K-means for Evolving Data Streams

1st Arkaitz Bidaurrazaga
Basque Center for Applied Mathematics
Bilbao, Spain
abidaurrazaga@bcamath.org

2nd Aritz Pérez
Basque Center for Applied Mathematics
Bilbao, Spain
aperez@bcamath.org

3rd Marco Capó
Basque Center for Applied Mathematics
Bilbao, Spain
mcapo@bcamath.org

***Abstract*—Nowadays, streaming data analysis has become a relevant area of research in machine learning. Most of the data streams available are unlabeled, and thus it is necessary to develop specific clustering techniques that take into account the particularities of the streaming data. In streaming data scenarios, the data is composed of an increasing sequence of batches of samples where the concept drift phenomenon may occur. In this work, we formally define the streaming K-means (SKM) problem, which implies a restart of the error function when a concept drift occurs. An approximated error function that does not rely on concept drift detection is proposed. We prove that such a surrogate is a good approximation of the SKM error. Then, we introduce an algorithm to deal with SKM problem by minimizing the surrogate error function each time a new batch arrives. Alternative initialization criteria are presented and theoretically analyzed for streaming data scenarios. Among them, we develop and analyze theoretically two initialization methods that search for the best trade-off between the importance that is given to the past and the current batches. The experiments show that the proposed algorithm with, the proposed initialization criteria, obtain the best results when dealing with the SKM problem without requiring to detect when concept drift takes place.**

I. Introduction

One of the most relevant unsupervised data analysis problems is clustering [1], which consists of partitioning the data into a number of disjoint subsets called clusters. Among a wide variety of clustering methods, K-means algorithm is one of the most popular [2]. In fact, it has been identified as one of the top-10 most important algorithms in data mining. Before explaining the K-means algorithm, we shall briefly introduce the K-means problem.

A. K-means Problem

Given a data set of d-dimensional points of size n, $X = \{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$, the K-means problem is defined as finding a set of K centroids $C = \{\mathbf{c}_k\}_{k=1}^K \subset \mathbb{R}^d$, which minimizes the *$K$-means error function*:

$$E(X, C) = \frac{1}{|X|} \cdot \sum_{x \in X} \|\mathbf{x} - \mathbf{c}_x\|^2 \; ; \; \mathbf{c}_x = \underset{\mathbf{c} \in C}{\arg\min} \|\mathbf{x} - \mathbf{c}\|, \quad (1)$$

where $\|\cdot\|$ denotes the Euclidean distance or L^2 norm. In this work, as it is common in most of the K-means problem-related literature [3]–[6], the number of clusters is assumed to be predetermined and constant over time.

a) K-means Algorithm: The K-means problem is known to be NP-hard for $K > 1$ and $d > 1$ [7]. The most popular heuristic approach to this problem is Lloyd's algorithm [8]. Given a set of initial centroids, Lloyd's algorithm iterates two steps until convergence: 1) assignation step and 2) update step. In the assignation step, given a set of centroids, $C = \{\mathbf{c}_k\}_{k=1}^K$, the set of points is partitioned into K clusters, $\mathcal{P} = \{P_k\}_{k=1}^K$, by assigning each point to the closest centroid. Next, the new set of centroids is obtained by computing the center of mass of the points in each partition. This set of centroids minimizes the K-means error with respect to the given partition of the set of points. These two steps are repeated until a fixed point is reached, meaning, when the assignation step does not change the partition. This process has a $\mathcal{O}(n \cdot K \cdot d)$ time complexity. The combination of an *initialization method* plus Lloyd's algorithm is called the K-means algorithm.

b) K-means Initialization: The solution obtained by K-means algorithm strongly depends on the initial set of centroids [9]–[11]. Consequently, in the literature different initializations have been proposed. One of the most simple, yet effective, methods is Forgy's approach [12]. Forgy's initialization consists of choosing K data points at random as initial centroids. The main drawback of this approach is that it tends to choose data points located at dense regions of the space, thus these regions tend to be over-represented. In order to address this problem, probabilistic methods with strong theoretical guarantees have been proposed. K-means++ (KM++) [5] initialization iteratively selects points from X at random, where the probability of selection is proportional to the distance of the closest centroid previously selected. This strategy has become one of the most popular due its strong theoretical guarantees. Among other popular alternatives, we have variations of KM++, such as the K-means|| [6] and the Markov Chain KM++ [3].

