CPSC 5990 - Independent Study

Prasanta Bhattacharjee

Summer 2023

Contents

1 Motivation

My dedication to working in emergencies, particularly in the realm of severe weather events, poses unique and immediate threats, demanding a specialized set of skills and rapid response techniques.

The motivation to serve in the emergency and services is rooted in a deep sense of duty and a desire to make a positive impact on society. When others are fleeing from danger, our emergency responders run toward it, displaying incredible courage and resilience [12]. Their motivation comes from the understanding that their actions can mean the difference between life and death, between chaos and order, and between despair and hope.

1.1 What is Emergency

An emergency that causes severe environmental damage and constitutes a possible threat to human life may need the use of technical assistance and particular expertise to respond efficiently and mitigate effects.

1.2 What is Emergency Management

According to the Jin et al. [12] the primary functions of emergency management are collecting data, assessment, decision-making, and presenting. This encompasses everything from acquiring facts to making judgements. To make the most use of emergency resources, it is essential to understand the many phases involved in disaster management.

1.3 Purpose of Emergency Management Teams

The purpose of the team, is in charge of coordinating evacuation servicemen and material shipment, arranging disaster relief system construction, supervising emergency preventive production activities, and aiding in aid operations during calamities [12].

1.4 The purpose of using ensemble methods and data streams in emergencies

The integration of ensemble methods and data streams in emergencies aims to enhance decision-making precision and responsiveness. Ensemble approaches use multiple models to offer accurate predictions while minimizing errors. Simultaneously, data streams allow for real-time updates, ensuring that emergency response systems react quickly to changing events, spot anomalies early, and optimize resource allocation for more effective emergency management.

2 Introduction to Data Streams

2.1 What is Data Stream

When representing data components that become available over time, an endlessly long sequence of objects is called a data stream. It is important to acknowledge that information may be retrieved at any moment and in any sequence. [20].

2.2 Data Stream Model

Combining streams from many sources might result in unstructured data streams with random contents or structured data streams with parts that follow a certain format for reorganising the source [20].

2.3 Type of Structure Model

Structured streams come in three most important varieties that vary in the manners that their constituent parts are associated with and impact one other [20].

2.3.1 Turnstile model

As the vector of the elements, each element in the stream represents a new element that has been added to or subtracted from the underlying vector.

2.3.2 Cash register model

The stream's constituent parts are only extensions of the fundamental vector, which they are never able to deviate from.

2.3.3 Time series model

Every segment within the stream is regarded as a distinct vector.

3 Introduction to Ensemble Methods

3.1 What is Ensemble Method or Learning

Several models are fitted utilising various data segments or distinct computations employing group techniques in order to enable predictive execution of learning computations [26].

3.2 Types of Ensemble Method

There are different methods for data-centered ensembles are used to processed the data properly [21].

3.2.1 Bagging

It is an algorithm that is entirely dependent on the given data. The term describes the process of dividing up an enormous amount of data into from original ones.

3.2.2 Boosting

The process is well-organized. Identifying and creating a suitable solution to address the mistakes caused by prior algorithms is the aim of this approach.

3.2.3 Stacking

It arranges data from several algorithms and presents them as a single algorithm.

4 Data Stream Research

This section provides an in-depth discussion of data streams, along with various problems faced by various authors and how they solved them and how to do better in the future using these methods.

4.1 Continuous Queries over Data Stream

In order to effectively handle and process the incoming data stream into the system, it needs efficient frameworks and algorithms. The work by Babu and Widom [1] demonstrates the difficulty of processing the streaming data with conventional mechanisms of a traditional RDBMS like triggers and materialized views.

The authors found that existing methods of stream processing, including Xpath and Xyleme, both have a specified query programming language and a certain range of functionality. In addition, other methods such as NiagaraCQ deal with scalability concerns for combining many queries, and OpenCQ monitors continuous queries based on incremental view maintenance. All of these methods never use any novel architecture, triggers, or materialised concepts.

A streaming query, according to the authors, is ongoing access to a single-supplied object. They developed an architecture with four components (Stream, Store, Scratch, and Throw) and six steps to ensure that the data objects that are requested are always consistent. In the first step of architecture, the stream component receives the incoming tuples into the system. The tuples that are not part of the current result in the future are separated and relocated to the store to confirm that it contains tuples that will not be shown in the results. The scratch component also perform the same task like store where scratch physically keeps the tuples in memory to make sure that if we send a query request, it can return the results. The final element, throw, collects all unnecessary tuples into it, which reduces the space in the system.

The authors described the techniques employed, which included employing triggers and materialized views to store and evaluate the incoming stream, rather than evaluating the effectiveness of the recommended architecture. Incoming data is processed using the trigger and components like Stream and Store may be left unfilled. Scratch is used as a search tool to evaluate the conditions. Similar to this, materialized views keep the views in Store and the data which is not included in conventional tables are stored in Scratch.

The network monitoring model is one of the alternatives to traditional stream processing research that the authors proposed. It will support the resolution of computational and system research problems.

4.2 Research issues in outlier detection for data streams

The problem of outlier detection for data streams (single stream and multiple streams), is more challenging than detecting outliers in non-stream data because of the exceptional qualities of data streams. The work by Sadik and Gruenwald [25] addresses multiple characteristics such as,

1. Transiency - a specific amount of time, after which it is discarded or archived), uncertainty (unreliability of the data points.
2. Dynamic data distribution - single data and multiple data in parallel.
3. Multidimensionality - handles all dimensions independently and fails to address the correlation among dimensions.
4. Dynamic relationship - asynchronous behavior and concept drift.

In addition, the authors also mentioned that due to the specific characteristics of data streams, such as high arrival rate, real-time, and change of data distribution or concept drift, compared to ordinary data, this form of data presents a unique set of research challenges for outlier detection.

The authors do not provide a literature survey. According to the authors a formal method for outlier detection has not been proposed.

The authors give a thorough outline of the research problems that must be solved in order to design a successful outlier identification method for data streams. The outlier identification method assumes the presence of labeled data in the supervised approach, which is the first method. The second method is known as the semi-supervised method where only inliers or outliers need to be labelled. The third technique, which is unsupervised and does not require any form of labeling, is particularly well-liked for outlier detection.

The authors do not provide any specific evaluation result of outlier detection techniques for data streams. Instead, they address research problems that need to be solved to develop efficient outlier detection techniques based on their proposed strategies for data streams.

Also, the authors do not suggest any future research directions.

4.3 Open challenges for data stream mining research.

Data stream mining is a field of study that studies methods and algorithms for extracting knowledge from fluctuating stream data. The work by Krempl et al. [14] addressed the eight core problems these include model simplification, privacy and security protection, legacy system management, stream preprocessing, information synchronization and availability, relational flow mining, event data analysis, and evaluation risk for the effective mining of data streams.

