MultiStream EvolveCluster

Christian Nordahl[†,*], Veselka Boeva[†], Håkan Grahn[†]
[†]Department of Computer Science
Blekinge Institute of Technology, Karlskrona, Sweden

Abstract

This paper proposes a novel multi-stream clustering algorithm, MultiStream EvolveCluster (MS-EC), that can be used for continuous and distributed monitoring and analysis of evolving time series phenomena. It can maintain evolving clustering solutions separately for each stream/view and consensus clustering solutions reflecting evolving interrelations among the streams. Each stream behavior can be analyzed by different clustering techniques using a distance measure and data granularity that is specially selected for it. The properties of the MultiStream EvolveCluster algorithm are studied and evaluated with respect to different consensus clustering techniques, distance measures, and cluster evaluation measures in synthetic and real-world smart building datasets. Our evaluation results show a stable algorithm performance in synthetic data scenarios. In the case of real-world data, the algorithm behavior demonstrates sensitivity to the individual streams' data quality and the used consensus clustering technique.

Keywords: evolve clustering, data stream mining, consensus clustering

1. Introduction

Streaming data has been growing in popularity since its inception. Today's data sources are increasingly becoming streams, partly due to the ever-growing Internet of Things (IoT) expansion. Streaming data presents challenges that are not typical in conventional datasets. Streams are faster and potentially infinite in size, which has put more conventional machine learning and data mining techniques on the sidelines. With the right type of devices, any system could be continuously monitored by small IoT devices.

To handle the enormous amount of non-labeled data sensors like these produce, algorithms have to be carefully designed to cope with these new challenges. Clustering revolves around identifying groupings of data, providing insight into the underlying structure of the data. For streaming data, clustering usually is performed with an online incremental component that uses summarization techniques to handle the tremendous amount of data, in conjunction with an offline anytime component that produces a final clustering solution.

Considering each stream as a specific view of the system, we can let each stream be modeled and analyzed individually and use the individual models to create a global model that explains the entire system using reduced communication. The communication size of sending the resulting clustering solution is much smaller than sending the raw data directly to a centralized location. This allows us both to distribute the computations and have more flexibility for the individual streams. Each stream could use its own unique combination of clustering algorithms, dissimilarity measures, and parameters. For instance, many households today contain real-time sensors for their electricity consumption. Having sensors for water, gas, outdoor and indoor climate consumption, etc., would give an overall and more complete view of the household's total consumption. The data from individual sensors could be analyzed to study and understand their specific patterns, but combining them gives an overall view of the household residents' consumption behavior.

This paper proposes a novel distributed multi-stream (multi-view) clustering algorithm entitled MultiStream EvolveCluster (MS-EC). A multi-source extension of our previously

[*]christian.nordahl@bth.se

introduced algorithm EvolveCluster [1]. We design and conduct several experiments to investigate how the MS-EC algorithms perform, using synthetic and real-world data, multiple consensus clustering algorithms, and against a baseline algorithm. The obtained results show that MS-EC performs better than the baseline algorithm with respect to used evaluation measures on both synthetic and real-world data. MS-EC copes with the dimensionality curse by providing a distributed data analysis technique that generates understandable and easily interpretable results that facilitate the practitioners in monitoring and analyzing the individual data streams' behavioral patterns and the whole system's behavior.

2. Background

In smart home monitoring applications, numerous sensors capture events that occur. A conventional approach is to send all the data to a single computing device and then do data mining on the entire data gathered from all the sensors. This would require a system with considerable computational resources to focus solely on analyzing all that data. Applying data analysis separately to each stream would allow for continuous monitoring of the behavior of the individual streams and fully capture the ongoing behavior of the smart home.

The individual clustering solutions can then be sent to a centralized location for further analysis. There, we can merge them into an overall clustering solution that enables us to identify inter-cluster relationships and dependencies between the individual behaviors of the clustering solutions. This general procedure enables distributed (potentially in real-time) computing and analysis of individual sensor streams and then performing an overall analysis at a higher level of the general behavior. The individual clustering solutions would also increase the transparency of the importance of the separate streams for the overall solution, as it is easier to understand and interpret the created models with fewer data sources. Finally, it would allow different data distributions to be clustered simultaneously and separately.

The proposed MS-EC algorithm resembles *feature-distributed ensemble clustering* and *multi-view clustering* principles. Therefore, we introduce these clustering paradigms and present three types of merging procedures in ensemble clustering, commonly referred to as consensus clustering. The latter procedures are used in our experiments.

Multi-view clustering considers the data as multiple aspects of the data and its system [2]. Conventional clustering, on the other hand, only interprets the data from a single view. Dividing the data into subsets and combining certain features gives a unique system view. The views can be completely distinct or multiple views containing the same feature [3, 4]. For example, the system can have operational, contextual, and performance views, where each view has a subset of the features present in the data.

2.1. Feature-Distributed Ensemble Clustering

Ensemble learning uses multiple algorithms or models and combine their individual results to determine the end result. The idea is to use several weaker learners to create one powerful one [5]. In the case of ensemble clustering, several clustering solutions are produced and are combined to create a global clustering solution [6]. One can either use many clustering algorithms, the same algorithm with different tunings, or divide the data to create different clustering models. This paper focuses on the latter, i.e., feature-distributed ensemble learning [7]. To create the global clustering solution, consensus clustering is performed. Below, we present the three algorithms we use to produce consensus clustering in our experiments.

