Ensemble Online Clustering through Decentralized Observations

Dimitrios Katselis, Carolyn L. Beck and Mihaela van der Schaar

***Abstract*— We investigate the problem of online learning for an ensemble of agents clustering incoming data, i.e., the problem of combining online local clustering decisions made by distributed agents to improve knowledge and accuracy of implicit clusters hidden in the incoming data streams. We focus on clustering using the well-known K-means algorithm for numerical data due to its efficiency in clustering large data sets. Nevertheless, our results can be straightforwardly extended to, e.g., the K-modes variant of the K-means algorithm to handle categorical data, as well as to other clustering algorithms. We show that the proposed ensemble online solutions, which are based on a simple majority-voting scheme, converge to the centralized solutions that would be made by a fusion center, that is, the solutions resulting from one agent with access to all information across agents. Given the dimensions of the clustering model, the aforementioned convergence is demonstrated to be achievable for relatively small sizes of the ensemble.**

I. Introduction

In the generic framework of supervised learning, we are given data in the form of pairs, $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$, which are related by an implicit mapping $y_n = f(\mathbf{x}_n)$. The goal is to identify or to approximate the mapping $f(\cdot)$ [4]. Nevertheless, in many applications there is a lack of feedback from the supervisor in terms of data labels, which motivates the need for unsupervised procedures while making efficient learning much more challenging. In the unsupervised learning context, the data misses the labels, i.e., the available data set is $\{\mathbf{x}_n\}_{n=1}^N$. The main goal here is to identify the *structure* in the data or more explicitly to use the available data set sampled from a particular distribution or mixed distributions in order to describe properties of the underlying probabilistic law. Among others, one possible procedure of descriptive statistics that could be cast in the unsupervised learning framework is *clustering* [11]. Clustering finds numerous applications, e.g., in biology for inferring population structures based on gene similarities, in marketing for market segmentation and product positioning, in social networks for recognizing communities in large groups of people, and in recommender systems for predicting users' preferences based on the preferences of other users in the same cluster [1], [4], [5], [11].

A generic definition of clustering is the process of grouping data together based on some measure of similarity. The usual outcome of such a process is the division of the feature space into regions of high and low data densities. Selecting representatives in each region, we can establish prototypes, e.g., for encoding/decoding the data. Forming the clusters we can detect outliers, while we can also simplify the data for further analysis and learning. Generally, there are no 'bad' or 'good' clusterings. What is actually the case is that different clusterings can unveil different aspects of the data. Clustering can be *hierarchical* or *partitional* [11]. In this paper, we focus on the latter case, also called *flat* clustering.

In the data mining framework, one generally deals with very large and complex data sets [1]. This creates the incentive for employing algorithms that can efficiently cluster large data sets, where the data points can take both numerical and categorical values. Such efficient algorithms are the $K-$means and the $K-$modes algorithms [5]. Furthermore, in many current applications there is a motivation for decentralizing the clustering procedure, e.g., in sensor networks where the nodes have different collections of data associated with possibly the same event across time which need to be collectively labeled, in robotics where multiple robots observe similar objects and try to identify them for example as obstacles or targets, and possibly in military applications where the agents observe various adversarial objects which require classification of the corresponding types [3], [7], [9]. Furthermore, real time operation of a group of coordinated agents collectively performing a task, e.g., that of clustering, requires online implementation of the underlying algorithms. This allows for introducing dynamics into the aforementioned task, while maintaining control over it.

In this paper, we propose two online ensemble learning implementations of the $K-$means algorithm. Specifically, we consider a set of distributed learners that at every time instant observe cluster-correlated data from a single or multiple sources. This setup is motivated by the aforementioned sensor network and robotics applications. Upon the arrival of a new observation, each learner assigns it to a cluster and broadcasts this information to all other learners. The final clustering decision is based on a *majority vote*. Some theoretical guarantees for the resulting clustering rules are then derived. The proposed approach shares in principle some common features with the approaches in [9] mainly developed in the supervised learning framework. Furthermore, only scalar decisions are exchanged *once* among the nodes at every time instant due to the majority rule in our framework.