B. Streaming Data

Although the K-means problem deals with a fixed data set X, its usage can be generalized to scenarios in which data evolves over time. We define streaming data (SD) as a set of data batches that arrive sequentially, where each batch is a set of d-dimensional points. One of the main concerns when processing SD is how much data to store, since the volume of data increases indefinitely. Normally, a maximum number of stored batches is determined, this way time consumption and

2374-8486/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDM51629.2021.00114

computational load of the clustering algorithm is controlled, and makes clustering tractable in this situation.

Another main issue when dealing with SD is the *concept drift* phenomenon. A concept drift occurs when the underlying probability distribution, associated to the batches, changes. In the presence of concept drifts we distinguish between passive and active approaches. On one hand, an active mechanism dynamically adjusts stored batches depending on whether a concept drift has occurred or not. On the other hand, in the passive approaches, more importance is given to recent batches. A detailed review on active and passive strategies to deal with concept drift can be found in [13].

C. Contributions

In this paper, we formally define the Streaming K-means (SKM) problem. Our proposal deals with the concept drift phenomenon by assigning exponentially decaying weights to older batches. We prove that the surrogate error is a good approximation to the SKM error, using Hoeffding's inequality [14]. Due to the importance of the initialization in the behaviour of weighted K-means, we present two initialization techniques that search for the best combination between past and novel information of clusters. We conduct experiments to compare them with other two straight-forward initialization strategies, which are to compute KM++ on the novel batch to obtain initial centroids or simply use previously computed centroids.

This paper[1] is organized in the following way. In section II the SKM problem is defined. Next, we propose a passive approach, and prove its suitability to deal with the SKM problem. In section III we propose some appropriate initialization methods for the SKM problem. Finally, we conduct the experiments[2] in section IV to compare and discuss the results obtained for each method under different scenarios.

II. Streaming K-means Problem

In this section, we define the SKM problem, a natural adaptation of the K-means problem for SD consisting of the minimization of the SKM error. We define the SKM error function as follows:

Definition 1. *Given a set of batches, $\mathcal{X} = \{B^t\}_{t\geq 0}$ and a set of centroids C, the SKM error function is defined as*

$$E_*(\mathcal{X}, C) = \frac{1}{M_T} \cdot \sum_{t=0}^{T-1} \sum_{x \in B^t} \|\boldsymbol{x} - \boldsymbol{c}_x\|^2, \tag{2}$$

where the index t describes the antiquity of each batch, thus B^0 is the latest batch received, and B^{T-1} represent the batch in which the last concept drift occurred, i.e., every batch $\{B^t\}_{t=0}^{T-1}$ shares the same underlying distribution. $M_T = \sum_{t\leq T-1} |B^t|$ is the sum of each batch size.

In order to compute the SKM error function, we need to know the batch in which the last concept drift occurred, B^{T-1}. The complexity of computing the SKM error is $\mathcal{O}(M_T \cdot K \cdot d)$. Clearly, the performance of an active approach to the problem will strongly depend on the behavior of the detection mechanism implemented. On one hand, when a fake drift is detected the past batches and the last centroids are discarded unnecessarily. On the other hand, if a concept drift occurs but is not detected, then the use of previously computed centroids could lead to a poor initialization and may infer a bad clustering. In this work, we describe an active algorithm, which we call Privileged SKM algorithm (PSKM). PSKM is an ideal baseline to the problem which knows in advance if a concept drift occurs, thus being able to compute and minimize the SKM error function. Alternatively, in the following section we propose a passive mechanism to approximate the solution of the SKM problem. PSKM will be the reference in the experimental section since we will simulate streaming data with concept drifts, and thus we will be able to compute the SKM error.

A. A Surrogate for SKM Error

Here we propose a *surrogate error function* for the SKM error function. This alternative function incorporates a forgetting mechanism based on a memory parameter, ρ, which assigns an exponentially decreasing weight ρ^t to a given batch based on its antiquity t. Note that $t = 0$ indicates the last batch obtained from the stream which has weight 1. In particular, the approximated error function is defined as follows:

Definition 2. *Given a set of batches of data points, $\mathcal{X} = \{B^t\}_{t\geq 0}$, the surrogate error function, for a given set of centroids C, is defined as*

$$E_\rho(\mathcal{X}, C) = \frac{1}{M_\mathcal{X}} \cdot \sum_{t\geq 0} \rho^t \cdot \sum_{x \in B^t} \|\boldsymbol{x} - \boldsymbol{c}_x\|^2 \tag{3}$$

where $M_\mathcal{X} = \sum_{t\geq 0} \rho^t \cdot |B^t|$ is the total weighted mass of the set of batches $\mathcal{X} = \{B^t\}_{t\geq 0}$.

The surrogate error is a weighted version of the K-means error for SD. Furthermore, the following theorem illustrates the suitability of this alternative function. For the sake of simplicity, we consider for this theorem that all batches have the same number of data points, $|B^t| = N$.

