**Research Proposal**

**Note : not to exceed 6 pages, Text to be in Arial font 11 pt, single spacing**

1. Project Title:

An In-depth Evaluation of Stream Clustering Algorithms

1. Project Objectives:

The proposed research targets at addressing the research issues relating to the modularity design and benchmark study of *data stream clustering* algorithms. Specifically, we will propose the following key mechanisms with algorithmic and technical innovations. **First**, we provide a taxonomy of existing stream clustering algorithms by carefully summarizing their key design aspects including *data structure*, *window model*, *offline refinement*, *outlier detection*, *summarization*, and *concept drift detection and adaption*. For each design aspect, we also discuss the state-of-the-art variants and reveals any potential improvements. **Second**, we assemble and implement about 20 state-of-the-art stream clustering algorithms following a modularity design, which separates key design aspects. Ultimately, each algorithm can be constructed by composing software components implemented for different design choices of each design aspect. This allows us to better comprehend the different trade-offs and performance behaviours among algorithms in a much finer granularity. **Third**, we design a comprehensive benchmark suite for benchmarking stream clustering algorithms. The benchmark must include diversity of data and application scenarios, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. **Fourth**, with the benchmark, we conduct a detailed profiling study on the different design aspects of stream clustering algorithms. We explore and further reveal how various key design aspects of the algorithm affect clustering performance respectively and the relationship between design components. Besides, the findings and optimizations proposed from this project can be applicable to future studies in terms of new algorithms and/or new hardware architectures. For example, the final evaluation results can be used as a reference in extensions of the stream clustering key design aspects in current and future stream clustering methods. We hope our work can bring the entire community to the common conscious and move forward. This is also a necessary and fundamental step to develop more efficient massive online AI-driven applications, where data stream clustering operation is often used as a preliminary and important building block.

1. Area of Research **(including innovative claims***):*

*• Overview of the proposal.*

Advanced analysis of big data streams from sensors and devices is becoming a key area of data mining research as the number of applications requiring such processing increases rapidly. Data stream clustering is one of the fundamental stream analysis operations and has been studied widely over the last decades. As streams consist of unbounded, non-stationary data objects that continuously arrive at rapid rates, their intrinsic nature requires stream clustering algorithms capable of processing data and extracting knowledge under the challenges of limited time, memory, and requires only single pass (i.e., allow nearly no revisit of the input data). In the last decades, enormous stream clustering algorithms have been proposed to satisfy these technical requirements. However, to the best of our knowledge, there is still a lack of in-depth comparison of the different approaches has been performed, and it remains unclear of key weaknesses and strengths of each of the existing stream clustering algorithms. On the one hand, many of existing algorithms are designed with implicit assumptions such as their specifically targeted application domains including network intrusion detection, sensor network monitoring and stock market analysis. Furthermore, many of them are evaluated based on different performance metrics (e.g., varying ways of determining clustering accuracy). As a result, there is no one “standard” implementation and researchers and practitioners are often left alone to decide which algorithm they should adopt in their applications. We observe that there are several design choices that have different trade-offs and performance behaviours, but there is no comprehensive experiment study to evaluate them on modern operating environment. On the other hand, many libraries and frameworks of data stream mining have been proposed such as MOA, SAMOA, StreamDM, and FlinkML. However, there are multiple limitations of existing libraries and frameworks. First, the current frameworks or packages do not cover comprehensive state-of-the-art stream clustering algorithms, partially because most of the state-of-the-art stream clustering algorithms are not open-sourced. As a result, most prior empirical evaluation works are biased especially if they rely on existing frameworks with a narrowed subset of algorithms. Second, their implementations of algorithms are fragmented, ad-hoc, and sometimes even based on different programming languages and compilers. This also leads to unfair comparisons due to the differences caused by programming languages and compilers. Moreover, they tend to use a complex language environment with dependence on others, which are hardly extensible.