The rest of the paper is organized as follows: Section II sets up the considered ensemble framework. Section III proposes different online implementations of the $K-$means algorithm. Approaches for selecting the initial means are provided in Section IV. Some theoretical analysis is given in Section V, while simulations are presented in Section VI. Finally, Section VII concludes the paper.

Dimitrios Katselis and Carolyn L. Beck are with the Coordinated Science Laboratory and the Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801-2925, Emails: {katselis|beck3}@illinois.edu.

Mihaela van der Schaar is with the Electrical Engineering Department, UCLA, Los Angeles CA 90095, USA. Email: mihaela@ee.ucla.edu.

Notation: Sets are denoted by uppercase calligraphic letters. Bold lower case letters are used to denote vectors. $\mathbb{R}$ stands for the real field and $\mathbb{Z}$ for the set of integer numbers. For $n \in \mathbb{Z}$, $[n]$ corresponds to the set $\{1, 2, \ldots, n\}$. For a set $\mathcal{A}$ with at least t elements obtained in a specific temporal order, $\mathcal{A}_{[t]}$ is the subset of $\mathcal{A}$ with elements up to time t and $\mathcal{A}_t$ is the singleton containing only the element obtained at time t. For a set $\mathcal{M} \subset \mathbb{R}^d$, $\mathrm{dist}(\mathbf{m}, \mathcal{M})$ denotes the usual distance of $\mathbf{m} \in \mathbb{R}^d$ from $\mathcal{M}$. Finally, $\mathbf{I}_d$ is the $d \times d$ identity matrix and $\mathcal{N}(\mathbf{m}, \mathbf{\Sigma})$ denotes the Gaussian distribution with mean $\mathbf{m}$ and covariance $\mathbf{\Sigma}$.

II. Ensemble-Based Learning Framework

Consider a set of M distributed learners or agents $\mathcal{L} = [M]$. Each learner observes a sequence of samples from one or many sources. The underlying assumption is that the sources exhibit correlation on a common event, namely the cluster to which their current sample is allocated. Except for this correlation, the data points from different sources are considered conditionally independent given the cluster to which they belong. Furthermore, we make the usual assumption that the data are stationary[1] and the first and second order moments of their underlying statistical laws are finite.

We consider the synchronous scenario where all learners receive a new observation at the beginning of each time slot. Let $\mathbf{x}_n^{(i)} \in \mathbb{R}^d$ be the observation of the ith learner at the beginning of the nth slot. Assuming the existence of K clusters[2] for all learners, the task of each learner is to assign this observation to the appropriate cluster so that a properly defined cost is minimized. The clustering process utilizes the idea of ensemble data mining: each learner generates a local clustering decision, the clustering decisions are then exchanged and each learner combines the received decisions with its own to produce the final clustering decision. We assume that the local clustering rules are given and we focus on their adaptivity when the aggregate clustering information becomes available. Upon the reception of the aggregate clustering information, each learner uses a simple *majority-voting* rule to finally decide the appropriate cluster for the datum. The update of the cluster means is based on this decision and it is performed on the basis of each learner's current observation. Clearly, in the case that $M > K$, such a majority notion is well-defined due to the pigeonhole principle[3]. On the other hand, if $M \leq K$, such a majority may fail to exist. In this case, each learner will use its original local clustering decision as the final one.

The decentralized clustering schemes in this paper will be compared with the best possible clustering that can be achieved by an agent who has access to the data across all learners and for all time instants. In practice, such an agent will wait for the end of the observation time interval and will perform a one time clustering over all the available data. We refer to such an agent as the *fusion center*. Centralized online algorithms will be also investigated. Although a fusion center can admit online operations, we will refer to the central node implementing the centralized *online* algorithms as the *central learner* to avoid any confusions between online and batch schemes.

[1] spatially

[2] Information a priori known in applications such as the design of vector quantizers for data compression or decided based on metrics such as the Akaike Information Criterion (AIC) etc [2], [11].

[3] Ties can occur but under some regularity conditions on the separation of clusters or the variances of the data points within each cluster and possibly on the size of M, such ties are assumed improbable.