Theorem 1. *Let $\boldsymbol{c} \in \mathbb{R}^d$ be a point, $\mathcal{X} = \{B^t\}_{t\geq 0}$ be a set of batches of points in $\mathbb{R}^d$, where $B^t = \{\boldsymbol{x}_i^t\}_{i=1}^N$ and t denotes the antiquity of B^t. Let the batches before the drift $\{B^t\}_{t>T-1}$ be i.i.d. according to the probability p, where $\mathbb{E}_p[\|\boldsymbol{x} - \boldsymbol{c}\|^2] = E$. Let the batches after the drift $\{B^t\}_{t\leq T-1}$ be i.i.d according to p', where $\mathbb{E}_{p'}[\|\boldsymbol{x} - \boldsymbol{c}\|^2] = (1+\varepsilon) \cdot E$ for $\varepsilon > 0$. Let us assume that $\|\boldsymbol{x}_i^t - \boldsymbol{c}\|^2$ is upper-bounded by $u \geq 0$, for $i = 1, ..., N$ and $t \geq 0$.*

Then, with at least probability $1 - \delta$, the difference $E_*(\mathcal{X}, \{\boldsymbol{c}\}) - E_\rho(\mathcal{X}, \{\boldsymbol{c}\})$ satisfies:

$$E_*(\mathcal{X}, \{\boldsymbol{c}\}) - E_\rho(\mathcal{X}, \{\boldsymbol{c}\}) \in (\rho^T \cdot \varepsilon \cdot E - \gamma, \rho^T \cdot \varepsilon \cdot E + \gamma), \tag{4}$$

where $\gamma = u \cdot \sqrt{\frac{(2\cdot\rho^T - 1)/T + (1-\rho)/(1+\rho)}{2\cdot N} \cdot \ln \frac{2}{\delta}}$.

[1] This paper is a reduced version of [15].

[2] Code available at *https://github.com/arkano29/Kmeans_Streaming_Data*

Algorithm 1 Forgetful SKM algorithm (FSKM)

Predetermined: Maximum number of batches saved T_{max} and forget parameter ρ.
Input: Set of previous batches $\mathcal{X}$ partitioned by $\mathcal{P}$ and new batch B^0.
Output: A set of centroids C and its associated partition $\mathcal{P}$.
if $|\mathcal{X}|$==T_{max} **then**
    Remove the oldest batch from $\mathcal{X}$
$\mathcal{X} \leftarrow$ Append B^0
$C \leftarrow$ Initialization$(\mathcal{X}, \mathcal{P})$
$C, \mathcal{P} \leftarrow$ Weighted Lloyd$(\mathcal{X}, C)$
return $C, \mathcal{P}$

Here we define the $(1+\varepsilon)$-drift, which occurs when two underlying distributions p and p' satisfy $\mathbb{E}_{p'}[\|\mathbf{x}-\mathbf{c}\|^2] = (1+\varepsilon)\cdot\mathbb{E}_p[\|\mathbf{x}-\mathbf{c}\|^2]$, for $\varepsilon > 0$. More importantly, observe that, according to Theorem 1, the expected value of the alternative error function tends to the SKM error function exponentially fast with T for a single centroid, since the mean value of their difference has the form $\rho^T \cdot \varepsilon \cdot E$. In particular, it shows that the surrogate function can be used to approximate the error for a single centroid, thus applying this result to every subgroup of points and their respective centroids yields a good approximation of the SKM error. In summary, Theorem 1 shows that we can deal with the SKM problem by minimizing the alternative error without having to detect concept drifts.

III. Streaming Lloyd's Algorithm

In order to deal with the SKM problem in a passive way, we propose the Forgetful SKM (FSKM) algorithm (Algorithm 1). FSKM approximates the solution of the SKM problem by minimizing the approximated error function. When a new batch B^0 arrives, FSKM runs an initialization procedure to find a set of initial centroids. Next, a weighted Lloyd's algorithm is carried out over the available set of batches $\mathcal{X}$. Recall that the running time of the weighted Lloyd's algorithm is $\mathcal{O}(n \cdot K \cdot d)$, where n is the total number of points to be clustered. However, we can compute an arbitrarily accurate surrogate error function by discarding batches with a negligible weight, thus defining a maximum number of batches stored T_{max}. By discarding the batches with negligible weights, the computational complexity of the weighted Lloyd's step of FSKM is reduced to $\mathcal{O}(M_{T_{\max}} \cdot K \cdot d)$, where $M_{T_{max}}$ is the sum of batch sizes of the stored batches. This is a passive approach, because it does not need to detect concept drifts, and it inherits the good properties of the surrogate function as alternative to the real SKM error.

