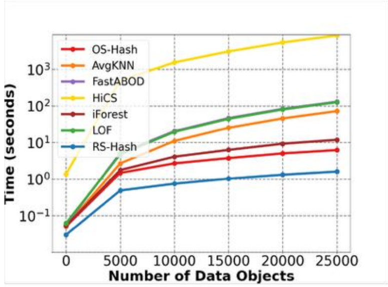
WAVEFORM



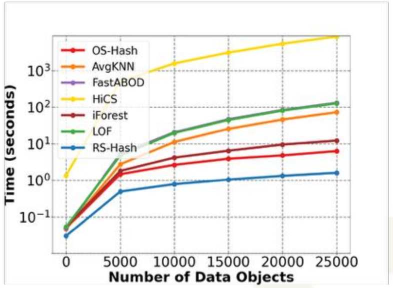
KDDCUP99



KDDCUP99



NORMAL(0,1)

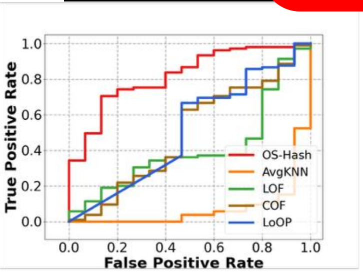


UNIFORM(0,1)

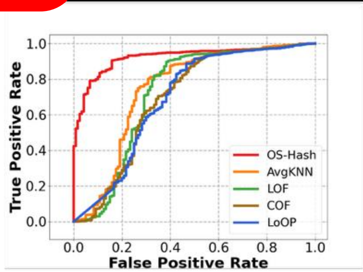
Multidimensional Dataset

Results (Static Data)

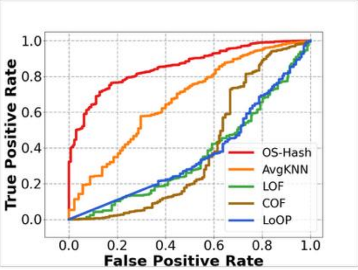
Dataset	OS-Hash	AvgKNN	LOF	COF	LoOP
PICKUP	**75.00**	74.50	73.00	68.50	63.75
PEBBLE	**79.17**	77.11	41.02	49.59	51.14
POWER	**66.03**	52.10	37.43	40.82	38.89
ECG5000	**92.17**	74.41	72.14	69.26	69.35
CROP	**83.96**	66.82	35.68	38.32	36.78



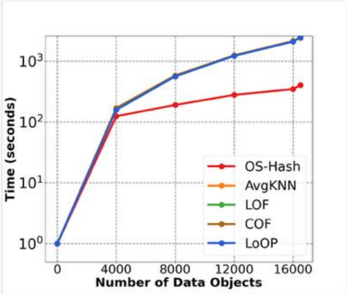
PEBBLE



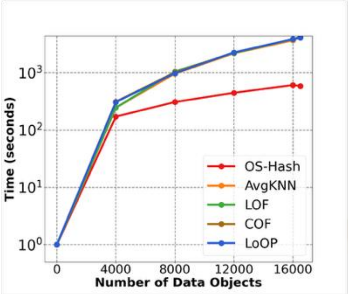
ECG5000



CROP



CROP

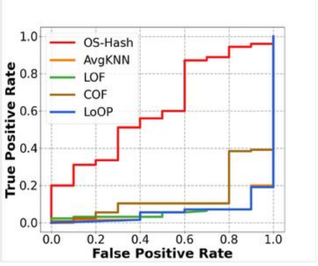


UNIFORM(0,1)

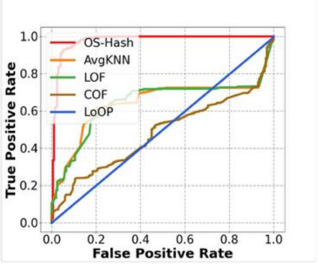
Time Series Dataset

Results (Static Data)