The authors state that traditional data mining methods are designed for static data sets and are ill-equipped to handle the dynamic, continuous nature of data streams that are employed posed by real-world applications. They also mentioned that there is currently no systematic method for handling data streams.

The authors elaborately describe the issues rather than propose any methods to overcome them. According to the authors,

1. Protecting data privacy- ensuring confidentiality in the analysis of streaming data is a crucial challenge in data stream mining.

2. Dealing with legacy systems- integrating data stream mining techniques into existing infrastructure poses challenges due to the compatibility and adaptability of legacy systems.

3. Handling incomplete and delayed information- real-time analysis of streaming data requires techniques to handle incomplete and delayed information, which is common in real-world applications.

4. Analyzing complex data streams- data streams often contain high-dimensional, heterogeneous, and evolving data, making the analysis challenging.

5. Evaluating stream mining algorithms- the performance and effectiveness of stream mining algorithms in real-world applications is a challenge due to the dynamic nature of data streams and the absence of ground truth.

6. Developing event detection methods- the methods for detecting events and predicting models for censored data in streaming environments is required.

7. Establishing a multi-criteria view towards evaluation- considering the absence of ground truth about how data changes.

8. Developing online monitoring systems- to ensure the reliability of updates and balancing the distribution of resources.

9. Relational flow mining.

The authors do not provide any specific evaluation result. However, they focus on highlighting the problems.

Based on the issues mentioned by the authors, they indicate that future research should focus on developing the methods to ensure the security of incomplete information when data arrives, managing inadequate delays or costs, relationships between streaming entities, proper event detection methods, predictive models for censored data, systematic methods for streaming pre-processing, setting up multi-criteria views for reviews, online monitoring system development, reliability of updating and the balance of resource allocation.

4.4 Other related works

4.4.1 Data Streams Management: Multidimensional Summary With Big Data Tools

The work by Sarr et al. [28] addressed that processing and storing multidimensional data streams is a real challenge, as it decrease the chances of querying them later.

The authors identify that previous works on data stream summarizing were unable to deal with high-dimensional data. Instead, particular summary techniques were designed to satisfy particular needs.

The authors propose a novel multidimensional summary method called Stream Cube Cascade Mode. The whole process, combined with an update function, an initialization function, and a proper data structure, is made possible by small chunks of data. The initialization function makes rectangular units of the data structure and incorporates data into them. The update function upgrades to these rectangular units when it is required.

The authors do not explicitly mention the results of an evaluation. By using the proposed method, they mentioned that it will optimize storage and processing with scalability.

Future research according to the authors should focus on applications like analysing emotion and identifying fraudulent activities in the energy and healthcare industries by using online analytical processing.

4.4.2 Distributed Publish/Subscribe Query Processing on The Spatio-Textual Data Stream.

The primary problem addressed by Chen et al. [6] is the efficient processing of spatio-textual queries in a distributed environment.

Existing systems such as IQ-tree for indexing the structure, NiagaraCQ for common factor identification from queries, Spatial Hadoop for grid index, PSoup and Tornado are costly to implement, unable to handle load balance and stream of subscription queries.

The authors propose a distributed publish/subscribe system called PS2Stream for processing spatio-textual data streams. The proposed system has three components dispatcher, worker, and merger. The dispatcher takes as input a stream of spatio-textual objects, and it receives two types of requests from users, namely submitting new subscriptions and dropping existing subscriptions, which correspond to two operations, query insertions and deletions. The worker accepts the workload sent from the dispatcher and conducts query insertion and deletion. Finally, after getting the duplicate matching results from workers are removed, and the merger sending the results to the users.

The authors conducted the experiment with large two datasets of spatio-textual tweets. The entire procedure carried out by operators that are AND or OR. The outcomes demonstrate how their method beats earlier baseline methods in terms of the average number of tuples and the average amount of time each tuple stays in the system, and it dramatically reduces on query processing times, workload balancing, and distribution.

The authors do not propose any further future research pathway.

4.4.3 Spatial-Aware Approximate Big Data Stream Processing

The problem mentioned by Jawarneh et al. [11] is the costly online use of spatial queries in non-stationary environments, where it is frequently impossible to have access to the entire dataset population. The lack of spatially aware online sampling methods in big data management systems today and the dependence on randomization, which introduces uncertainty into total estimation, aggravate this issue even more.

The existing versions of stream processing engines (SPE) are not ideal for geographic applications because earlier work only focused on balancing the throughput/latency colliding needs, without taking into account the spatial peculiarities of arriving data streams.

The authors developed a QoS-online spatially-aware sampling technique for estimating spatial parameters in real time. The technique operates dynamically on top of Spark Structured Streaming, a cutting-edge micro-batch-based SPE representation. It is transforming the two dimensional into a one-dimensional space by using specifically a space-filling curves (SFC) method called Geohashing.

The proposed strategy was assessed and contrasted with industry standard baselines. The approach was evaluated on a cohort of six months' worth of data from NY City taxicab journeys, or about nine million units. The performance evaluation's findings showed how well the suggested approach worked to provide precise approximations for spatial query answers in a spatially aware way.

Subsequent investigations may concentrate on refining the suggested approach, investigating its scalability and suitability for other spatial queries and datasets, and contrasting it with other extant spatially-aware sampling methodologies to ascertain the skewness and data arrival rate.

4.4.4 Continuous Query Processing in Data Streams Using Duality of Data and Queries

The problem being addressed by Lim et al. [18] is the issue of asymmetrical behaviour, which includes data and queries arriving continuously, quickly, unboundedly, and in real-time in continuous query processing systems.

According to the authors, existing techniques have only concentrated on data-initiative methods, which involve choosing a data element to start the question processing process. Query-initiative methods have not been considered.

In order to treat data and queries symmetrically, the authors introduced a duality model of data and queries. Their continuous query processing approach uses a multi-dimensional spatial join which is called Spatial Join CQ. It looks for pairs of overlapping regions between a set of data elements and a set of queries in the multi-dimensional space.

The authors evaluate how well basic selection queries are processed against sliding window join inquiries, when compared to previous approaches, the algorithm delivers notable performance gains. It outer forms older techniques by as much as 36.2 times in batch processing, up to 3.2 times in immediate processing, and up to 6.9 times in sliding window join continuous queries.

Future work should concentrate on developing a technique for automatically determining the ideal size of the data cluster and refining the algorithm for processing multiway sliding window join continuous queries.

4.4.5 Window Query Processing for Joining Data Streams with Relations

Towne et al. [30] addressed the problem efficiently involving both of processing queries data streams and relations and maximizing the total importance of the approximation results while considering system resource limitations.

Earlier techniques support queries only with static relations. The authors mentioned that these systems either do not support the joins of stream with traditional relations, or transform relations into streams beforehand.