A. Initialization Step

As previously discussed, the initialization stage has a major effect on the convergence of Lloyd's algorithm, and therefore of the FSKM algorithm. For this reason, in this section, we propose efficient procedures for the initialization step of FSKM. However, first we describe two simple approaches, which will be used for comparison. Once a new batch is received, a straight-forward initialization strategy is to use the previously computed set of centroids. We call this approach Previous Centroids (PC), and the set of centroids obtained in previous iterations will be denoted as $C^* = \{\mathbf{c}_k^*\}_{k=1}^K$. PC uses a set of locally optimal centroids for the past set of batches, which can be a good and efficient choice once a new batch is presented. An alternative straight-forward initialization is to use centroids obtained by applying a standard initialization procedure over the newest batch B^0, such as KM++. We call this approach Current Centroids (CC). The set of centroids obtained from initializing over the current batch is denoted as $C^0 = \{\mathbf{c}_1^0\}_{k=1}^K$. Clearly, CC allows FS$K$M to adapt rapidly when a concept drift occurs. However, this initialization does not take into account either the batches from the past or the set C^*. This could imply a waste of very valuable information, especially when a concept drift has not occurred for a long period of time.

B. Weighted Initialization

Considering the trade-off between the PC and CC approaches, we propose two efficient initialization strategies that combine information from PC and CC, by minimizing an upper-bound to the surrogate error function. The next result defines an upper-bound for the surrogate error function that will allow us to determine a competitive initialization for the FSKM algorithm.

Theorem 2. *Given two sets of centroids $C^* = \{\mathbf{c}_k^*\}_{k=1}^K$ and $C^0 = \{\mathbf{c}_k^0\}_{k=1}^K$, for any set of centroids $C \in \mathbb{R}^d$, the surrogate function $E_\rho(\mathcal{X}, C)$ can be upper-bounded as follows:*

$$E_\rho(\mathcal{X}, C) \leq f^\rho(\mathcal{X}, C) + const,$$

where $f^\rho(\mathcal{X}, C)$ is

$$\frac{1}{M_\mathcal{X}} \cdot \sum_{k=1}^{K} \left(w_k^* \cdot \|\boldsymbol{c}_{k'} - \boldsymbol{c}_k^*\|^2 + w_k^0 \cdot \|\boldsymbol{c}_{k''} - \boldsymbol{c}_k^0\|^2 \right), \quad (5)$$

for $\boldsymbol{c}_{k'} = \arg\min_{\boldsymbol{c}\in C} \|\boldsymbol{c}_k^* - \boldsymbol{c}\|$, $\boldsymbol{c}_{k''} = \arg\min_{\boldsymbol{c}\in C} \|\boldsymbol{c}_k^0 - \boldsymbol{c}\|$, where $w_k^* = \sum_{t\geq 1} \rho^t \cdot |B^t \cap P_k^*|$ and $w_k^0 = |B^0 \cap P_k^0|$ are the weights related to each centroid and const is a value independent of the set of centroids C. $B^t \cap P_k$ are the set of points in batch B^t that belong to the partition P_k.

Theorem 2 shows that the surrogate error is upper-bounded by $f^\rho(\mathcal{X}, C)$ plus a constant. In fact, observe that f^ρ has the form of a weighted K-means error with $\{\mathbf{c}_k^*, \mathbf{c}_k^0\}_{k=1}^K$ as the data points, and weights $W = \{w_k^*, w_k^0\}_{k=1}^K$. Hence, we propose an initialization procedure based on the weighted K-means algorithm over the union of both sets of centroids. We refer to this initialization as Weighted Initialization (WI, Algorithm 2), where its computational complexity is $\mathcal{O}(K \cdot \max\{|B^0|, K\} \cdot d)$.

C. Hungarian Initialization

An interesting analytical result can be acquired considering another assumption along with Theorem 2. Assume that each

Algorithm 2 Weighted K-means initialization (WI))