Majority Vote: An intuitive and simple consensus clustering is majority voting [8]. Each clustering solution's cluster labels are used to vote about which pairs of objects belong together. If a threshold is reached for a pair of objects, they are paired together in the global clustering solution. To perform the majority vote, we identify the pairwise occurrences of the

data objects in each clustering solution. A co-occurrence matrix, C_o, of size $n \times n$, where n is the number of data objects in the clustering solutions, is created and initialized to 0. For each pair of objects (i,j) in each clustering solution, we increment the corresponding cell in the matrix, $C_o(i,j)$ with $1/m$, where m is the number of clustering solutions.

All pairs with co-occurrence scores larger than the threshold indicate that those two items belong in the same cluster in the consensus clustering solution. If two items are not in any cluster in the consensus clustering, a new cluster is created, and the two items are added. If one of the items belongs to a cluster and the other does not, the other item is added to that cluster. If both items belong to different clusters, those clusters are merged. Finally, if an item has no co-occurrence score over 0.5 with any other item, it is added to a singleton cluster.

Hypergraph Partitioning: Instead of creating a co-occurrence matrix, we can consider each clustering solution's labels as hyperedges in a hypergraph. A hypergraph is a generalized graph where edges can connect more than two nodes, creating more of an area rather than an edge. Partitioning this hypergraph into distinct groups requires cutting edges so that no edge that overlaps groups exist. We can interpret each clustering solution's labels as hyperedges to create a hypergraph out of clustering solutions. Each vertex in the graph is a data instance from the clustering solutions. The vertices are connected via the hyperedges from the cluster labels [7]. This hypergraph is partitioned by cutting hyperedges that cause overlap between the groups of vertices. The resulting groups of data are the consensus clustering solution.

There are two common ways to define the stopping criterion for the partitioning procedure: k-way partitioning and k-cuts. The k-way partitioning identifies the best cuts that result in k clusters of vertices, while k-cuts partitioning identifies and performs the k cuts that provide the best distinction between the groups.

Markov Clustering: A middle ground between hypergraphs and co-occurrence matrices is simple graphs. To create a graph from the cluster labels, we can create a matrix by the same procedure as the one used for the co-occurrence matrix. However, each cell represents an edge in the graph. A lower score shows an edge with low weight, i.e., a closer relationship between the data objects. Any graph clustering algorithm could be used to identify the consensus clusters. In this study, we have used Markov Clustering (MC) [9], a graph clustering algorithm that performs a series of inflations and expansions on the graph matrix to cluster the vertices.

3. Related Work

With the world's ever-increasing number of devices and sensors, we are acquiring many possible data streams. Clustering data streams has been extensively studied recently [10–13]. Most research, though, focuses on only a single data stream. In contrast, the algorithm proposed in this paper is intended for multiple streams. Multi-stream approaches can broadly be divided into two categories. The first category covers the cases where multiple streams themselves are clustered based on their characteristics [14–17]. A typical application for these is, e.g., electricity consumption, where user profiles (streams) are clustered together to identify consumer behavior. The second category comprises approaches that cluster the arriving objects themselves. These objects are either summarized (micro-clustered) on a stream level and the summaries are sent to a centralized location for a global clustering solution [18, 19], or the streams are all integrated into a single solution directly [20]. The second category focuses more on monitoring the system's behavior and health.

The MS-EC algorithm resembles the approaches from the second category but with some caveats. Each stream represents the overall system from its perspective, and we let each stream do that by having a clustering module for each stream. The resulting clustering labels are then sent to the centralized server to perform consensus clustering. The cluster labels determine how the data points relate, seen from each stream's perspective. A procedure in

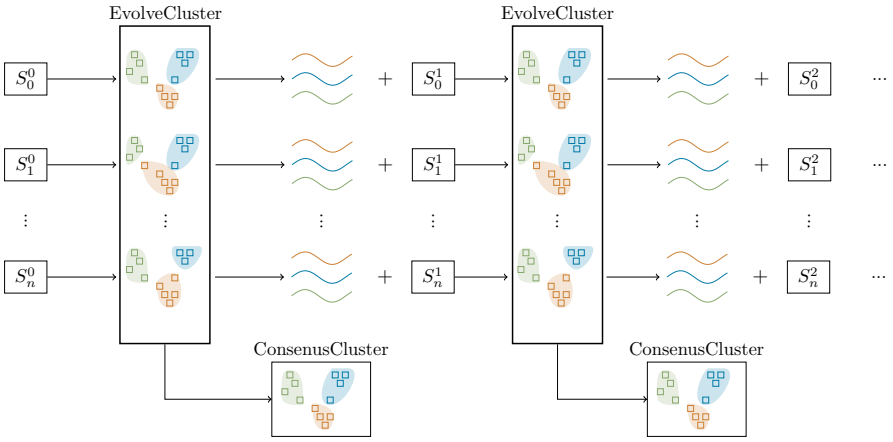

Figure 1. MultiStreamEvolveClusters general procedure while clustering a set of data streams. S is a set of n streams, i.e. $S = \{S_0,...,S_n\}$. Each stream is divided into equal-sized segments, so S_i^j is the jth segment of stream S_i. Each stream is clustered separately, and after a segment is clustered, the clusterings are combined to determine a consensus.

line with feature-distributed clustering [7] and multi-view clustering [3]. This provides an abstract view of the system while allowing each stream to be monitored individually.