The authors present techniques to process queries that join data streams with relations, without treating relations as special streams, and focus on a typical type of such queries, called star-streaming joins. They process these queries based on the semantics of (sliding) window joins over data streams and apply a load shedding approximation and semantic load shedding techniques when system resources are limited. A count-based and time-based windows are common types of windows, and the authors consider a time-based window of size T holding tuples that arrived during the last T time units tuples arriving within the last T time units are stored in a time-based window of size T. Random load shedding techniques randomly drop tuples from a data stream without regard to the output the tuples may produce when joined with another data stream. The semantic load shedding technique avoids the randomness by choosing to drop tuples with low probabilities for matching tuples in the opposite data stream to maximize the query result size.

By the authors compare the effectiveness of offline and online algorithms, experiments were carried out. The result indicate that the method incorporating pre-filtering performs better than the one that does not. Due to an increase in pre-filtering power, its performance improves as the size of the fact relation decreases.

Reducing big fact relation sizes, creating star-streaming join access methods, researching queries combining various data streams, and investigating DBMS system problems are the main areas of future research stated by the authors.

4.4.6 Research Issues in Mining Multiple Data Streams

The authors Wu and Gruenwald [31] addresses the problems of the lack of research on multiple data streams (MDS) mining, the complexity compared to single data streams, challenges in data integration, synchronization, and analysis in mining.

Most of the existing work in data stream mining has focused on single data streams and did not focus on the complexities and challenges associated with multiple data streams.

The main focus of the authors is on the characteristics of mining multiple data streams (MDS), and they propose a formal definition for that. They also analyse MDS applications in terms of mining requirements, like large-scale scientific observation, network traffic analysis, and web log analysis. Finally, they identify research issues related to MDS mining, such as Heterogeneous Synchronisation processing, approximate compression of local data, merging data streams in order, confidentiality versus data disclosure, and multiple univariate time series versus multivariate integration at each data source.

Since it is more of a conceptual and analytical discussion on MDS mining. The authors do not present a specific methodology or evaluation outcomes.

According to the authors, future work should focus on the development of technologies that address the problems found in scientific discoveries across a wide range of fields such as ecology, biology, and oceanography, and enhance the decision-making process in real time.

4.4.7 Adaptive Query Processing in Data Stream Management

The authors Farag et al. [8] highlighted the challenge of providing data stream monitoring systems with all the resources they require, given the periods of high volume, low volume, and stillness in data stream sources.

According to the authors, previous research that substitutes row-by-row data representations with column-oriented physical design approaches in data stream contexts has constraints when it comes to column store deployment. Moreover, the implementation of data management strategies with timely, ordered output release and minimal system resource consumption did not take into account the strong constraints of effective query processing.

For adaptive query processing in data stream management systems with constrained memory resources, the authors suggest two algorithms: EM-SWJoin and ADEDAS. To manage varying data arrival rates and reduce disc access latency, EM-SWJoin makes use of external memory data structures. While minimising the effects of processing delays, ADEDAS ensures an ordered release of output findings.

An extensive assessment of the suggested algorithms is not provided by the authors.

However, they suggest the implementation of column-oriented data representation in data stream systems as the main focus of future study.

4.4.8 A continuous query evaluation scheme for a detection-only query over data streams

The problem that the authors Park and Lee [24] are attempting to address is the missing of specific work required to handle a detection-only query form in a data stream context.

The authors state that previous studies have focused on developing a robust multi-way join managing element because to its higher cost as compared to unsigned administrators for a query processing such as prediction or decision.

The authors propose a new query evaluation scheme called StaMe for a detection-only continuous query. The approach works by creating a matrix-based synopsis for each source stream of an n-way path join query, where each source stream has a minimum of one and a maximum of two join predicates provided. The number of entries in a synopsis at compile time is determined by the number of buckets in the hash functions of the join attributes in the corresponding source stream of the synopsis. The entry itself tracks the number of tuples whose join attribute values are hashed into each matrix-based synopsis entry for a source stream.

To illustrate the proposed scheme, the performance of an n-way path join detection-only query is experimentally investigated on two synthetic datasets and one real data set. The evaluation of the results shows that the input rate of each source stream is varied from 1 to 30 tuples/sec. As the number of buckets for a join attribute domain is increased, the packing density of a join attribute synopsis decreases, so that the error rate is also decreases.

Future research directions are not provided by the authors.

4.4.9 A burst resolution technique for data streams management in the real-time data warehouse

The authors Majeed et al. [19] address the challenge of efficiently managing data streams in live data storage environments with a focus on handling spikes of data includes the rate of entry and the machine's capacity, which can lead to information loss and execution delays.

The authors noted that while previous research concentrated on joining and loading data streams into data warehouses, bursts of streams remain unsolved, which eventually reduces data dropping because of the unpredictable nature of arriving bursts of data streams.

The authors suggested using a token bucket system as a flow control method in a real-time data warehouse to manage sudden influxes of data streams. Tokens are accumulated at a predetermined pace in the token bucket, which is then used to regulate the data stream flow. Tokens are used to allow data to go through during bursts in data streams, guaranteeing a controlled and balanced flow.

The experiments are based on synthetic data comprising smaller stream of 5000 tuples, and larger stream of 25000 tuples. The memory size is set 20 percent of the input stream size. Based on experimental results, it is found that the drop rate of data streams is reduced up to less than 5 percent.

To demonstrate the technique's uniqueness from other approaches, future study should concentrate on applying it to various datasets—aside from synthetic data—and conducting a comparison with them.

The authors suggest that future studies concentrate on applying the method to various datasets, excluding synthetic data, and contrasting them with other methods to identify their positions.

4.4.10 Design of a Scalable Data Stream Channel for Big Data Processing

Lee et al. [15] address the challenge of coordinating data input across different interaction directions, which can lead to problems with transmission delay, the significance of speed and efficiency when processing large amounts of data, and how parallel processing is used more widely to handle demanding tasks.

The authors do not explicitly mention the limitations of previous works.

However, they propose a stream channel design that can handle large amounts of data properly. The method is designed for two things a multi-instance task and a data stream channel. The multi-instance task performs general operations such as reading the input, sending it to the stream, and outputing the streams in a proper file format. Tasks are frequently conducted in response to incoming streams of input or output from earlier tasks. Data stream channels create two fundamental transport protocols pub-sub and push-pull which, respectively, disperse streams to all connected peers and distribute messages to connected peers. In order to analyse data streams and perform aggregation tasks efficiently, they also build a fast-path data stream channel with two transmission methods: round-robin and key-value fashion.

As an initial result, the authors demonstrate the elapsed time of various tasks and analyse the data processing time of a sample directed acyclic graph (DAG) with six servers. Multiple-instance tasks, such multi-instanceSplitTaskRR and multi-instanceSplitTaskKV, are mentioned as producing almost the same channel throughput and having almost the same elapsed time as the offspring jobs. Given that the tasks in the graph are divided into smaller, parallel-executed sub-jobs, multi-instance jobs have the least amount of work overall and a shorter elapsed time.