III. Centralized and Ensemble $K-$means Schemes

In the classical setup of the $K-$means algorithm, the ith learner has a set of N observations $\mathcal{O}_{[N]}^{(i)} = \{\mathbf{x}_n^{(i)}\}_{n=0}^{N-1}$, where $\mathbf{x}_n^{(i)} \in \mathbb{R}^d, \forall i \in [M], n \in [N]$, and wants to organize them into K subsets $\{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$. The partitioning aims at solving the following nonconvex combinatorial optimization problem:

$$\begin{aligned} \min_{\{b_{kl}^{(i)} \in \{0,1\}\}_{k,l=1}^{K,N}, \{\mathbf{m}_k^{(i)}\}_{k=1}^{K}} & \sum_{k=1}^{K} \sum_{l=1}^{N} b_{kl}^{(i)} d\left(\mathbf{x}_l^{(i)}, \mathbf{m}_k^{(i)}\right) \\ \text{s.t.} \quad & \sum_{k=1}^{K} b_{kl}^{(i)} = 1, \quad \forall l \in [N], \\ & b_{kl}^{(i)} \in \{0,1\}, \quad \forall k \in [K], l \in [N], \end{aligned} \tag{1}$$

where $b_{kl}^{(i)} = 1$ if $\mathbf{x}_l^{(i)} \in \mathcal{C}_k$ and $b_{kl}^{(i)} = 0$ otherwise. Additionally, $d\left(\mathbf{x}_l^{(i)}, \mathbf{m}_k^{(i)}\right)$ can be any distance function. Usually, the squared Euclidean distance is used, i.e., $d\left(\mathbf{x}_l^{(i)}, \mathbf{m}_k^{(i)}\right) = \|\mathbf{x}_l^{(i)} - \mathbf{m}_k^{(i)}\|^2$. After initializing the means of the clusters, the $K-$means algorithm solves the last problem iteratively by alternating between the two different sets of variables $\left\{b_{kl}^{(i)}\right\}_{k,l=1}^{K,N}$ and $\left\{\mathbf{m}_k^{(i)}\right\}_{k=1}^{K}$ in (1). The algorithm converges since it generates a sequence of decreasing objective values. Moreover, it works well for compact and hyperspherical clusters [5]. It additionally tessellates the observation space with convex clusters. Nevertheless, due to the nonconvexity of (1), the convergence point can be a local minimum highly dependent on the initialization of the cluster means.

In the case of multiple learners, the problem is as follows: Assume that the samples of all learners were available centrally at the fusion center. Then, one would want to cluster all data jointly. Ideally, the goal is to have each learner assigning its data to the cluster that the fusion center would have assigned it only by using the assumption that observations in the same time slot correspond to the same cluster. Moreover, as already mentioned in the previous section, the central clustering performed by the fusion center is assumed to be implemented in a *batch* fashion, i.e., after all measurements have been collected. Assuming that the learners make some initial clustering decisions on the incoming data, the aforementioned ideal goal has to be satisfied when only these initial clustering decisions are exchanged among the learners to keep the communication cost at a minimal level.

Let us first examine the possibility of different online clustering approaches performed by a *central learner*. Collecting all sets of observations, the central learner obtains $\mathcal{O}_{[N]} = \cup_{i=1}^{M} \mathcal{O}_{[N]}^{(i)}$. Using the fact that contemporary samples belong to the same cluster, the corresponding optimization problem becomes:

$$\begin{aligned} \min_{\{\beta_{kl}\in\{0,1\}\}_{k,l=1}^{K,N}, \{\mathbf{m}_k\}_{k=1}^{K}} & \sum_{k=1}^{K}\sum_{l=1}^{N} \beta_{kl} d'\left(\cup_{i=1}^{M}\mathcal{O}_l^{(i)}, \mathbf{m}_k\right) \\ \text{s.t.} \quad & \sum_{k=1}^{K} \beta_{kl} = 1, \quad \forall l \in [N] \\ & \beta_{kl} \in \{0,1\}, \quad \forall k \in [K], l \in [N]. \end{aligned} \tag{2}$$

Here, $d'\left(\cup_{i=1}^{M}\mathcal{O}_l^{(i)}, \mathbf{m}_k\right) = \sum_{i=1}^{M} d\left(\mathbf{x}_l^{(i)}, \mathbf{m}_k\right)$, which serves as the candidate distance function.