None of the previously mentioned methods focus on clustering multiple streams individually, using the resulting clustering solutions to fuse. We identified one related study, multi-view stream (MVStream) clustering [21], that approaches multi-stream clustering similarly to our proposed MS-EC algorithm. Our MS-EC algorithm is simpler, more explainable, requires less communication, and enables individual streams to be analyzed in addition to the whole system.

4. MultiStream EvolveCluster

The MS-EC algorithm consists of two separate modules, *EvolveCluster* and *ConsensusCluster*. The EvolveCluster module is an online procedure used individually by each stream to continuously maintain the partitioning of its data in clusters of similar objects. Conversely, the ConsensusCluster module is an offline procedure that can be applied at each data segment to analyze all the streams' current clustering solutions. The ConsensusCluster module is agnostic concerning the consensus clustering technique, i.e., different techniques can be implemented in this module, and the proper one can be selected especially for the task and data under consideration. We first describe these two modules and then formally describe the MS-EC algorithm's general procedure. The latter is also illustrated in Figure 1.

4.1. MS-EC Modules

EvolveCluster [1] is a sequential data stream clustering algorithm capable of modeling consistent and changing behaviors in a data stream. It builds upon dividing a stream into specified segments, or windows, and clusters them sequentially as they arrive. When a segment is clustered, it is initially partitioned using prior knowledge from the previous segment. Namely, it is seeded with the cluster centroids identified in the last data segment. By incorporating this prior knowledge, the algorithm assumes the incoming data to be relatively stable. When changes occur in the stream, there is an explicit cluster-splitting criterion, based on a user-defined threshold τ, to make the clustering solution adaptable to newly appeared concepts.

Three different variants are implemented and studied for the ConsensusCluster module of our proposed algorithm. Namely, in our experiments, we use the majority vote and two

graph-based alternative algorithms (see Section 2). The ConsensusCluster module receives cluster labels from each stream which are then integrated by applying a consensus clustering technique selected from several implemented in the module.

4.2. MS-EC General procedure

The general working mechanism of the MultiStream EvolveCluster algorithm is illustrated in Figure 1. First, each data stream, or group of streams, uses its own EvolveCluster module to cluster its data. After all the streams' EvolveCluster modules have clustered a segment, the clustering solutions are sent to the server. The server performs a consensus clustering, using a pre-selected consensus clustering technique that best fits the problem under interest. The server component is an offline procedure used to analyze and interpret the system's behavioral patterns more abstractly. This also facilitates a better understanding of the system's behavior by providing an overview of its performance in terms of clustering quality and the stream sensors' performance.

Let us formally explain the two modules of MS-EC algorithm. Assume a set of streams $S = \{S_0, S_1, S_2, ..., S_k\}$ is given, where each stream is its unique source of information. However, each stream follows the same data collection principle and has aligned timestamps for their data. We employ a landmark window setting on our streams, creating a set of segments for each stream $S_i = \{S_i^0, S_i^1, S_i^2, ..., S_i^j\}$, where $i \in \{0,1,...,k\}$ and $j \in \{0,1,...,\infty\}$.

The corresponding sets of clustering solutions $C = \{C_0, C_1, C_2, ..., C_k\}$ are equally divided into segments such that $C_i = \{C_i^0, C_i^1, C_i^2, ..., C_i^j\}$, where $i \in \{0,1,...,k\}$ and $j \in \{0,1,...,\infty\}$. Each stream is individually clustered using its EvolveCluster module. When clustering stream segment S_i^j, centroids from the previous segments' clustering solution C_i^{j-1} are used as initialization centroids. These initial centroids are removed after the initial partitioning, after which the inner clustering algorithm refines the solution with merge and split operations.

After segment S^j is clustered, creating the clustering solutions $C_0^j, C_1^j, ..., C_k^j$, the cluster labels is sent to a server. This server has a ConsensusCluster module which is used to obtain a consensus clustering solution, $CC = \{CC_0, CC_1, CC_2, ..., CC_j\}$, where $j \in \{0,1,...,\infty\}$. The resulting consensus clustering solution, CC, represents the final clusters according to the ensemble knowledge gained from all the individual streams.

MS-EC's majority of computation lies within the EvolveCluster module. The overhead added by the ConsensusCluster module is dependent on the chosen algorithm but is significantly smaller. A thorough analysis of EvolveCluster is available in our previous work [1].

5. Empirical Evaluation

This section provides the results obtained from our empirical evaluation of the proposed MS-EC algorithm. We first introduce the dissimilarity measures, evaluation measures, and software used and conclude by thoroughly explaining how our experiments have been designed.

Dissimilarity Measures: We investigated the use of Euclidean distance, dynamic time warping, Canberra distance, Minkowski distance, Chebyshev distance, and Mahalanobis distance by performing initial experiments on our datasets. These experiments showed that Euclidean distance performed best for the synthetic data, and for the real-world data, a mix between Euclidean distance, Minkowski distance, and Canberra distance was used.