Future research directions are not provided by the authors.

4.4.11 Prioritized Query Shedding Technique for Continuous Queries Over Data Streams

Helmy et al. [10] addresses the unexpected variation in arrival rate in data stream applications, which can lead to a system overload. The ongoing processing of submitted queries, which increases the system's processing burden, makes this issue worse.

The authors noted that former methods prioritised shedding based on discarding input tuples based on regions' priorities, or assumed that all inquiries are equally essential, without taking the priority of a query as a whole into account.

They propose a novel strategy called Prioritized Query Shedding to address the identified limitations. This technique involves dynamically prioritizing queries based on their importance and resource requirements. Lower-priority queries are shed or postponed ensuring that higher-priority queries receive adequate resources and are processed promptly.

In order to reduce the system load, the authors compared the proposed method with shedding zones. The comparison is based on calculating the output ratio between the number of results that were generated when the system was overloading and the number of results that were anticipated. The suggested method has been assessed and contrasted with alternative shedding methods. However, no particular test findings or numerical measurements are given by the authors.

According to the authors, future work should focus on extending query shedding techniques to handle the load when having multiple continuous queries running on multiple streams without impacting the standard of the output.

4.4.12 An Iterative Strategy for Deep Learning Classification on Spatial Data Streams

The authors King and Osborn [23] addresses the problem of classification of data in a spatial data stream, which has received very little attention compared to the classification of spatial objects in a static data set.

Prior research in geographic data stream mining has mostly concentrated on k-nearest neighbour search and grouping. The extant literature on the categorization of streaming geographic data suggests a tree structure to expedite entropy computations during decision tree construction. It does not delve into the use of neural networks.

The authors suggest employing deep neural networks for the continuous classification of geographic input streams. Three factors oversaw the entire process: initially, all the data needed to be available for the purpose of evaluating and training a classifier. Secondly, despite the availability of the all data, it was not possible to keep it all in a server and third, it is essential to develop and assess the model step-by-step.

The proposed method was evaluated experimentally by training the neural network gradually using a stream of spatial objects as input and comparing the results to a fully trained network to assess accuracy. Three distinct splits were employed by the authors to prepare and evaluation: 90/10, 80/20, and 70/30. According to the findings, the incremental strategy's accuracy increased with the proportion of training items. But the 80:20 split was the most accurate when compared to the accuracy obtained when the complete dataset was evaluated. There was a greater discrepancy between cumulative and total accuracy in the splits 90:10 and 70:30, despite having the greatest average cumulative accuracy.

The suggested method may be enhanced, according to the authors, to accommodate real-time information mining of geographical information streams.

5 Ensemble Method Research

This section gives a brief summary of the main ideas behind the ensemble approach and talks about how the authors, who have had a big impact on the field overall, see how it will impact many other disciplines.

5.1 Bagging Predictors

Breiman [3] addressed the problem of how to improve the accuracy of prediction models, especially when the prediction method is unstable. The author proposes the ”Bagging predictors” method as a solution to this problem which can give substantial gains in accuracy.

The author does not provide a literature survey but instead focuses on introducing and demonstrating the effectiveness of the ”Bagging predictors” method for improving accuracy in predicting numerical outcomes or classes. The author mentions that the method may not work well if the prediction method is already stable or if the bootstrap samples are similar to each other. Additionally, the method may not be suitable for very large datasets due to computational constraints.

The proposed Bagging method by the author is designed to improve the accuracy and robustness of predictive models, particularly decision trees, by combining multiple models trained on different subsets of the training data. The process is divided into different steps. Firstly, the data must be prepared. The preprocess dataset is cleaned and categorical variables are encoded if needed. The dataset split into two parts: a training set and a testing set. Bootstrap Sampling involves randomly selecting data points from the training set with replacements but may contain duplicates data points. Secondly, Model Training is based on selecting the base model for prediction and decision trees are a common choice due to their ability to handle both categorical and numerical data, making them versatile for various types of datasets. Thirdly, Aggregating Predictions are used to make predictions on the testing set or new data once all the individual models are trained. It works with two types of problems. One is the classification problem, which use majority voting. Each model votes for a class, and the class with the most votes is chosen as the final prediction. The second one is the regression problem, which can average the predictions made by each model to obtain the final prediction. Before final deployment, tuning parameters is essential it will optimize the performance of the bagged ensemble. Once the performance of the bagged ensemble is satisfactory, it is ready to be deployed for making predictions on new, unseen data.

The author uses commonly used datasets for classification and regression tasks. For classification author used the UCI Machine Learning Repository like waveform (simulated), heart, breast cancer (Wisconsin), ionosphere, diabetes, glass, and soybean. For Regression tasks used five type of datasets such as Boston Housing, Ozone, Friedman 1, Friedman 2, and Friedman 3. To evaluate the results, the method give substantial gains in accuracy when performed the test using classification and regression trees and subset selection in linear regression. It is consistently reducing the prediction error compared to a single model. For classification, the reduction rates range from 6 percent to 77 percent, and for regression trees, the rate ranges from 21 percent to 46 percent. It performs well even when the base model is unstable or prone to over-fitting.

The author do not mentioned any future research direction.

5.2 A streaming ensemble algorithm (SEA) for large-scale classification

The work by Street and Kim [29] addressed the problem of dealing with the challenge of large-scale or streaming classification and the limitations of traditional ensemble methods, such as Boosting and Bagging, in this context.

The authors state that existing classification algorithms are designed for batch processing and required the entire dataset to be available in memory before learning and prediction can be performed. Additionally, scalability and adaptability become crucial factors when handling streaming data, as the classification model must be continuously updated as new instances arrive.

The authors propose a new algorithm called the streaming ensemble algorithm (SEA) which is a tree replacement strategy and aims to overcome the limitations of existing methods for large-scale classification. According to the proposed methodology, comparatively small sections of the data are read in blocks and used to build individual classifiers. The ensemble that is created by combining component classifiers has a set size. After the datasets are created, additional classifiers are only included if they meet a quality standard determined by how much they are expected to enhance performance. To keep the size of the ensemble constant in this situation, one of the current classifiers must be eliminated.

The authors uses some real-life data like income range for adults based on their age and educational qualification from the U.S. Census Bureau, a breast cancer data set from the Surveillance, Epidemiology, and End Results (SEER), and browsing data from 32,720 anonymous visitors. The number of classifiers is set to 25 trees, and their predictions are combined with majority voting. The evaluation of the result shows that, the improvement according to the accuracy observed in 90 percent of runs for adult data, 84 percent for SEER data, and 58 percent for anonymized data after the first 25 trees were planted.

The authors suggest that the future step would be simply to run the algorithm and compare it against other meta-learning techniques in terms of speed and accuracy.