Predetermined: Number of clusters K,forget parameter ρ.
Input: A set of batches $\mathcal{X} = \{B^t\}_{t\geq 0}$, a set of previous centroids C^* which are induced by the partition $\mathcal{P}^*$.
Output: A set of new optimized centroids C and its associated partition $\mathcal{P}$.
$C^0, \mathcal{P}^0 \leftarrow$ KM++(B^0)
$w_k^*, w_k^0 \leftarrow$ Compute weights from $\mathcal{P}^*$ and $\mathcal{P}^0$
$C, \mathcal{P} \leftarrow$ Weighted K-means$(X = \{C^*, C^0\}, W = \{w_k^*, w_k^0\}_{k=1}^K)$
return $C, \mathcal{P}$

centroid $\mathbf{c}_k$ has a single pair of centroids $\mathbf{c}_k^*, \mathbf{c}_{\sigma(k)}^0$ which are the closest to itself from both sets C^* and C^0, and are distinct for each centroid $\mathbf{c}_k$. Thus, we can index the centroid $\mathbf{c}_k$ like the centroids in C^*, but a different indexation $k' = \sigma(k)$ may be needed for the centroids in C^0, represented by the permutation $\sigma(k)$. Then, we can re-write the upper-bound given in (5) as follows:

$$f^\rho(\mathcal{X}, C = \{\mathbf{c}_k\}_{k=1}^K) = \frac{1}{M_\mathcal{X}} \cdot \sum_{k=1}^{K} \big(w_k^* \cdot \|\mathbf{c}_k - \mathbf{c}_k^*\|^2 + \\ + w_{\sigma(k)}^0 \cdot \|\mathbf{c}_k - \mathbf{c}_{\sigma(k)}^0\|^2\big), \quad (6)$$

where the weights w_k^* and $w_{\sigma(k)}^0$ are the weights of c_k^* and $c_{\sigma(k)}^0$, respectively. The following theoretical result shows that the upper-bound $f^\rho(\mathcal{X}, C)$ can be analytically minimized with respect to $\mathbf{c}_k$ with this assumption.

Theorem 3. *Let $f^\rho(\mathcal{X}, C)$ be the function defined in (6) for a set of centroids $C = \{\boldsymbol{c}_k\}_{k=1}^K$ of size K, where $\boldsymbol{c}_k^*$ and $\boldsymbol{c}_{\sigma(k)}^0$ are given, and they are the closest points to $\boldsymbol{c}_k$ of the sets $\{\boldsymbol{c}_k^*\}_{k=1}^K$ and $\{\boldsymbol{c}_{\sigma(k)}^0\}_{k=1}^K$. Then the set of centroids that minimizes this function is given by:*

$$\boldsymbol{c}_k = \frac{1}{w_k^* + w_{\sigma(k)}^0} \cdot (w_k^* \cdot \boldsymbol{c}_k^* + w_{\sigma(k)}^0 \cdot \boldsymbol{c}_{\sigma(k)}^0), \quad (7)$$

Theorem 3 shows that, just by making the one-to-one assumption given by $\sigma(k)$, the optimal centroids C can be simply expressed as a linear combination between the elements of C^* and C^0. Notice that with this assumption we achieve an analytical minimum of $f^\rho(\mathcal{X}, C)$.

a) Linear Sum Assignment Problem: If we want to compute the optimal centroids under the previous assumption, $\sigma(k)$ must be found. In order to do so, we use the result in Theorem 3 to rewrite (6):

$$f^\rho(\mathcal{X}, C) = \frac{1}{M_\mathcal{X}} \cdot \sum_{k=1}^{K} \frac{w_k^* \cdot w_{\sigma(k)}^0}{w_k^* + w_{\sigma(k)}^0} \cdot \|\mathbf{c}_k^* - \mathbf{c}_{\sigma(k)}^0\|^2 \quad (8)$$

Hence, we define the matrix:

$$f_{k,k'} = \frac{w_k^* \cdot w_{k'}^0}{w_k^* + w_{k'}^0} \cdot \|\mathbf{c}_k^* - \mathbf{c}_{k'}^0\|^2 \ ; \ k, k' \in \{1, \ldots, K\}, \quad (9)$$

Algorithm 3 Hungarian Initialization (HI)

Predetermined: Number of clusters K,forget parameter ρ.
Input: A set of batches $\mathcal{X} = \{B^t\}_{t\geq 0}$, a set of previous centroids C^* which are induced by the partition $\mathcal{P}^*$.
Output: A set of new optimized centroids C.
$C^0, \mathcal{P}^0 \leftarrow$ KM++(B^0)
$w_k^*, w_k^0 \leftarrow$ Compute weights from $\mathcal{P}^*$ and $\mathcal{P}^0$
for k in 1,...,K **do**
  for k' in 1,...,K **do**
    $f_{k,k'} \leftarrow \frac{w_k^* \cdot w_{k'}^0}{w_k^* + w_{k'}^0} \cdot \|\mathbf{c}_k^* - \mathbf{c}_{k'}^0\|^2$
$\sigma \leftarrow \arg\min_{\sigma \in \Sigma} \sum_{k=1}^K f_{k,\sigma(k)}$
$C \leftarrow \emptyset$
for k in 1,...,K **do**
  $\mathbf{c}_k \leftarrow \frac{1}{w_k^* + w_{\sigma(k)}^0} \cdot (w_k^* \cdot \mathbf{c}_k^* + w_{\sigma(k)}^0 \cdot \mathbf{c}_{\sigma(k)}^0)$
  $C \leftarrow C \cup \mathbf{c}_k$
return C