Evaluation Measures: We have used multiple cluster validation indexes (CVIs) to evaluate different aspects of the clustering solutions produced by the MS-EC algorithm on the synthetic datasets. Namely, we have applied the following measures to assess the quality of the generated clustering solutions: F_1 [22], Silhouette Index (SI) [23], Adjusted Rand Index (ARI) [24], and Adjusted Mutual Information Index (AMI) [25]. These measures have been

used to evaluate the individual streams' clustering and consensus clustering solutions. To evaluate the clustering solutions on the real-world dataset, in addition to the SI, we apply the average intra-cluster distance (IC-av) [26]. These measures are also used to study and identify the optimal cluster number for each stream and baseline in real-world data.

Software: To implement the MC [9] variant of MS-EC we used the publicly available implementation[1] and for HGP we chose KaHyPar [27] to perform the partitioning[2]. The MC algorithm has one parameter that controls the cluster granularity, depicted as I, which is required for basic usage and was empirically chosen and set to 5.0 throughout our experiments. Finally, for KaHyPar we set the k parameter in k-way partitioning to the same as the maximum number of clusters available of any stream in that segment.

5.1. **Experiments**

We have divided our experiments into three parts. The first experiment uses synthetic data to study the algorithm properties in a small control scenario. Namely, it exhibits a toy example of clustering 3 streams containing 2 features each. The second experiment studies the scalability of the MS-EC algorithm in a more complex scenario, again based on the synthetics data. Namely, it consists of clustering 12 8-dimensional streams. The final experiment is the clustering of the real-world dataset. The data comes from the smart building domain, consisting of individual electricity, water, and gas consumption streams with additional weather data. In this experiment, we study the algorithm sensitivity with respect to data quality in the different streams, i.e., how the degradation in performance of some stream clustering solutions affects the overall consensus clustering. All experiments' source code and results are available online[3].

5.1.1. **Part 1: 3 streams of 2-dimensional data**

In this experiment, we investigate and compare the different implemented variants (based on different consensus clustering techniques) of the proposed MS-EC algorithm against a baseline version of EvolveCluster that analyzes all the data gathered from all the streams together. The data streams were generated by a Radial Basis Function Generator (RBFGenerator), creating spherical clusters with a constant movement to simulate concept drift. The RBFGenerator, mentioned in [1], is based on the versions available in MOA and scikit-multiflow.

Each of the 3 streams contains 2 features in the [0,1] range and 10,000 data points divided into 5 clusters. The baseline algorithm is fed the combined version of the three streams, a dataset consisting of 6 features and 10,000 data points. As both MS-EC and EvolveCluster algorithms rely on the initial data segment, S^1, to be clustered beforehand, we provided the ground truth labels from S^1 as cluster assignments to both algorithms and the corresponding streams. The medoid for each cluster was identified by calculating the centremost point in each cluster. The τ values for all EvolveCluster modules and the baseline were set to 0.1.

5.1.2. **Part 2: 12 streams of 8-dimensional data**

The streams were generated with the same RBFGenerator as in Part 1 but with 12 streams of 8 features each and 10,000 data points divided into 5 clusters. The combined dataset provided for the baseline algorithm consisted of 96 features and 10,000 data points. This experiment is intended to investigate the scalability of the algorithms and is evaluated identically to Part 1. The τ values for all EvolveCluster modules and the baseline were set to 0.1.

---

[1] https://micans.org/mcl/
[2] https://kahypar.org
[3] https://github.com/christiannordahl/MultiStream-EvolveCluster

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
S_0	F_1	1	1	1	1	0.99	0.81	0.99	1	1	1
	SI	0.84	0.89	0.90	0.85	0.77	0.58	0.78	0.80	0.81	0.83
	ARI	1	1	1	1	0.99	0.63	0.99	1	1	1
	AMI	1	1	1	1	0.99	0.77	0.99	1	1	1
	#clust	5	5	5	5	5	5	5	5	5	5
S_1	F_1	1	1	1	1	1	1	1	1	1	1
	SI	0.90	0.88	0.82	0.83	0.87	0.86	0.80	0.76	0.80	0.76
	ARI	1	1	1	1	1	1	1	1	1	1
	AMI	1	1	1	1	1	1	1	1	1	1
	#clust	5	5	5	5	5	5	5	5	5	5
S_2	F_1	1	0.97	0.98	0.99	0.99	0.91	1	1	1	0.99
	SI	0.59	0.73	0.76	0.75	0.73	0.67	0.75	0.83	0.87	0.73
	ARI	1	0.98	0.98	0.99	0.99	0.84	1	1	1	0.96
	AMI	1	0.97	0.97	0.99	0.99	0.86	1	1	1	0.96
	#clust	5	5	5	5	5	5	5	5	5	5

Table 1. Evaluation scores generated by four CVIs, and the number of clusters, on the individual clustering solutions generated for the first 10 data segments of 3 two-dimensional streams by the EvolveCluster modules.

5.1.3. **Part 3: The Almanac of Minutely Power Dataset**

The real-world dataset used in this experiment is the Almanac of Minutely Powered dataset (AMPds) version 2 [28]. It consists of minute measurements of a single residential household's electricity, water, and gas consumption in Canada from April 2012 to May 2014. Additionally, hourly weather data was collected from a nearby weather station. The streams were segmented into 12 segments, each containing 2 calendar months. They ranged from 59 to 62 days in each segment. Each stream was also normalized to the [0,1] range.

All streams were summarized, except in the weather stream, where the mean was calculated, into hourly measurements and divided into daily profiles. We investigated the granularity of the daily profiles in the initial segment S^0 by clustering 1, 2, 3, 4, 6, and 8-hour measurements, to determine an acceptable granularity for use with clustering. The final granularities chosen were 4 hours for the electricity stream (S_e), 8 hours for the water stream (S_{wa}), 6 hours for the gas stream (S_g), 4 hours for the weather stream (S_{we}), and 4 hours for the baseline.