5.3 A survey on ensemble learning for data stream classification

One of the most widely used methods of classification of data streams is the ensemble method which is integrated with ensuring the long-term accuracy of algorithms and real-time updates, such as identifying which classifiers are removed or added. The work by Gomes et al. [9] addresses multiple obstacles in data stream learning such as,

1. Conception float - that invalidates the information model).
2. Dynamic data distribution - single data and multiple data in parallel.
3. Temporal relationships - based on former on common conduct with coming behavior.
4. A big number of instances.
5. Specified labeled instances.
6. Innovative classes.
7. Option drifts.
8. Confined resources like time and memory.
9. Missing values.
10. Improper features.
11. Structure imbalance.

The authors state that existing works are based on performing the learning on static datasets and refer to the ensemble method just as an option for data stream learning. The earlier works are concentrated on either a more general problem like learning from data streams or a specific machine learning task like unsupervised learning. Another limitation is that batch learning environments is missing, as a result it is difficult to obtain a strong learner.

Data streaming requires a unique ensemble composition based on the constantly changing nature of the data. To accomplish this motive, authors propose a taxonomy of general techniques and present a classification of over 60 ensemble algorithms such as OzaBag, DWM, Streaming Ensemble Algorithm (SEA), M3, BLAST (Best Last), HSMiner (Hierarchical Stream Miner), SAE2 (Social Adaptive Ensemble 2).

As a survey, the authors do not present new experimental results or empirical evaluations of the proposed taxonomy or the reviewed algorithms. Instead, it aggregates and summarizes the findings from various studies conducted by other researchers and highlights the strengths and weaknesses of different ensemble methods based on the existing literature.

The authors expected that future studies will focus on big data stream learning, using random subspaces for high dimensionality, parallelization, ensure scalability, and focused on transforming traditional classification to new methods to face real-world scenarios, like partially supervised and imbalanced data streams.

5.4 Other related works

5.4.1 Building an Ensemble from A Single Naive Bayes Classifier in The Analysis Of Key Risk Factors For Polish State Fire Service

Identifying critical risk variables from various injuries sustained by firefighters, rescuers, children, or citizens is a challenge that the authors Nikolić et al. [26] take on for the State Fire Service of Poland.

The limitations of existing work are not explicitly mentioned by the authors.

A method to build an ensemble of Naive Bayes classifiers from a single classifier was suggested by the authors. The process divided the set of attributes utilized in the Naive Bayes classifier after producing a single classifier. To build new Naive Bayes classifiers that would form an ensemble, the attribute subsets were utilized. The ensemble was created using two methods: splitting the single classifier and looking over all of the attributes that were taken into account when making the single classifier.

The suggested approach, according to the authors, performed better than the ensemble that was created by searching over all of the qualities taken into account when developing the single classifier as well as the single classifier alone. The forward search method produced a model with an overall accuracy of 0.950 by reducing the set of attributes to a beginning set of 41 attributes. The ultimate group, assembled with the selected techniques, attained the maximum precision.

The authors propose that future studies investigate ensembles with imbalanced relationship as well as situations where different classification algorithms in an ensemble may have the same features.

5.4.2 Forest Disaster Detection Method Based on Ensemble Spatial–Spectral Genetic Algorithm

The work by Cao et al. [4] addressed the problem of creating innovative, quick, and precise change-detecting systems to detect environmental issues that damage the forest ecosystems, like powerful winds, fires in forests, and shortages.

The authors state that current methods for detecting forest changes based on genetic algorithms (GAs) solely concentrate on spectral cues, which leaves them vulnerable to noise in satellite pictures and increases the likelihood of mistakenly omitting damage to vegetation. In situations where there are few labels, supervised classification techniques—which are more expensive in terms of labelled data and human intervention—become less practical for detecting forest changes. The general-purpose usefulness of unsupervised classification algorithms for forest change detection is limited since they frequently depend on parameter adjustment and presumptions.

For the purpose of detecting forest changes, the authors put forth a method that they called Ensemble Spatial-Spectral Genetic Algorithm (E-nGA). The initial step in E-nGA is to only consider ”important” regions. This reduces the impact of non-region spots and increases the accuracy of detection. Second, based on neighbourhood factors, E-nGA uses an objective function that improves noise suppression by allowing the individual fitness to be influenced by spatial information.

The authors used a variety of data, including the 10-m spatial resolution and 13-band Sentinel-2A photos and the bitemporal Formosat-2 images with an 8-m spatial resolution and four spectral bands (red, green, blue, and near-infrared). In regard to kappa statistics, overall error (OE), and overall accuracy (OA), the E-nGA is compared with current techniques. Experiments demonstrate that the suggested approaches significantly increase convergence velocity and noise immunity, yielding the best overall accuracy (OA) of 96.54 percent, durability is low, OE is high, and the consistency kappa statistic rises initially before declining.

The authors hope to use the suggested technique in additional domains in the future, like tracking changes in ecosystem the plant matter, soil deforestation, and ocean loss.

5.4.3 Heath-PRIOR: An Intelligent Ensemble Architecture to Identify Risk Cases in Healthcare

The work by Neves et al. [22] addresses the issue of an architecture required for recommender systems that may rank emergency cases such as diseases or the likelihood of a postoperative worsening in order to help physicians and patients spot problems early on through ongoing observation.

The authors discovered that previous studies looked at disease classification and identification, but none of them prioritised cases. Previous researches are concentrated on anticipating a patient's full pathological response following neoadjuvant chemotherapy, attempting to customise emergency department waiting time prediction, developing a diabetes monitoring framework, and forecasting diseases by identifying patients who exhibit traits and symptoms that are similar to others, among other things.

The authors proposed a prediction architecture-based recommender system that generates tailored suggestions based on patient needs in home environments by utilising IoT devices. The architecture that is being proposed has five levels. Data extraction is handled by the first layer, information filtering and application of the most adherent filtering types by the second layer, prediction models for resources and user adherence by the third layer, resource identification and selection by the fourth layer with repository support, and user recommendation by the fifth layer concerning the selected resource.

According to the authors, they used two case studies wounds healing control and chronic kidney disease (CKD) to assess the proposed architecture. The authors measured the architecture using F-measures for accuracy, root mean square error (RMSE) for error and mean absolute error (MAE) for precision and recall. The outcomes of each procedure demonstrated that the dataset needed to contain precise data with no noise or missing values, and the classifications needed to be well-defined for the specific context.

The authors stated that future research should concentrate on incorporating real-time evaluations into predictions with patients undergoing treatment. This would involve recommending services or actions that could aid in the patients' healing process and utilising images in addition to raw data that could aid in diagnosis and forecast treatment-related worsening.

5.4.4 Mobile User Identification Across Domains Based on The Stacking Method In Ensemble Learning

By utilising behavioural data from mobile devices, Chen et al. [5] are addressing the issues of mobile user identification, tailored suggestions, and protecting one's privacy.