and find the permutation $\sigma(k)$ such that the sum $\sum_{k=1}^K f_{k,\sigma(k)}$ is minimal. This is a linear sum assignment problem and we can make use of the Hungarian (or Kuhn-Munkres) algorithm [16] to determine $\sigma(k)$ with a computational complexity of $\mathcal{O}(K^3)$. Thus, we propose another initialization method called Hungarian Initialization (HI) based on this procedure. HI firstly computes a set of optimized centroids $\mathbf{c}_{k'}^0$ over the new batch B^0. Then the matrix $f_{k,k'}$ is constructed, which is used to determine the permutation that maps $k \to k' = \sigma(k)$, via the linear sum assignment problem (Algorithm 3). This way, the sum $\sum_{k=1}^K f_{k,\sigma(k)}$ is guaranteed to be the minimum value of $f^\rho(\mathcal{X}, C)$, and hence the new set of centroids can be computed as defined in Theorem 3. The computational complexity of this algorithm is $\mathcal{O}(K \cdot \max\{\max\{K^2, K \cdot d\}, |B^0| \cdot d\})$.

IV. Experiments

In this section we analyse the performance of the FSKM algorithm with the proposed initialization procedures: Previous Centroids (PC), Current Centroids (CC), Hungarian Initialization (HI) and Weighted Initialization (WI). The converged SKM error obtained by FSKM with different initialization strategies is compared with the gold-standard PSKM. In order to control the strength of the drifts, the experiments are performed using simulated streaming data with $(1+\varepsilon)$-drifts generated using real datasets taken from the *UCI Machine Learning Repository* [17], for different values of ε. For further insight on how we simulated streaming data see the extended version of this paper [15].

A. Experimental Setup

a) Datasets: The experiments have been carried out in 8 different datasets simulated based on real datasets from the *UCI Machine Learning Repository* [17], details about each dataset are available on the extended version of this paper. The simulated data consists of a sequence of batches with size $N = 500$, and a $(1+\varepsilon)$-concept drift takes place every 10 batches.

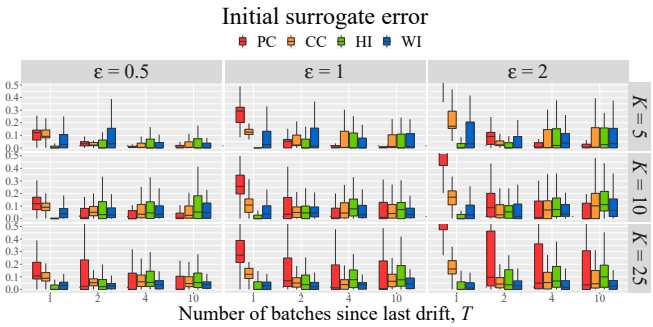

Fig. 1: Initial surrogate error for the FSKM algorithm with different initialization methods. ρ was determined with $m = 2$, and errors where normalized as specified in the former section.

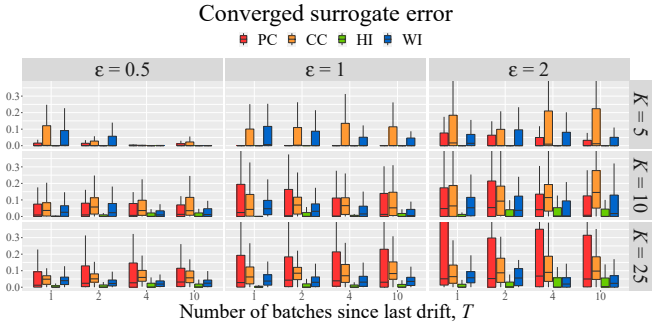

Fig. 2: Converged surrogate error for the FSKM algorithm with different initialization methods. ρ was determined with $m = 2$, and errors where normalized.

b) Procedure: To analyze the behavior of the algorithms in streaming scenarios, we perform a burning-out step by storing T_{max} batches from the first concept. After this step, we measure the evolution of the performance of PSKM, and FSKM with different initialization techniques. To fairly compare their behaviour, the set of centroids C^* and C^0 are the same for each initialization procedure each time a new batch arrives. After the burning-out step, a stream of 100 batches is processed with concept drifts every 10 batches. This procedure is repeated for each dataset and value of the hyperparameters. Because the results did not vary too much for intermediate batches, we show measurements for the first and second batch (indexed by 1 and 2), an intermediate batch and the last batch before the next concept drift (indexed by 4 and 10).