The τ values for each EC module was set to 0.3 for S_e, 0.37 for S_{wa}, 0.07 for S_g, and 0.1 for S_{we}. The τ value for the baseline algorithm was set to 0.07. Euclidean distance was used for S_e, S_{wa}, S_g, Canberra distance for S_{we}, and Minkowski distance for the baseline.

6. **Results, Analysis, and Discussion**

6.1. **Part 1 results: 3 streams of 2-dimensional data**

In Table 1, we can see that each EvolveCluster module follows suit and produces good clustering solutions overall in each stream. However, the EvolveCluster modules in stream S_0 and stream S_2 produce lower-quality clustering solutions for segment S^6, where we can see a drop in all CVIs. This drop occurs due to two clusters drifting on top of each other before separating away from each other again in the next segment.

In Table 2, we can see that when consensus clustering occurs, the above-mentioned flaw is rectified using the MV version of the MS-EC algorithm. We observe a perfect score for all segments' clustering solutions when the MV is used, but both the HGP and MC create generally lower-scoring clustering solutions. This is especially true for the HGP algorithm, the lowest-performing approach for consensus clustering in the studied context. Interestingly,

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
MV	F_1	1	1	1	1	1	1	1	1	1	1
	SI	0.88	0.86	0.88	0.88	0.85	0.82	0.80	0.82	0.87	0.88
	ARI	1	1	1	1	1	1	1	1	1	1
	AMI	1	1	1	1	1	1	1	1	1	1
	#clust	5	5	5	5	5	5	5	5	5	5
HGP	F_1	0.81	0.61	0.79	0.77	0.63	0.74	0.67	0.66	0.61	0.67
	SI	0.48	0.09	0.43	0.45	0.10	0.36	0.18	0.11	0.09	0.19
	ARI	0.69	0.38	0.62	0.63	0.46	0.62	0.47	0.50	0.38	0.45
	AMI	0.79	0.51	0.72	0.74	0.63	0.74	0.60	0.62	0.52	0.63
	#clust	5	5	5	5	5	5	5	5	5	5
MC	F_1	1	0.96	0.96	0.96	0.84	0.83	0.94	1	1	0.94
	SI	0.88	0.67	0.73	0.68	0.32	0.35	0.24	0.82	0.87	0.57
	ARI	1	0.91	0.91	0.75	0.54	0.56	0.68	1	1	0.69
	AMI	1	0.94	0.94	0.89	0.73	0.75	0.86	1	1	0.87
	#clust	5	4	4	4	3	3	4	5	5	4
Base	F_1	1	0.24	0.23	0.25	0.25	0.24	0.25	0.24	0.24	0.24
	SI	0.88	-0.043	-0.047	-0.064	-0.050	-0.034	-0.063	-0.061	-0.070	-0.032
	ARI	1	-0.002	-0.004	-0.002	0.002	0.002	-0.001	0.005	-0.002	0.001
	AMI	1	-0.0006	0.0008	0.0005	0.002	0.0006	0.003	0.001	0.0006	-0.002
	#clust	5	5	5	5	5	5	5	5	5	5

Table 2. Comparison of the performance of the three implemented variants of the MS-EC algorithm against that of the EvolveCluster multi-source baseline version on 3 two-dimensional data streams with respect to four different CVIs and the number of clusters.

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}
min	0.86	0.75	0.98	0.93	0.94	0.80	0.82	0.86	0.97	0.97	0.96	1
max	0.99	0.99	0.99	1	0.99	0.96	0.99	1	0.98	0.98	0.99	1
mean	0.97	0.94	0.98	0.99	0.97	0.94	0.96	0.98	0.97	0.97	0.98	1
median	0.98	0.98	0.99	1	0.95	0.96	0.99	1	0.97	0.97	0.98	1
std dev	0.04	0.081	0.0019	0.023	0.020	0.05	0.053	0.044	0.0039	0.0039	0.0084	0

Table 3. Statistics for the F_1 measure calculated for each segment on twelve 8-dimensional streams. S^j stands for the jth segment of the streams.

the MC algorithm fluctuates in its number of clusters between the segments. The performance scores of the EvolveCluster multi-source version, which we use as a baseline model are also given in Table 2. It is clearly a trickier problem to cluster the combined product of the streams rather than clustering them separately first.

6.2. **Part 2 results: 12 streams of 8-dimensional data**

We have increased the number of streams and features for the second experiment to analyze our proposed algorithm's scalability. Table 3 provides the summary statistics of the F_1 measure on each stream on the 8-dimensional 12-stream dataset. Overall, we see a high score for all individual modules and segments. Only a few segments in streams S_0, S_1, S_5, and S_6 produce scores lower than 0.9, with the median values staying well above 0.95.