The authors note that previous work in the field of user identification has primarily used data from a single source, which has had less success. The experimental datasets used in earlier studies were in ideal conditions with few users, a high sampling rate, and good precision, so it is not representative of activities in the actual world.

The authors propose a mobile user identification framework based on the stacking method of ensemble learning. The framework focuses on user identification through geographic location, app usage data returned from mobile devices and can be easily extended to data from other domains if necessary. The method is using recall module to deal with massive mobile devices and filter out certain devices into the sorting module and then the function of the sorting module is to compare and sort the devices returned by the recall module with precise similarity.

An authentic anonymous dataset comprising behavioural data from many mobile devices was used to assess the framework; nevertheless, the dataset had a low sampling rate and accuracy. Area Under the Curve (AUC) and F1 score measures are used to gauge the framework's performance. According to the findings, the framework surpasses cutting-edge techniques, achieving an AUC of 0.917 and an F1 score of 0.839.

In order to identify appropriate base discriminators and meta-learners for mining the features of mobile devices and the correlation between user uniqueness and the dataset's sampling rate, future research should concentrate on more intricate and multi-domain datasets.

5.4.5 Empirical analysis of asymptotic ensemble learning for big data

The problem of processing and analysing enormous data quantities that exceed petabyte scales presents analytical and technological challenges, as discovered by Salloum et al. [27].

Existing research might not offer scalable large data analysis solutions, particularly in situations when approximations rather than accurate outcomes are required. Based on authors, for large-scale datasets, it might not be possible to take a random sample of the data as required by the majority of approximation techniques for big data analysis.

The authors proposed an asymptotic ensemble learning paradigm based on block-based sampling. There are five steps in the strategy as follows:

1. Data chunking - based on available computing resources.
2. Block selection - number of selected blocks also based on available computing resources.
3. Base classifiers - a decision tree derived from selected blocks.
4. Ensemble classifier - basic classifiers combined into an ensemble model via majority voting.
5. Ensemble update - the process continues until the accuracy of the ensemble model no longer increases.

Three real data sets were utilised by the authors to evaluate their suggested framework: NYC Taxi Trips (which uses trip records from New York City from 2010 to 2014), HIGGS (which classifies particle detector events into Higgs bosons and background noise), and Covertype (which predicts the type of cover that a forest would have). The experimental findings demonstrate that, in order to achieve results that are almost identical to those obtained with the whole data set, ensemble models with varying block sizes and percentages of the training set can be accurately trained.

Larger data sets analysed using statistics on a computational group should be the main area of study, according to the authors.

5.4.6 An efficient ensemble classification method based on novel classifier selection technique

Classifier selection in ensemble classification methods has not been undertaken as classifier fusion, mostly because of its higher computing cost, according to the work of Bagheri and Gao [2].

The focus of existing ensemble classification techniques has shifted from classifier selection to classifier combination, particularly when handling challenging classification issues.

To enhance classifier selection in ensemble classification, the authors proposed a novel method called Dynamic Classifier Selection by Divide and Conquer (DCS-DQ). They use a straightforward divide-and-conquer tactic, breaking down a difficult classification problem into more manageable binary sub-classification tasks.

The suggested classifier selection approach was compared by the authors with existing popular ensemble combining methods, as well as a single classifier method. Extensive experiments on 14 multi-class datasets from the University of California, Irvine (UCI) repository are used to evaluate the suggested technique. While the approach drastically reduces the execution time, it only marginally increases overall classification accuracy.

Future research, according to the authors, should concentrate on enhancing the suggested approach by incorporating a first guesser classifier that is more effective and assessing how well alternative algorithms—like decision trees or the Bayes classifier—perform when used as the base learner.

5.4.7 A Cost Sensitive Ensemble Method for Medical Prediction

The authors Li et al. [17] address the problem of a cost-sensitive process and misdiagnosis in medical disease prediction.

The authors noted that earlier research on disease classification, including survivability prediction for breast cancer, technique survival prediction for patients receiving peritoneal dialysis, and delayed renal allograft function prediction, attempted to minimise the overall classification error rate and implicitly assumed that all misclassifications were equally costly.

In order to enhance efficacy while also reducing misdiagnosis costs, the authors suggested a cost-sensitive ensemble technique based on C5.0. The process of the method is initially select the important factors from the peritoneal dialysis PD datasets. Subsequently, several cost matrices derived from the C5.0 decision tree were constructed, and the bagging technique was employed to construct the ensemble model.

The method makes use of variable selection using 25 fields from the data from the Changhai Hospital Renal Registered (CHRR). The goal of the experiment is to use an ANN model to determine the relevant features and, with the help of medical specialists, select the top 15 input attributes out of a total of 25 to use in the next model. Based on the suggested methodology, the cost ensemble method is superior in the comparison. The evaluation of the results reveals that a high sensitivity of 100 percent or a negative result is considered as a confirmed diagnosis for the patients.

Future research directions are not provided by the authors.

5.4.8 An Ensemble Approach to Learning to Rank

A single learned ranker's difficulty in reflecting the variety of user queries and its large extension error were addressed in the work of Li et al. [16].

The authors do not specifically address the shortcomings of the previous research.

The authors provide an ensemble method for ranking that uses clustering and bootstrapping to generate base rankers with a high degree of variety and accuracy. Rather than training a single base ranker, the approach trains a group of them. Each base ranker has an own prediction choice and the same degree of accuracy. By leveraging rankers to combine these base rankers, the ensemble is able to achieve a smaller generalisation error than base rankers. Furthermore, these techniques adequately take into account and make use of the flaws in the training data.

The authors ran tests on two real-world datasets OHSUMED and Gov and used two assessment metrics mean average precision (MAP) and normalized discounted cumulative gain (NDCG) to assess the performance of ranking techniques. In the result evaluation it shows that, bootstrapping gains 20 percent and clustering gains 12 percent in ranking accuracy across all measures and bootstrapping outperforms clustering on both MAP and NDCG.

The authors do not give away any information for further study.

5.4.9 An ensemble learning approach to independent component analysis

Biasing and vulnerability to regional peak values are two issues with the approximation of max prospect in Independent Component Analysis (ICA) that are addressed in the work of Choudrey et al. [7].

The authors do not explicitly mention the limitations of the existing work.

The authors suggest using Variational Bayes, also known as Ensemble Learning, as a Bayesian learning strategy for the Independent Component Analysis (ICA) model's parameters as well as latent variables. The technique suggests the factorised source distribution and the mixing matrix's subsequent distributions across the model's parameters. The method yields a more thorough representation of the data structure by approximating the posterior over the ensemble of hidden variables and parameters.

The authors focus more on the methodology and techniques used rather than the specific results or evaluation.

No information for further investigation is disclosed by the authors.

5.4.10 Ensemble Methods for Spatial Data Stream Classification

The problem of classifying data in a geographic data stream, which has become much more concentrated than classifying geographical objects in a fixed data, is the focus of work by King and Osborn [13].