To respond to this situation, the proposed research targets at addressing the research issues relating to the modularity design and benchmark study of data stream clustering algorithms. Specifically, we have the following inter-related goals:

1. **A taxonomy of stream clustering algorithm:** enormous stream clustering algorithms have been proposed and many of them are designed with implicit assumptions such as their specifically targeted application domains. This naturally leads to an open question: which one is better? Rather than comparing algorithms entirely, we provide a taxonomy of 20 state-of-the-art stream clustering algorithms by summarizing their key design aspects. For each design aspect, we discuss the state-of-the-art variants and reveals any potential improvements. This is critical to identify the entire solution space for stream clustering algorithms rather than proposing incremental improvements over existing work.
2. **A modularity design of stream clustering algorithms:** We implement stream clustering algorithms following a modularity design, which separates key design aspects. This allows us to better comprehend their different trade-offs and performance behaviours and is only made possible by our careful analysis of existing algorithms and on our taxonomy of stream clustering algorithms. Different from prior works, we implement all existing algorithms from ground up inside the same code-base to eliminate the differences among existing implementations caused by programming languages and compilers.
3. **A benchmark for stream clustering:** The benchmark must include diversity of data and workloads, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. In particular, there are two important aspects to consider in the designing of our benchmark: 1) it must cover a wide range of representative application domains; 2) it must be scalable to suit different scale of input workloads.
4. **A detailed profiling study of benchmarking stream clustering algorithms:** We conduct a detailed profiling study on the different design aspects of stream clustering algorithms. The findings and optimizations can be applicable to future studies in terms of new algorithms and/or new hardware architectures. We systematically examine the profiling results of stream clustering algorithms running the benchmark, with careful considerations on the following dimensions: 1) accuracy: there are different ways to measure the accuracy of stream clustering results and we need to consider all of them comprehensively; 2) processor architectures such as stalls in pipelines and cache/memory systems; 3) applications with different resource demands and service-level-agreements.

*• Novelty of the proposal viz-a-viz current approaches in this field.*

This proposal is timely, and we are inspired by the ubiquity of data streams in today’s emerging applications such as xxx@Zhenyu. However, to the best of our knowledge, there is still a lack of in-depth comparison of the different algorithms has been performed, and it remains unclear of key weaknesses and strengths of each of the existing stream clustering algorithms.

First, many of existing algorithms are designed with implicit assumptions such as their specifically targeted application domains including network intrusion detection, sensor network monitoring and stock market analysis. Subsequently, they implicitly assume different internal data structures, outliner detection mechanisms and many of them are evaluated based on different performance metrics (e.g., varying ways of determining clustering accuracy). As a result, there is no one “standard” implementation and researchers and practitioners are often left alone to decide which algorithm they should adopt in their applications. We observe that there are several design choices that have different trade-offs and performance behaviours, but there is no comprehensive experiment study to evaluate them on modern operating environment leaving an unpleasant literature gap.

Second, we acknowledge that many open-sourced libraries and frameworks of data stream mining have been already proposed such as MOA, SAMOA, StreamDM, and FlinkML. However, there are multiple limitations of existing libraries and frameworks. First, the current frameworks or packages do not cover comprehensive state-of-the-art stream clustering algorithms, partially because most of the state-of-the-art stream clustering algorithms are not open-sourced. As a result, most prior empirical evaluation studies are biased especially if they rely on existing frameworks with a narrowed subset of algorithms. Second, their implementations of algorithms are fragmented, ad-hoc, and sometimes even based on different programming languages and compilers. This also leads to unfair comparisons due to the differences caused by programming languages and compilers. Moreover, they tend to use a complex language environment with dependence on others, which are hardly extensible.

Third, although general-purpose SPEs that can provide fast stream computations do exist in both industry and academia (e.g., Apache Spark streaming and Apache Flink), they are not natively designed nor optimized for stream mining tasks. In reality, data scientists usually prefer to first load the entire datasets into tools like Weka, and then apply mining algorithms. Those tools are generally lacking the capability of stream processing, elastic scaling, fault tolerance mechanisms, as well as the absence of the unified API specifications and common data abstraction limits data scientists’ ability to efficiently program and maintain large machine learning pipelines. Additionally, as the pipeline gets longer and more complex, the combined workflow’s performance deteriorates due to expensive data movement across the functions and lack of cross-operation optimizations.

In summary, to the best of our knowledge, the proposed project is the first comprehensive study in benchmarking and optimizing modern stream clustering algorithms. The ultimate result of the proposal project is a highly customizable, extendable, scalable benchmark framework equipping with state-of-the-art stream clustering algorithms, a comprehensive set of application workloads, and performance evaluation tools.

1. Potential Impact of Proposed Research:

The impact of the proposed research is multi-folds.