Clearly, (2) can be solved using the $K-$means algorithm. Furthermore, it is not hard to see that (2) admits an online implementation of the $K-$means algorithm: At time instant n, the central learner re-initializes and executes the $K-$means algorithm with $\mathcal{O}_{[n]} = \cup_{i=1}^{M}\mathcal{O}_{[n]}^{(i)}$. For every re-initialization, the mean values are always set to the same initial values as those in the batch implementation performed by the fusion center. This point is necessary, since the clustering solution produced by the fusion center is the reference basis for all the comparisons in this paper. More explicitly, at the Nth time instant we would like the just described central online solution to coincide with the fusion center solution. We call this online implementation **Central Online K-means with Re-Initialization** (CRI). Furthermore, the clustering decisions up to time $n-1$ are updated by the $K-$means execution at time n. This update may or may not be ignored by the system designer.

Algorithm 1 CRI

Require: Initial means $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$.

1: **loop**
2: while $n \leq N$
3: Re-initialize the means to $\mathcal{M}^0$.
4: Solve (2) using K-means with $N = n$.
5: Return the clustering decision(s) for $\mathcal{O}_n$ or $\mathcal{O}_{[n]}$.

Under computational considerations, a different online implementation of the central K-means is possible: initially, all cluster means are set to the same values as those used by the fusion center. At time instant n, the new data is assigned to a cluster by the central learner and only the corresponding mean is updated. Already clustered data up to time $n-1$ are considered decided and they are not re-clustered. Therefore, only the mean of the selected cluster is updated. We call this online implementation **Central Online K-means with Hard Temporal Decisions** (CHTD). Clearly, depending on the accuracy of the intial mean values the CHTD does not generally deliver the same solution as the central $K-$means implemented by the fusion center. This is the price paid for the computational efficiency of the CHTD and its possible flexibility to time-varying statistics of the incoming data streams.

Algorithm 2 CHTD

Require: Initial means $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$.

1: **loop**
2: while $n \leq N$
3: Assign $\mathcal{O}_n$ to the closest cluster.
4: Declare the clustering decision of $\mathcal{O}_n$ as **final**.
5: Update the corresponding cluster mean.

We now focus on the ensemble setup. Suppose that $n = 0$ corresponds to an instant that all learners initialize their cluster means to the same a priori agreed values. Therefore, all learners are assumed to have the initial cluster means $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$, where $\mathbf{m}_j^0 \in \mathbb{R}^d, \forall j \in [K]$. This common mean initialization is essential for setting up the reference basis of the cluster decisions to be exchanged among the learners later on. Furthermore, $\mathcal{M}^0$ is assumed to coincide with the initial mean values employed by the fusion center. At time instant n, each learner receives a new datum. A local $K-$means algorithm is initialized at $\mathcal{M}^0$ and it operates on $\mathcal{O}_{[n]}^{(i)}, \forall i \in [M]$. The clustering decisions of $\mathbf{x}_n^{(i)}$ are exchanged. Based on the majority rule, the final cluster decision about $\mathbf{x}_n^{(i)}$ is made. We call this algorithm **Ensemble Online Majority K-means with Re-Initialization** (EMRI). One may observe that in order to make the majority decisions meaningful, the already clustered data up to time $n-1$ are not re-clustered, i.e., the online clustering decisions are *temporally hard*. This is a major difference from the centralized analog CRI, if the system designer chooses to implement the CRI with re-clustering of the past data. To elaborate on this point, note that the local $K-$means procedures executed at time n may cluster the data up to time $n-1$ to different clusters than those already decided based on the majority rule. Nevertheless, although the clustering decisions of the data up to time $n-1$ by the local $K-$means algorithms are correlated with the clustering decision of $\mathbf{x}_n^{(i)}, \forall i \in [M]$, this point is ignored by the EMRI.

Algorithm 3 EMRI

Require: Common initial means $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$.

1: **loop**
2: while $n \leq N$
3: Each learner performs a local K-means based on $\mathcal{O}_{[n]}^{(i)}$ with re-initialization to $\mathcal{M}^0$.
4: The learners exchange their votes.
5: Final clustering decisions for $\mathcal{O}_n^{(i)}$ are made.