According to the authors, including the unavailability of all the data at once, existing classification algorithms are ineffective for processing geographic information and incompatible with vector-based spatial data stream.

The authors provide an iterative ensemble technique for deep learning of a spatial data stream, whereby the classification system is constructed and assessed continually using three different deep neural networks to reach the necessary accuracy. The method employs a simple voting procedure to select the label that receives the greatest votes among the models; if there is a tie, a randomly selected label is applied.

The coordinate system, including rectangles represented by a tuple containing coordinates for the rectangle's top-right and bottom-left corners, is employed by the simulation dataset's authors to execute the ensemble approach and hyper-parameter optimisation test. Examining the findings revealed that the ensemble approach's average accuracy outperformed the single, unoptimized models by 2.77 percent. In addition to producing a 5.12 percent lower measure of variability at 22.03 percent, the ensemble advance also delivers a much higher accuracy of 7.89 percent variance compared to the single, unoptimized model.

In order to make non-simulated datasets suitable for the suggested strategy, the authors suggested that future research concentrate on customising them.

6 Data Requirements for Emergency Services

An accurate and timely data, including time, location, and specific conditions, enables emergency services to make informed decisions and allocate resources efficiently.

6.1 For Severe Weather

1. Current weather conditions (temperature, wind speed, precipitation).
2. Affected areas and population density.
3. Information on vulnerable populations (e.g., elderly, children).
4. Evacuation routes and shelter locations.

6.2 For Flooding

1. River gauges and water level.
2. Rainfall forecasts and accumulation data.
3. Floodplain maps.
4. Information on potential hazards in floodwaters (e.g., downed power lines, contaminants).
5. Status of flood control infrastructure (dams, levees, etc.).

6.3 For Health

1. Disease outbreak data (e.g., infectious disease transmission rates).
2. Availability of medical supplies and personnel.
3. Hospital capacity and occupancy levels.
4. Specific season of flood.
5. Communication systems for public health alerts.

6.4 For Power failures and utility disruptions

1. Data on affected areas and the extent of the power outage.

2. Status of critical infrastructure (e.g., power substations, water treatment plants).
3. Estimated time for power restoration.
4. Information on emergency generators and backup systems.
5. Outage reports from utility companies.
6. Coordination with utility companies for restoration efforts.

6.5 For Transportation Hazards

1. Information on traffic accidents and congestion.
2. Road and highway conditions, including closures and detours.
3. Data on the availability of alternative transportation options.
4. Communication systems for transportation alerts.

6.6 For Fire Explosion

1. Fire location and size.
2. Information on hazardous materials and structures in the fire's path
3. Evacuation orders and routes.

6.7 For Hazard Identification and Risk Management

1. Risk assessments for various hazards.
2. Geographic Information System (GIS) data for hazard mapping.
3. Hazard-specific response plans and protocols.
4. Data on vulnerable infrastructure and critical assets.
5. Information on emergency response resources and capabilities.
6. Real-time hazard monitoring and early warning systems.
7. Public education and awareness campaigns related to hazards.

7 References

References

[1] Shivnath Babu and Jennifer Widom. “Continuous queries over data streams”. In: *ACM Sigmod Record* 30.3 (2001), pp. 109–120.

[2] Mohammad Ali Bagheri and Qigang Gao. “An efficient ensemble classification method based on novel classifier selection technique”. In: *2nd International Conference on Web Intelligence, Mining and Semantics, WIMS ’12, Craiova, Romania, June 6-8, 2012*. Ed. by Dumitru Dan Burdescu, Rajendra Akerkar, and Costin Badica. ACM, 2012, 22:1–22:7. DOI: 10.1145/2254129.2254157. URL: https://doi.org/10.1145/2254129.2254157.

[3] Leo Breiman. “Bagging Predictors”. In: *Mach. Learn.* 24.2 (1996), pp. 123–140. DOI: 10.1007/BF00058655. URL: https://doi.org/10.1007/BF00058655.

[4] Yang Cao et al. “Forest Disaster Detection Method Based on Ensemble Spatial-Spectral Genetic Algorithm”. In: *IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens.* 15 (2022), pp. 7375–7390. DOI: 10.1109/JSTARS.2022.3199539. URL: https://doi.org/10.1109/JSTARS.2022.3199539.

[5] Xingyu Chen et al. “Mobile User Identification across Domains Based on the Stacking Method in Ensemble Learning”. In: *6th IEEE International Conference on Computer and Communication Systems, ICCCS 2021, Chengdu, China, April 23-26, 2021*. IEEE, 2021, pp. 416–423. DOI: 10.1109/ICCCS52626.2021.9449279. URL: https://doi.org/10.1109/ICCCS52626.2021.9449279.

[6] Zhida Chen et al. “Distributed Publish/Subscribe Query Processing on the Spatio-Textual Data Stream”. In: *33rd IEEE International Conference on Data Engineering, ICDE 2017, San Diego, CA, USA, April 19-22, 2017*. IEEE Computer Society, 2017, pp. 1095–1106. DOI: 10.1109/ICDE.2017.154. URL: https://doi.org/10.1109/ICDE.2017.154.

[7] R Choudrey, WD Penny, and SJ Roberts. “An ensemble learning approach to independent component analysis”. In: *Neural Networks for Signal Processing X. Proceedings of the 2000 IEEE Signal Processing Society Workshop (Cat. No. 00TH8501)*. Vol. 1. IEEE. 2000, pp. 435–444.

[8] Fatima Farag, Moustafa A. Hammad, and Reda Alhajj. “Adaptive query processing in data stream management systems under limited memory resources”. In: *Proceedings of the Third Ph.D. Workshop on Information and Knowledge Management, PIKM 2010, Toronto, Ontario, Canada, October 30, 2010*. Ed. by Anisoara Nica and

Aparna S. Varde. ACM, 2010, pp. 9–16. DOI: 10.1145/1871902.1871905. URL: https://doi.org/10.1145/1871902.1871905.

[9] Heitor Murilo Gomes et al. "A Survey on Ensemble Learning for Data Stream Classification". In: *ACM Comput. Surv.* 50.2 (2017), 23:1–23:36. DOI: 10.1145/3054925. URL: https://doi.org/10.1145/3054925.

[10] Yehia M Helmy, Doaa S El Zanfaly, and Nermin A Othman. "Prioritized query shedding technique for continuous queries over data streams". In: *2009 International Conference on Computer Engineering & Systems*. IEEE. 2009, pp. 418–422.

[11] Isam Mashhour Al Jawarneh et al. "Spatial-Aware Approximate Big Data Stream Processing". In: *2019 IEEE Global Communications Conference, GLOBECOM 2019, Waikoloa, HI, USA, December 9-13, 2019*. IEEE, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9014291. URL: https://doi.org/10.1109/GLOBECOM38437.2019.9014291.