Turning our attention to consensus clustering, we can see in Table 4 that the trend of more straightforward solutions performing better is also apparent here. Namely, the MV approach to consensus clustering shows a generally high score with respect to all used CVIs, with perfect scores of F_1, ARI, and AMI. The MC algorithm did not produce any good results, providing only 1 cluster for all its segments. We have investigated this further, and it turns out that when

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}
MV	F_1	1	1	1	1	1	1	1	1	1	1
	SI	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96
	ARI	1	1	1	1	1	1	1	1	1	1
	AMI	1	1	1	1	1	1	1	1	1	1
	#clust	5	5	5	5	5	5	5	5	5	5
HGP	F_1	0.58	0.71	0.71	0.71	0.70	0.71	0.70	0.70	0.71	0.71
	SI	0.15	0.26	0.22	0.22	0.19	0.15	0.19	0.18	0.26	0.23
	ARI	0.36	0.56	0.58	0.58	0.55	0.56	0.56	0.56	0.59	0.57
	AMI	0.50	0.72	0.72	0.76	0.72	0.70	0.73	0.73	0.76	0.73
	#clust	5	5	5	5	5	5	5	5	5	5
MC	F_1	1	0.55	0.54	0.58	0.58	0.56	0.59	0.58	0.54	0.55
	SI	0.96	-	-	-	-	-	-	-	-	-
	ARI	1	0	0	0	0	0	0	0	0	0
	AMI	1	0	0	0	0	0	0	0	0	0
	#clust	5	1	1	1	1	1	1	1	1	1
Base	F_1	1	0.24	0.23	0.24	0.24	0.24	0.24	0.23	0.23	0.23
	SI	0.96	-0.10	-0.097	-0.077	-0.079	-0.057	-0.12	-0.046	-0.034	-0.062
	ARI	1	-0.005	-0.002	0.008	-0.002	-0.006	-0.002	0	0	-0.004
	AMI	1	0.0008	-0.004	0.004	0.001	-0.0008	0.0004	0.003	-0.001	-0.001
	#clust	5	5	5	5	5	5	5	5	5	5

Table 4. Comparison of the performance of the three variants of the MS-EC algorithm against that of the EvolveCluster multi-source version on 12 eight-dimensional data streams with respect to four different CVIs and the number of clusters.

we add the 6th stream, and onwards, the produced clustering solution quickly decayed into this behavior. Both the HGP and multi-source version of EvolveCluster (the baseline) have performed similarly to what they did in the first experiment. The baseline algorithm's number of clusters has remained the same as the ground truth, but all CVIs show that the produced clustering solutions are not on par with either the MV or HGP versions of the MS-EC algorithm.

6.3. **Part 3 results: AMPds2**

In this experiment, we study the algorithm performance in a real-world data context. The sensitivity of the ConsensusCluster module is investigated in several experiments to reveal how the degradation in performance of some of the individual stream clustering solutions affects the overall consensus clustering.

In Table 5, one can see the SI and IC-av scores produced by EvolveCluster modules on the first 12 segments of the four data streams. It can be observed that the EvolveCluster has not extracted any meaningful structure for the weather stream since, from the fifth segment forward, it produces just a single cluster. This indicates that this view does not significantly impact the system behavior, and maybe it should be excluded from the data input of the ConsensusCluster module. The latter is also confirmed by the results generated by the ConsensusCluster on the integration of all four streams (see Table 6). Namely, we believe this affects the consensus clustering solutions produced by MV and MC. These algorithms seem to be sensitive to the quality of the clustering structure of the studied streams. A similar trend can also be noticed in the results generated by the multi-source version of EvolveCluster, the baseline.

To investigate this further, we have conducted multiple experiments studying different combinations of the four streams. In the paper, we have only included the results generated on water, electricity, and gas streams presented in Table 7. We can see that when excluding the weather stream, HGP's behavior remains the same while the performance of the MV and MC

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}	S^{11}	S^{12}
S_e	IC-av	21.8	21.6	25.1	17.5	23.7	21	21.5	18.4	17.6	23.6	24.3	21.9
	SI	0.77	0.79	0.80	0.76	0.75	0.75	0.76	0.81	0.79	0.77	0.73	0.75
	#clust	6	6	6	6	6	6	6	4	4	5	5	5
S_{wa}	IC-av	19.4	19.3	23.5	19.1	18.9	21.8	22.6	21.8	16.2	21.7	18.6	21.5
	SI	0.96	0.93	0.89	0.92	0.91	0.91	0.91	0.87	0.90	0.93	0.92	0.93
	#clust	5	5	4	3	3	4	5	5	5	5	5	5
S_g	IC-av	21.67	21.7	20.7	19.8	20.3	21.4	19.5	11.9	18.2	13.5	16.5	11.6
	SI	0.92	0.91	0.93	0.89	0.74	0.81	0.88	0.89	0.92	0.83	0.72	0.72
	#clust	5	5	4	6	6	5	4	2	3	2	2	2
S_{we}	IC-av	15.6	17.9	10.8	16.6	7.6	4.6	4.2	3.2	3.7	6.3	10.6	9.8
	SI	0.58	0.83	0.84	0.80	-	-	-	-	-	-	-	-
	#clust	6	3	3	2	1	1	1	1	1	1	1	1

Table 5. Evaluation scores on the individual clustering solutions generated for the first 12 data segments of 4 real-world data streams by the EvolveCluster modules.