Finally, there is an ensemble analog of the CHTD algorithm. The operation is exactly the same as that of EMRI with the difference that at time instant n the new data are clustered by the learners based on their distance from the current cluster means. The local decisions are exchanged and

the majority rule is applied. Only the cluster mean of the selected cluster is updated. Assuming that the final cluster mean at learner i is $\mathbf{m}_k^{n(i)}$ and the number of cluster points in this cluster is[4] N_k^n, the mean update equation is

$$\mathbf{m}_k^{n+1(i)} = \mathbf{m}_k^{n(i)} + \frac{1}{N_k^n + 1}\left(\mathbf{x}_n^{(i)} - \mathbf{m}_k^{n(i)}\right) \quad (3)$$

with $N_k^{n+1} = N_k^n + 1$. By definition this update is meaningful when $N_k^n \geq 1$ for good initial mean choices. We call this algorithm **Ensemble Online Majority K-means with Hard Temporal Decisions** (EMHTD).

Algorithm 4 EMHTD

Require: Common initial means $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$.

1: **loop**
2: while $n \leq N$
3: Each learner assigns $\mathcal{O}_n^{(i)}$ to the closest cluster mean.
4: The learners exchange their votes.
5: Final clustering decisions for $\mathcal{O}_n^{(i)}$ are made. The selected cluster means are updated based on (3).

Remark: The CHTD algorithm assigns the incoming data at time n to the cluster whose mean is the nearest one. This assignment is based on minimizing $d'\left(\cup_{i=1}^M \mathcal{O}_n^{(i)}, \mathbf{m}_k^n\right) = \sum_{i=1}^M d\left(\mathbf{x}_n^{(i)}, \mathbf{m}_k^n\right)$ at time instant n. Here $\mathbf{m}_k^n$ is one of the K current centroids within CHTD's framework.

All presented algorithms are sensitive to the initial means selection. In the next section, we briefly discuss ways that such a choice can be made.

IV. Initial Means Selection

In this section, we investigate a crucial point, namely, the choice of the initial cluster means. Theoretical guarantees about the proposed algorithms are sensitive to this choice. We assume that we have an initialization phase where all algorithms use observations to select their cluster means. Suppose that it is known a priori that the cluster means are well separated by more than some value $\Delta > 0$. The first M points selected by the learners[5] are exchanged and the first cluster mean is formed as the sample mean of the current observations. The next M observations are again exchanged. If the corresponding sample mean has a distance less than $\Delta/2$ of the already selected mean, this mean is rejected[6]. The process is continued until the selection of K means. Each time, every new sample mean is tested against all previously selected means. Such a procedure terminates in finite steps with probability 1 (w.p.1), if all clusters have a nonzero probability of occurrence. Moreover, $N_k^0 = M, \forall k \in [K]$.

Lemma 1: The initial mean selection process corresponds to an irreducible and recurrent Markov chain. Moreover, it terminates in finite steps w.p. 1.

[4]Note that the means may differ per learner, but the number of points per cluster coincide for all learners based on the majority rule.

[5]The procedure is applicable by the centralized online algorithms as well.

[6]A different option would be to update the closest sample mean.

Algorithm 5 Initial Means Selection

Require: Initial cluster mean $\mathbf{m}_1^0$. Set $\mathcal{M}^0 = \{\mathbf{m}_1^0\}$.

1: **loop**
2: while the number of means is $\leq K$
3: The learners exchange their observations.
4: A new mean $\tilde{\mathbf{m}}$ is formed locally based on these observations.
5: **if** dist $\left(\tilde{\mathbf{m}}, \mathcal{M}^0\right) \geq \Delta/2$ **then**
6: $\mathcal{M}^0 = \mathcal{M}^0 \cup \{\tilde{\mathbf{m}}\}$.
7: Return $\mathcal{M}^0 = \{\mathbf{m}_1^0, \mathbf{m}_2^0, \ldots, \mathbf{m}_K^0\}$.