[12] Wenbo Jin et al. "Research on Emergency Management Information Business and Information System Framework". In: *2023 IEEE 13th International Conference on Electronics Information and Emergency Communication (ICEIEC)*. IEEE. 2023, pp. 53–59.

[13] Liam King and Wendy Osborn. "Ensemble Methods for Spatial Data Stream Classification". In: *Procedia Computer Science* 224 (2023), pp. 155–162.

[14] Georg Krempl et al. "Open challenges for data stream mining research". In: *SIGKDD Explor.* 16.1 (2014), pp. 1–10. DOI: 10.1145/2674026.2674028. URL: https://doi.org/10.1145/2674026.2674028.

[15] Yong-Ju Lee et al. "Design of a scalable data stream channel for big data processing". In: *2015 17th International Conference on Advanced Communication Technology (ICACT)*. IEEE. 2015, pp. 537–540.

[16] Dong Li et al. "An Ensemble Approach to Learning to Rank". In: *Fifth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2008, 18-20 October 2008, Jinan, Shandong, China, Proceedings, Volume 2*. Ed. by Jun Ma et al. IEEE Computer Society, 2008, pp. 101–105. DOI: 10.1109/FSKD.2008.188. URL: https://doi.org/10.1109/FSKD.2008.188.

[17] Liangyuan Li et al. "A Cost Sensitive Ensemble Method for Medical Prediction". In: *First International Workshop on Database Technology and Applications, DBTA 2009, Wuhan, Hubei, China, April 25-26, 2009, Proceedings*. IEEE Computer Society, 2009, pp. 221–224. DOI: 10.1109/DBTA.2009.139. URL: https://doi.org/10.1109/DBTA.2009.139.

[18] Hyo-Sang Lim et al. "Continuous query processing in data streams using duality of data and queries". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data, Chicago, Illinois, USA, June 27-29, 2006*. Ed. by Surajit Chaudhuri, Vagelis Hristidis, and Neoklis Polyzotis. ACM, 2006, pp. 313–324. DOI: 10.1145/1142473.1142509. URL: https://doi.org/10.1145/1142473.1142509.

[19] Fiaz Majeed et al. "A burst resolution technique for data streams management in the real-time data warehouse". In: *2011 7th International Conference on Emerging Technologies*. IEEE. 2011, pp. 1–5.

[20] Alessandro Margara and Tilmann Rabl. "Definition of Data Streams". In: *Encyclopedia of Big Data Technologies*. Ed. by Sherif Sakr and Albert Y. Zomaya. Springer, 2019. DOI: 10.1007/978-3-319-63962-8_188-1. URL: https://doi.org/10.1007/978-3-319-63962-8%5C_188-1.

[21] Ammar Mohammed and Rania Kora. "A comprehensive review on ensemble deep learning: Opportunities and challenges". In: *J. King Saud Univ. Comput. Inf. Sci.* 35.2 (2023), pp. 757–774. DOI: 10.1016/J.JKSUCI.2023.01.014. URL: https://doi.org/10.1016/j.jksuci.2023.01.014.

[22] Felipe Neves et al. "Heath-PRIOR: An Intelligent Ensemble Architecture to Identify Risk Cases in Healthcare". In: *IEEE Access* 8 (2020), pp. 217150–217168. DOI: 10.1109/ACCESS.2020.3042342. URL: https://doi.org/10.1109/ACCESS.2020.3042342.

[23] Wendy Osborn and Liam King. "An Iterative Strategy for Deep Learning Classification on Spatial Data Streams". In: *iiWAS2021: The 23rd International Conference on Information Integration and Web Intelligence, Linz, Austria, 29 November 2021 - 1 December 2021*. Ed. by Eric Pardede et al. ACM, 2021, pp. 532–537. DOI: 10.1145/3487664.3487804. URL: https://doi.org/10.1145/3487664.3487804.

[24] Hong Kyu Park and Won Suk Lee. "A continuous query evaluation scheme for a detection-only query over data streams". In: *Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, 2011*. Ed. by Craig Macdonald, Iadh Ounis, and Ian Ruthven. ACM, 2011, pp. 2405–2408. DOI: 10.1145/2063576.2063978. URL: https://doi.org/10.1145/2063576.2063978.

[25] Md. Shiblee Sadik and Le Gruenwald. "Research issues in outlier detection for data streams". In: *SIGKDD Explor.* 15.1 (2013), pp. 33–40. DOI: 10.1145/2594473.2594479. URL: https://doi.org/10.1145/2594473.2594479.

[26] Salman Salloum, Joshua Zhexue Huang, and Yu-Lin He. “Empirical analysis of asymptotic ensemble learning for big data”. In: *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, BDCAT 2016, Shanghai, China, December 6-9, 2016*. Ed. by Ashiq Anjum and Xinghui Zhao. ACM, 2016, pp. 8–17. DOI: 10.1145/3006299.3006306. URL: https://doi.org/10.1145/3006299.3006306.

[27] Salman Salloum, Joshua Zhexue Huang, and Yu-Lin He. “Empirical analysis of asymptotic ensemble learning for big data”. In: *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, BDCAT 2016, Shanghai, China, December 6-9, 2016*. Ed. by Ashiq Anjum and Xinghui Zhao. ACM, 2016, pp. 8–17. DOI: 10.1145/3006299.3006306. URL: https://doi.org/10.1145/3006299.3006306.

[28] Jean Gane Sarr, Ndiouma Bame, and Aliou Boly. “Data Streams Management: Multidimensional Summary with Big Data Tools”. In: *2022 5th International Conference on Computing and Big Data (ICCBD)*. IEEE. 2022, pp. 50–55.

[29] W. Nick Street and YongSeog Kim. “A streaming ensemble algorithm (SEA) for large-scale classification”. In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA, August 26-29, 2001*. Ed. by Doheon Lee et al. ACM, 2001, pp. 377–382. DOI: 10.1145/502512.502568. URL: https://doi.org/10.1145/502512.502568.

[30] Kristine Towne et al. “Window query processing for joining data streams with relations”. In: *Proceedings of the 2007 conference of the Centre for Advanced Studies on Collaborative Research, October 22-25, 2007, Richmond Hill, Ontario, Canada*. Ed. by Bruce Spencer, Margaret-Anne D. Storey, and Darlene A. Stewart. IBM, 2007, pp. 188–202. DOI: 10.1145/1321211.1321231. URL: https://doi.org/10.1145/1321211.1321231.

[31] Wenyan Wu and Le Gruenwald. “Research issues in mining multiple data streams”. In: *Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques, StreamKDD ’10, Washington, D.C., USA, July 25, 2010*. Ed. by Margaret H. Dunham, Michael Hahsler, and Myra Spiliopoulou. ACM, 2010, pp. 56–60. DOI: 10.1145/1833280.1833288. URL: https://doi.org/10.1145/1833280.1833288.