**First**, our proposed benchmark includes the state-of-the-art stream clustering algorithms and evaluation mechanisms. It can guide users to apply the suitable algorithm in their applications appropriately including but not limited to customer click streams, telephone records, large sets of Web pages, multimedia data, financial transactions, and observational science data. For example, the event detection in social networks aims to detect events that burst into attention at certain time and space. The events can be represented as clusters of entities, and the entities are clustered according to the similarities of their belonging contexts with a stream clustering algorithm. The outcome of this research is also timely with the popularity of big-data applications and monitoring services such as sensor networks and Web. We believe that the success of this research project is able to bring economic and societal impact to not just the computing domain but also other business sectors to enable real-time data stream analysis.

**Second**, our proposed benchmark is modularized according to key design components of the prominent stream clustering algorithms. It can offer insight into the merits and limitation of existing stream clustering algorithms and future research can rely on our benchmark to further explore new design options and hence new stream clustering algorithms. The benchmark can also serve as a reference in examine the superiority of future stream clustering approaches and can hopefully bring the entire community to the common conscious and move forward.

**Third**, driven by the widely deployed 5G and IoT technology, we envision that massive stream mining will soon become a mandatory task for every AI-driven application. Considering the huge amount of data storage and transmission overhead, we simply cannot collect all the data sources before processing. To this end, the proposed research is a necessary and fundamental step before we can proceed to design a more efficient stream data mining algorithms and systems in follow up works.

1. Technical Description:
   * Methodology & Approach

We propose to research technical design, implementation, and benchmark study of data stream clustering algorithms. The major thrusts in this proposal are to: 1) design a taxonomy of stream clustering algorithm; 2) A modularity design of stream clustering algorithms; 3) A benchmark for stream clustering; and 4) A detailed profiling study of benchmarking stream clustering algorithms. They are grouped into four task packages accordingly. We will describe the details of those thrusts one by one.

**First, a taxonomy of stream clustering algorithm:** We provide a taxonomy of around 20 stream clustering algorithms by summarizing their key design aspects. For each design aspect, we discuss the state-of-the-art variants and reveals any potential improvements. This is critical to identify the entire solution space for stream clustering algorithms rather than proposing incremental improvements over existing work. As a start, we plan to include a list of widely used streaming clustering algorithms into our benchmark. Among them, the source code of *StreamKM++*, *CluStream*, *ClusTree*, *DenStream* and *D-Stream*, *BICO* and *COBWEB*, *D-Stream*, *DBSTREAM*, *BICO*, *BIRCH*, *PreDeConStream*, and *HDDStream* can be extracted from the existing frameworks including *MOA* and *subspaceMOA*. Some newly proposed but not open-sourced algorithms such as *EDMStream* and *TSF-DBSCAN* will be implemented based on the corresponding papers.

**Second, a modularity design of stream clustering algorithms:** Different from prior works, we implement all existing algorithms from ground up inside the same code-base to eliminate the differences among existing implementations caused by programming languages and compilers. Instead of implementing each algorithm separately, our framework will be modularized according to the a few key design aspects summarized in the following table.

|  |  |
| --- | --- |
| **Design Aspects** | **Design Choices** |
| Data structure for statistical summary | Feature vectors, prototype arrays, coreset trees, and data grids |
| Window model | Landmark model, sliding-window model, and damped model |
| Offline refinement strategy | With (and which) or without an offline clustering algorithm to refine the online clustering results. |
| Outlier detection mechanism | Statistics-based, density-based |
| Summarization methods | Sampling methods, histograms, wavelets, sketches, micro cluster, grid. |
| Concept drift detection mechanism | Error-based, distribution-based, multiple hypothesis test based |
| Concept drift adaption mechanism | Retrain, ensemble, adjust |