c) Measurements: We have measured the quality of the solutions obtained by different procedures in terms of the SKM and approximated error function. In order to have comparable scores for different datasets, the obtained scores (error values) on initialization and convergence errors are normalized. For each new batch, the score E_M obtained with algorithm $M \in \mathcal{M}$ as $\hat{E}_M = (E_M - \min_{M' \in \mathcal{M}} E_{M'}) / \min_{M' \in \mathcal{M}} E_{M'}$. The computational load of the methods considered in our experimental setting is dominated by the number of distance computations. Therefore, as it is common practice in K-means problem related articles [3], [18], we use the number of distances computed to measure their computational performance. The computed distances D_M were also normalized, but we simply divide by the minimum $\hat{D}_M = D_M / \min_{M' \in \mathcal{M}} D_{M'}$.

d) Hyperparameters: A key parameter is the forget parameter ρ, since the approximated function directly depends on this parameter. Theorem 1 shows that the surrogate differs from the real SKM error with $\rho^T \varepsilon$, but the confidence interval grows as ρ decreases, hence a proper balance is necessary. Assuming that a difference of 0.01 is negligible, we can determine the ρ value by solving the equality $\varepsilon \cdot \rho^{\tau/m} = 0.01$, where τ is our prior knowledge about the (average) number of batches in which a concept is stable and m represents the fraction of the period in which we want the difference to become negligible. Intuitively, m determines how fast the term $\rho^T \cdot \varepsilon$ shrinks relative to the period of when a drift happens, τ. The magnitude of the concept drift and the number of clusters can affect how fast each algorithm adapts. For this reason, when generating streaming data, we use the next set of values for the parameters ε and K: $\varepsilon \in \{0.5, 1, 2\}$, $K \in \{5, 10, 25\}$. Note that for each value of ε and m, we set a different value of ρ. In this paper, we show results for $m = 2$, for the sake of brevity. For $m = 2$, the values of ρ depended on ε, which were $\rho \in \{0.457, 0.398, 0.347\}$ in the order of ε given before.

B. Initial and Converged Errors

HI and WI show better initial surrogate errors compared to PC and CC when a concept drift occurs (see Figure 1, $T = 1$), for every ε and K. When a concept drift occurs, PC performs poorly, since its initial centroids are focused on minimizing the approximated error function for the previous concept. For smaller values of ρ, CC gets better results than PC when a drift occurs, because previous batches contribute less to the surrogate error. In this sense, CC gets better results than PC as ε increases, because previously computed centroids become an even worse approximation for the novel concept. As new batches arrive, we observe that PC obtains the best initial surrogate error, because stored batches share the same underlying distribution and previously converged centroids are good for initialization.

Figure 2 summarizes the surrogate error function of FSKM at convergence. HI and WI stand out over the trivial initialization methods. Moreover, HI obtains median normalized scores close to 0 for every value of K and ε. In the previous figure, it was shown that WI obtained a better initialization error, but now HI obtains a lower converged error. HI initialization is more restricted than WI, obtaining a worse initialization error. However, this restriction seems to be reasonable since the fixed points where HI arrives get a better converged error. Furthermore, WI executes K-means over centroids, and completely ignores the structure of data points, which may lead to re-assignations that increase the error. PC shows a higher variance, especially for bigger values of K.

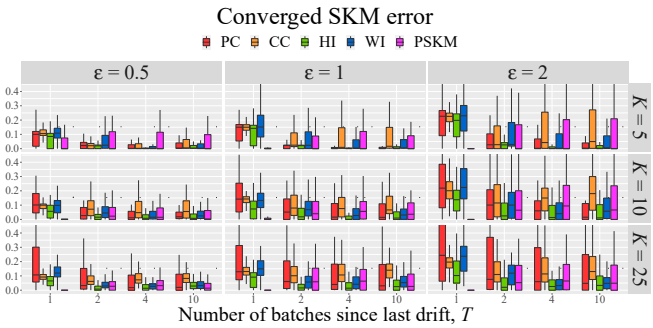

Fig. 3: Converged real SKM error for the FSKM (multiple initialization) and PSKM algorithms. Here $m = 2$ and errors were normalized.