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}	S^{11}	S^{12}
MV	IC-av	9.2	9.5	9.3	10.9	12.8	12.2	10.8	10.1	9.1	11.3	12.7	12.0
	SI	-	-	-	-	-	-	-	-	-	-	-	-
	#clust	1	1	1	1	1	1	1	1	1	1	1	1
HGP	IC-av	9.0	9.9	9.8	8.9	12.2	10.6	9.3	8.2	9	8.6	13.3	11.5
	SI	-0.03	-0.01	-0.08	-0.06	0	-0.03	-0.07	-0.04	-0.06	-0.06	0.01	-0.06
	#clust	6	6	6	6	6	6	6	5	5	5	5	5
MC	IC-av	9.3	9.5	9.3	10.9	12.8	12.2	10.8	10.1	9.1	11.3	12.7	12.0
	SI	0.03	-	-	-	-	-	-	-	-	-	-	-
	#clust	2	1	1	1	1	1	1	1	1	1	1	1
Base	IC-av	8.9	9.3	10.3	9.4	12.5	11.5	11.1	9.2	8.7	9.7	13.1	11.9
	SI	0.15	0.16	0.10	0.19	0.23	0.14	0.14	0.11	0.15	0.20	0.25	0.24
	#clust	7	8	8	4	3	3	3	2	3	2	2	3

Table 6. Comparison of the performance of the three variants of the MS-EC algorithm against that of the EvolveCluster multi-source version on 4 real-world data streams.

variants slightly improves. An additional threshold for the majority vote, 80% beside the normal 50%, was also included, and we see an apparent increase in the number of clusters. In this scenario, many singleton clusters are created as many of the pairwise co-occurrences fall below the threshold, as now all three streams have to agree on which clusters to place the elements.

It is also interesting to notice that the trend observed in the synthetic data experiments is not confirmed in the real-world data. Namely, the best-performing consensus clustering technique is not the simplest, i.e., the MV, but the HPG algorithm. This once more supports that evaluating novel ML algorithms only on synthetic datasets cannot provide an objective view of the algorithms' behavior. In many cases, this is not even confirmed in a real-world data context.

One clear aspect of the MS-EC algorithm is that the ConsensusCluster module's use of only cluster labels greatly relies on the individual streams' clustering solutions. For instance, in the synthetic data experiments, we have identified that when the individual stream clustering solutions are reasonable, the resulting consensus clustering solution tends to be of good quality. While in the real-world data experiments, the streams have been harder to cluster, leading to poor-quality clustering solutions.

The MS-EC algorithm provides a distributed clustering approach that limits communication size and keeps the data's privacy and integrity. Many stream data mining algorithms do

		S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8	S^9	S^{10}	S^{11}	S^{12}
MV_{50}	IC-av	19.3	19.2	19.4	24.2	28.8	26.6	22	21.6	18.2	24.2	27.6	25.3
	SI	-0.05	-	-	-	-	-	0.01	-	-	-	-	-
	#clust	3	1	1	1	1	1	3	1	1	1	1	1
MV_{80}	IC-av	12.2	11.9	16	10.1	16.2	10.8	11	12.7	10.5	10.8	19.0	14.4
	SI	-0.10	-0.10	-0.17	-0.11	-0.09	-0.07	-0.11	-0.10	-0.11	-0.12	-0.10	-0.14
	#clust	28	31	26	32	29	37	30	23	23	29	22	26
HGP	IC-av	19.1	19.7	21.3	17.6	27.9	24.7	20.4	17.6	16.4	19	26.2	25.3
	SI	-0.03	-0.05	-0.06	-0.05	-0.05	-0.02	-0.08	-0.07	-0.07	-0.01	-0	-0.04
	#clust	6	6	6	6	6	6	6	5	5	5	5	5
MC	IC-av	18.7	18.7	21.3	24.2	28.8	22.3	22.7	21.6	16.5	20.4	27.6	25.6
	SI	-0.06	-0.06	-0.16	-	-	-0.11	-	-	-0.03	0.11	-	0.10
	#clust	7	7	8	1	1	11	1	1	3	4	1	2

Table 7. Comparison of the performance of the three variants of the MS-EC algorithm on three real-world data streams. The number of clusters produced in each segment is also depicted. The MV variant is present two times with thresholds of 50% and 80%.

not respect the same integrity and privacy perseverance MS-EC does; either raw data is sent directly or summary statistics to the centralized location, which then performs clustering. This is an essential aspect for the current and future computing landscapes, where cloud computing and 5G networks are still growing, to keep the privacy and integrity of the data intact.

7. Conclusions and Future Work

This paper has proposed a novel distributed multi-stream clustering algorithm, Multi-Stream EvolveCluster (MS-EC), with a low communication cost. Each stream is clustered individually, and the only information sent to the centralized location is cluster labels, which preserves the privacy and integrity of the data. The foundations of the algorithm build on modularity and simplicity, where each stream can be configured precisely to its needs.

Our experiments have shown that two of the three MS-EC variants perform substantially better in clustering the synthetically made data. The third variant is also significantly better in the smaller dataset, but its performance drops when dimensionality and the number of streams increases. In the real-world data, though, we have identified that MS-EC relies heavily on the goodness and clusterability of the data in the streams. The privacy and integrity retaining setting of only submitting cluster labels is an exciting and challenging problem.

Our future plans are to dive further into the low communication and privacy setting by investigating more consensus clustering procedures. We also intend to investigate whether adding additional information to the individual streams and their clustering algorithms will positively affect the produced consensus clustering solution viability.

Acknowledgements

We appreciate and thank Marie Netz for her support and involvement. This research was funded partly by the Knowledge Foundation, Sweden, through the Human-Centered Intelligent Realities (HINTS) Profile Project (contract 20220068).