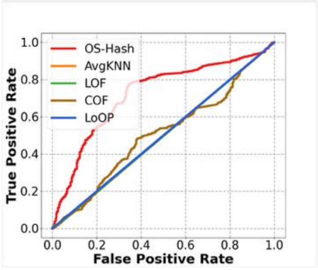
Dataset	OS-Hash	AvgKNN	LOF	COF	LoOP
MUTAG	**61.62**	5.76	6.04	13.84	5.52
FINGER	**55.03**	51.59	41.38	48.43	51.59
AIDS	**97.51**	64.43	64.22	48.09	49.75
MUTAGEN	**63.52**	56.51	55.14	60.40	58.05
TOX21	**71.97**	49.58	49.67	50.51	50.00



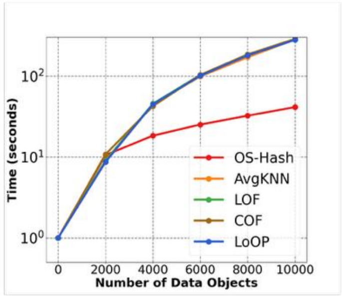
MUTAG



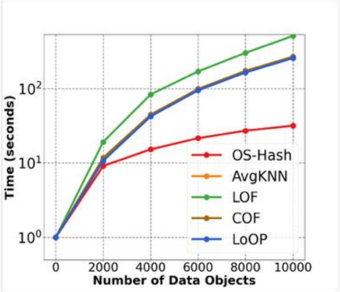
AIDS



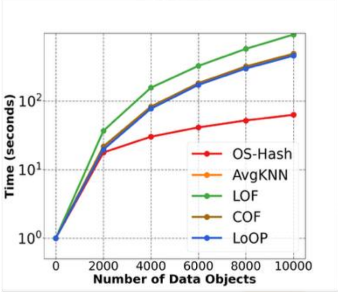
TOX21



TOX21



EDGE18



NODE18

Graph Dataset

Results (Stream Data)

Dataset	OS-Stream	RS-Stream	AvgKNN-Stream	LOF-Stream
ACTIVITY	**99.96**	84.51	33.59	38.55
KDDCUP99-T	87.09	**95.27**	12.43	66.35

AUC results in multidimensional data streams

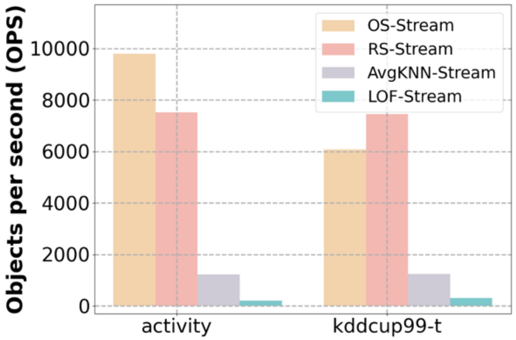
Dataset	OS-Stream	AvgKNN-Stream	LOF-Stream
ACTIVITY-T	**81.89**	35.08	40.57
CROP-T	**81.99**	71.89	52.97

AUC results in time series data streams

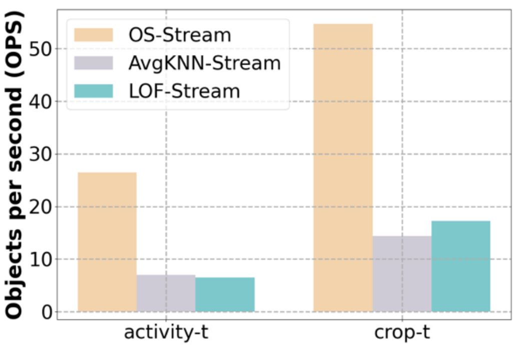
Dataset	OS-Stream	AvgKNN-Stream	LOF-Stream
TOX21-AR-T	**72.07**	52.87	53.64
MCF-7-T	**60.08**	52.94	56.18