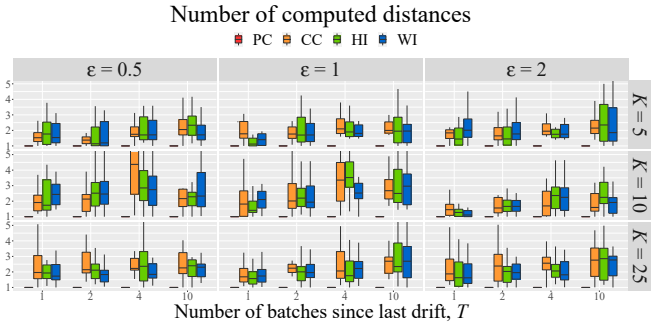

Fig. 4: Number of computed distances, normalized as $\hat{D}_M = D_M / \min_{M' \in \mathcal{M}} (D_{M'})$. PC's boxplot is flat since its initialization needs no computation, thus saving a lot of computed distances.

In Figure 3, we show the SKM error at convergence. Here the results of PSKM are shown as a reference. Observe that, in general, the medians of the converged SKM error are comparable for every algorithm, especially when many batches of the same concept have already happened ($T = 10$). Recall that FSKM does not minimize the SKM error, concluding that the surrogate is a good approximation and that every initialization technique (except for CC) works properly. We see that even though PSKM obtains the best scores when a drift occurs, after the next batch (index 2) HI and WI already attain scores comparable to PSKM in terms of medians. In terms of dispersion, HI and WI are even more stable (smaller variance) than PSKM. We know from Theorem 1 that the surrogate error approximates the SKM error better when more batches occurred since the last concept drift, this can explain why, even though FSKM does not explicitly minimize the SKM error, its convergence value is better than the one obtained by PSKM. We can see that in the last batch, before a concept drift occurs, FSKM obtains scores comparable to PSKM as well.

C. Computed Distances

Not needing any extra computation for the initialization makes PC compute less distances, thus we use PC as a reference in Figure 4, where the number of distances is shown relative to PC's number of computed distances. Because distances are normalized, what we observe in the Y axis is how many times more distances have been computed compared to PC. Considering every boxplot, we conclude that the medians of HI, WI and CC are around 2, thus they compute twice as many distances as PC in general. In the previous section we have observed that HI outperformed in terms of converged SKM error. Thus, this extra distance computation is a trade-off in order to adapt to concept drifts more effectively.

Acknowledgments

This research is supported by the Basque Government through the BERC 2018-2021 program and by Spanish Ministry of Sciences, Innovation and Universities: BCAM Severo Ochoa accreditation SEV-2017-0718.

References

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.
[2] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.
[3] O. Bachem, M. Lucic, S. H. Hassani, and A. Krause, "Approximate K-means in sublinear time," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
[4] M. Capo, A. Perez, and J. A. A. Lozano, "An efficient Split-Merge re-start for the K-means algorithm," *IEEE Transactions on Knowledge and Data Engineering*, 2020.
[5] A. David, "K-means : The Advantages of Careful Seeding," in *18th annual ACM-SIAM symposium on Discrete algorithms (SODA), New Orleans, Louisiana*, pp. 1027–1035, 2007.
[6] K. Makarychev, A. Reddy, and L. Shan, "Improved guarantees for K-means++ and K-means++ parallel," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[7] A. Vattani, "K-means requires exponentially many iterations even in the plane," *Discrete and Computational Geometry*, vol. 45, no. 4, pp. 596–616, 2011.
[8] S. Lloyd, "Least squares quantization in PCM," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.
[9] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the K-means clustering algorithm," *Expert Systems with Applications*, vol. 40, no. 1, pp. 200–210, 2013.
[10] P. Fränti and S. Sieranoja, "How much can K-means be improved by using better initialization and repeats?," *Pattern Recognition*, vol. 93, pp. 95–112, 2019.
[11] D. Steinley and M. J. Brusco, "Initializing K-means batch clustering: A critical evaluation of several techniques," *Journal of Classification*, vol. 24, no. 1, pp. 99–121, 2007.
[12] E. W. Forgy, "Cluster analysis of multivariate data: efficiency versus interpretability of classifications," *Biometrics*, vol. 21, pp. 768–769, 1965.
[13] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014.
[14] W. Hoeffding, *Probability inequalities for sums of bounded random variables*, pp. 409–426. The Collected Works of Wassily Hoeffding, Springer, 1994.
[15] A. Bidaurrazaga, A. Pérez, and M. Capó, "K-means for evolving data streams," 2021. arXiv: 2012.03628.
[16] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
[17] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.
[18] M. Capo, A. Perez, and J. A. Lozano, "An efficient approximation to the K-means clustering for massive data," *Knowledge-Based Systems*, vol. 117, pp. 56–69, FEB 2017.