References

[1] C. Nordahl, V. Boeva, H. Grahn, and M. Persson Netz. “EvolveCluster: an evolutionary clustering algorithm for streaming data”. In: *Evol. Syst.* 13.4 (2022), pp. 603–623.

[2] S. Bickel and T. Scheffer. "Multi-view clustering." In: *IEEE Int. Conf. Data Min. 2004*. Vol. 4. 2004, pp. 19–26.

[3] Y. Yang and H. Wang. "Multi-view clustering: A survey". In: *Big Data Min. Anal.* 1.2 (2018), pp. 83–107.

[4] L. Fu, P. Lin, A. V. Vasilakos, and S. Wang. "An overview of recent multi-view clustering". In: *Neurocomputing* 402 (2020), pp. 148–161.

[5] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma. "A survey on ensemble learning". In: *Front. Comput. Sci.* 14 (2020), pp. 241–258.

[6] O. Sagi and L. Rokach. "Ensemble learning: A survey". In: *Wiley Interdiscip. Rev.: Data Min. Knowl. Discovery* 8.4 (2018), e1249.

[7] A. Strehl and J. Ghosh. "Cluster ensembles—a knowledge reuse framework for combining multiple partitions". In: *J. Mach. Learn. Res.* 3.Dec (2002), pp. 583–617.

[8] A. Fred. "Finding consistent clusters in data partitions". In: *Int. Workshop on Multiple Classifier Systems*. Springer. 2001, pp. 309–318.

[9] S. Van Dongen. "Graph Clustering Via a Discrete Uncoupling Process". In: *SIAM J. Matrix Anal. Appl.* 30.1 (2008), pp. 121–141.

[10] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. d. Carvalho, and J. Gama. "Data stream clustering: A survey". In: *ACM Comput. Surv. (CSUR)* 46.1 (2013), pp. 1–31.

[11] M. Ghesmoune, M. Lebbah, and H. Azzag. "State-of-the-art on clustering data streams". In: *Big Data Anal.* 1.1 (2016), pp. 1–27.

[12] M. Carnein and H. Trautmann. "Optimizing data stream representation: An extensive survey on stream clustering algorithms". In: *Bus. Inf. Syst. Eng.* 61.3 (2019), pp. 277–297.

[13] A. Zubaroğlu and V. Atalay. "Data stream clustering: a review". In: *Artif. Intell. Rev.* 54.2 (2021), pp. 1201–1236.

[14] P. Laurinec and M. Lucká. "Interpretable multiple data streams clustering with clipped streams representation for the improvement of electricity consumption forecasting". In: *Data Min. Knowl. Discovery* 33.2 (2019), pp. 413–445.

[15] K. Gajowniczek, M. Bator, and T. Ząbkowski. "Whole time series data streams clustering: dynamic profiling of the electricity consumption". In: *Entropy* 22.12 (2020), p. 1414.

[16] K. Gajowniczek, M. Bator, T. Ząbkowski, A. Orłowski, and C. K. Loo. "Simulation Study on the Electricity Data Streams Time Series Clustering". In: *Energies* 13.4 (2020), p. 924.

[17] A. Balzanella and R. Verde. "Histogram-based clustering of multiple data streams". In: *Knowl. Inf. Syst.* 62.1 (2020), pp. 203–238.

[18] Z. Razavi Hesabi, T. Sellis, and K. Liao. "DistClusTree: A Framework for Distributed Stream Clustering". In: *Databases Theory Appl.* Cham: Springer, 2018, pp. 288–299.

[19] J. S. Challa et al. "Anytime clustering of data streams while handling noise and concept drift". In: *J. Exp. Theor. Artif. Intell.* 34.3 (2022), pp. 399–429.

[20] M.-H. Le-Nguyen et al. "Continuous Health Monitoring of Machinery using Online Clustering on Unlabeled Data Streams". In: *2022 IEEE Int. Conf. on Big Data*. IEEE. 2022, pp. 1866–1873.

[21] L. Huang, C.-D. Wang, H.-Y. Chao, and S. Y. Philip. "MVStream: Multiview data stream clustering". In: *IEEE Trans. Neural. Netw. Learn. Syst.* 31.9 (2019), pp. 3482–3496.

[22] N. Chinchor. "MUC-4 Evaluation Metrics". In: *Proc. 4th Conf. on Message Understanding*. MUC4 '92. McLean, Virginia: Association for Computational Linguistics, 1992, pp. 2229.

[23] P. J. Rousseeuw. "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *J. Comput. Appl. Math.* 20 (1987), pp. 53–65.

[24] L. Hubert and P. Arabie. "Comparing partitions". In: *J. Classif.* 2 (1985), pp. 193–218.

[25] N. X. Vinh, J. Epps, and J. Bailey. "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" In: *Proc. 26th Annu. Int. Conf. Mach. Learn.* 2009, pp. 1073–1080.

[26] A. E. Baya and P. M. Granitto. "How many clusters: A validation index for arbitrary-shaped clusters". In: *IEEE/ACM Trans. Comput. Biol. Bioinf.* 10.2 (2013), pp. 401–414.

[27] S. Schlag. "High-Quality Hypergraph Partitioning". PhD thesis. Karlsruhe Institute of Technology, Germany, 2020.

[28] S. Makonin et al. "Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014". In: *Sci. Data* 3.1 (2016), pp. 1–12.