AUC results in graph data streams

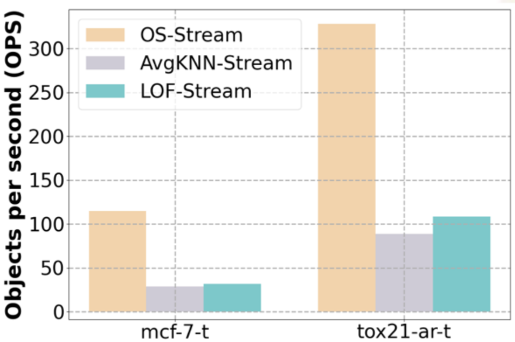
Results (Stream Data)



Multidimensional

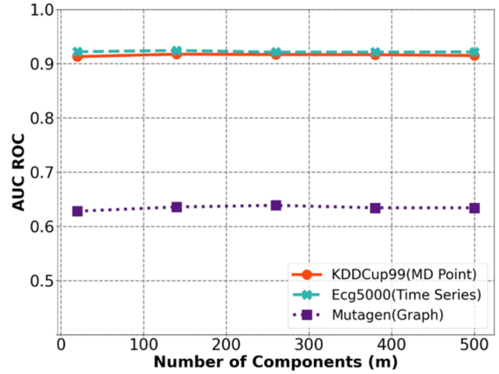


Time Series

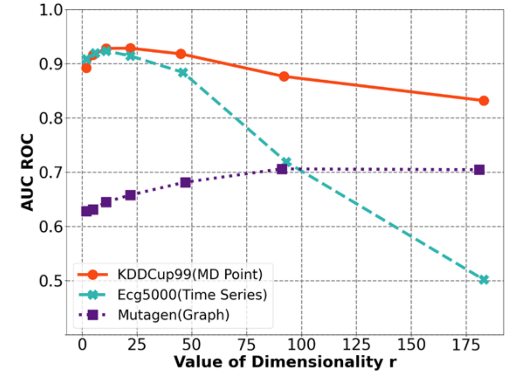


Graphs

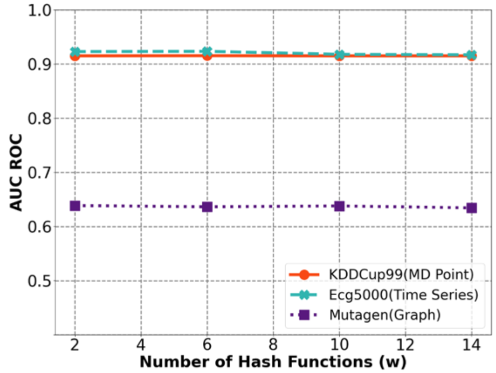
Parameter Analysis



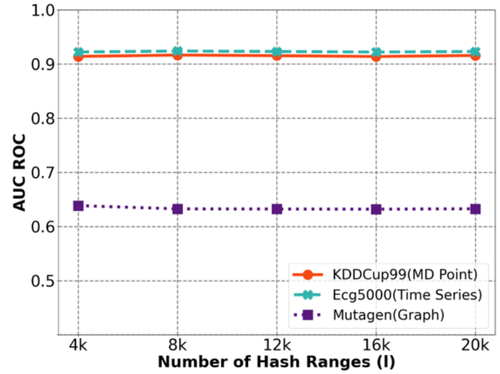
Base Detectors, m



Dimensionality, r



Hash Functions, w



Hash Table Size, l

Conclusion & Takeaways

- **New notion**: oblique subspaces → beyond axis-parallel limitations
- **OS-Hash**: linear-time, constant-space subspace outlier detector
- **OS-Stream**: first general streaming variant for arbitrary data types
- **Performance**: Outperforms state-of-the-art in both accuracy and efficiency

Contact Information

- **Author**: Bowen Li
- **Email**: bli@cs.fsu.edu
- **GitHub**: https://github.com/BowenLi1994/OSHash

CIKM SEOUL 2025
November 10 - 14

Thank You!