**Third, a benchmark for stream clustering:** The benchmark must include diversity of data and workloads, which is a prerequisite for any meaningful benchmark for evaluating stream clustering algorithms. As a start, we include the following workloads into our benchmark: *1) KDD-CUP'99*. One of the applications of stream clustering is network intrusion detection. It is available at the UCI repository. This data set has two weeks of raw TCP dump data for a local area network and simulates an environment with occasional attacks. It contains a total of 23 clusters, and up to 34 continuous attributes are used for clustering. Due to its large size, it has also been consistently used to assess data stream clustering algorithms; *2) Forest Cover Type.* Stream clustering can be applied to extract forest cover types. We will use the data from UCI machine learning repository, which provides observation of forest cover containing a total of 581,012 observations with 54 attributes. Each observation is labelled as one of seven forest cover types. 3) KDD-CUP’98. This data set contains 95,412 records about people who made charitable donations in response to direct mailing requests. Clustering can be used to group donors with similar donation behaviors in order to maximize the donation profit. 4) Annual Power Supply. It contains hourly power supply of an Italy electricity company which records the power from two sources: power supply from main grid and power transformed from other grids. This stream contains three year power supply records from 1995 to 1998. The concept drift in this stream is mainly driven by the issues such as the season, weather, hours of a day (e.g., morning and evening), and the differences between working days and weekend. 5) RBF. It is a non-stationary data stream with objects originated from multiple Gaussian components that move in a 2-dimensional space, overlapping in some zones. This data set is usually be applied in many streaming data algorithms to test their detection and adaptation abilities towards the different types of concept drift. The presence of noise, along with appearance/disappearance of clusters in the data set makes the scenario analysis even more challenging.

**Forth, a detailed profiling study of benchmarking stream clustering algorithms:** We conduct a detailed profiling study on the different design aspects of stream clustering algorithms. The findings and optimizations can be applicable to future studies in terms of new algorithms and/or new hardware architectures. We systematically examine the profiling results of stream clustering algorithms running the benchmark, with careful considerations on various aspects. As a start, we determine the performance metrics for stream clustering applications based on four aspects: effectiveness, efficiency, adaptability, and scalability. i) *Cluster quality* is an essential performance metric for all clustering algorithms to detect whether the cluster is effective. We use both internal and external measures for cluster quality. Specifically, internal measures consider the inner structure and properties of the clusters, such as the compactness of clusters or the distance between them. External measures are external validity criteria that compare the clusters against a ground truth. In these measures, the existing true groups of input data are compared with the data partition obtained by algorithms. Data stream clustering must process data objects quickly and incrementally within a limited time and memory to provide timely results and rapidly detect outliers with corresponding actions; ii) To measure the efficiency of algorithms, *throughput* (defined by the number of input data objects processed by algorithm per unit of time) is chosen as another performance metric; iii) As one of the requirements for stream clustering is the adaptability for handling with concept drift of data streams and data distribution changes. We *use Clusters and outlier evolution tracking* as a metric in our evaluation to represent algorithms' adaptability; iv) Lastly, we measure the scalability to data dimensions and continuously arriving objects by response time as our metric varying different dimensions and data stream arriving rates to represent respectively. Optimizing algorithms for requirements in every aspect is not possible. Performance metrics can be conflicting with each other sometimes. Thus, existing stream clustering algorithms are designed to optimize only a few performance metrics and sometimes even sacrifice their performance in other metrics. Besides those algorithmic aspects, we also compare the algorithms in terms of hardware utilizations such as CPU and memory bandwidth utilizations. We also take specific service-level-agreement requirements from real-world applications into consideration while comparing different stream clustering algorithms.

Challenges, risks & proposed mitigation plans

There are mainly three key challenges raised from the designs and implementations of our chosen algorithms, performance metrics and benchmark suite.

First, our research aims to include the most comprehensive state-of-the-art algorithms. Ther are two obstacles to achieve the goal: 1) enormous stream clustering algorithms have been proposed and can be applied in our benchmark; 2) many newly proposed stream clustering algorithms are not open sourced. We propose to first summarize design aspects of different stream clustering algorithms into a clear taxonomy based on the existing surveys and corresponding research papers. Then, we only need to implement a subset of representative algorithms that can cover different design choices in our benchmark.

Second, it is challenging to compare the results of clustering as it is commonly considered as a subjective task. Unless with ground truth, prior works are often biased towards a subset of evaluation criteria to evaluate the structure and properties of resulting clusters, e.g., the sum of squared errors (SSE) (evaluating compactness of clusters), sum squared distances SSQ, silhouette coefficient, etc. Meanwhile, validating partitions generated with non-stationary data evolved with time needs to consider the temporal criterion to evaluate the quality of the partition and their behaviour over time. To give a comprehensive comparison, we propose to implement all those evaluation metrics inside the same benchmark, and include workloads with ground truth, e.g., purity, f-measure, homogeneity, etc. In order to further capture the temporal criterion, we propose to generate the performance metric as a function of time.

Find some recent works on the similar topics and list down their name, organization, country.

@Wangxin, Zhenyu