Sketch of Proof: Assume that the initialization process has just begun and that the first M observations have been received, i.e., $\mathcal{O}_1 = \cup_{i=1}^M \mathcal{O}_1^{(i)}$ is available. The sample mean of the received data points is formed and the centroid of the first cluster is determined. Given our assumptions, it is easy to see that the remaining mean selection process can be modeled as a finite state Markov chain with K states, each state being associated with a different cluster. The transition probabilities from any state to any other state are $p_i, i = 1, 2, \ldots, K$, where p_i is the probability of occurrence of the ith cluster. Note that these probabilities are the same for any current state. Clearly, the Markov chain is irreducible since all the states form a single communicating class. Additionally, the Markov chain is recurrent, since starting from any state the probability of never returning to the same state is zero. To prove this, consider the usual definition of the hitting time for state i as the minimum time required to return to state i when starting at state i, $T_i = \inf\{n \geq 1 : X_n = i | X_0 = i\}$, where X_n is the state of the Markov chain at time n. Denoting as f_{ii}^n the probability $\Pr\{T_i = n\}$, it is not hard to see that $f_{ii}^n = p_i(1-p_i)^{n-1}$, which leads to

$$\Pr\{T_i < \infty\} = \sum_{n=1}^{\infty} f_{ii}^n = \frac{p_i}{1-p_i}\sum_{n=1}^{\infty}(1-p_i)^n = 1$$

Therefore, we conclude that the hitting time for every state is finite w.p.1. This holds for all states of the Markov chain. Moreover, starting from any state i, the probability to visit any other state j in finite steps is 1. To see this, notice that this probability can be computed as follows:

$$\sum_{n=1}^{\infty} \Pr\left\{X_n = j, X_{n-1} \neq j, \ldots, X_1 \neq j | X_0 = i\right\} =$$
$$\sum_{n=1}^{\infty} p_j(1-p_j)^{n-1} = 1. \quad \blacksquare$$

In this paper, we will focus only on the previously described selection of the initial means. Nevertheless, we can possibly describe a different approach to the choice of the initial means that would be investigated in a different venue. The idea is to use local clustering rules that are nonsensitive to the initialization process. To this end, we can employ clustering algorithms based on the *Maximum Entropy Principle* (MEP), described in [8], [10]. More specifically, we can employ an approach also known in the context of

vector quantization for data compression as the *deterministic annealing* (DA) algorithm, which has the following hierarchical property: it progressively seeks finer subclusters inside larger clusters. Due to this property, the DA algorithm is empirically verified to avoid local minima and to be robust with respect to the initial means selection. During the initialization phase, the ensemble learners collect data and cluster them according to local DA algorithms, which aim at minimizing the *free energy*, while employing an appropriate geometric *cooling law*. It is straightforward to describe the corresponding central and ensemble online algorithms developed in this paper for the K-means procedure in the context of the DA algorithm. The local learners can again base their individual clustering decisions on the majority rule. Upon obtaining K means, the initialization phase is terminated. All learners have to agree on the mean values of a preselected node, who is a priori agreed to be the *leader*. In this sense, we have the common initial means required by the ensemble online algorithms described in Section III.

In the sequel, we present some theoretical guarantees for the proposed algorithms based on the first initialization procedure described in Alg. 5.

V. Analysis of the Proposed Algorithms

In this section, we give some theoretical results characterizing the behavior of the described algorithms in Section III with the first proposed means initialization.

Proposition 1: Consider the problem given by (2). Assume that common initial mean values are used by the fusion center implementing the batch $K-$means algorithm and the CRI algorithm. Then the CRI algorithm, implemented to update the past clustering decisions, will give the same solution as the fusion center at time instant $n = N$.

Sketch of Proof: Clearly, at $n = N$ the CRI executes a $K-$means algorithm based on $\mathcal{O}_{[N]}$, which coincides with the operation of the centralized batch $K-$means algorithm implemented by the fusion center. Since the CRI is assumed to update the clustering decisions made so far, the result is obvious. ■

Proposition 2: Let $\mathbf{m}^n_{\text{CRI}}$ denote the cluster mean associated with $\mathcal{O}_n$ as determined by the CRI algorithm. Then the CHTD algorithm will identify the same cluster mean for $\mathcal{O}_n$, almost surely (a.s.) as $M \to \infty$.

Sketch of Proof: Letting $M \to \infty$, the means of the clusters at time $n-1$ obtained by both the CRI and the CHTD algorithms coincide a.s. with the true cluster means due to the fact that the cluster means at time $n-1$ are the sample means of infinite many conditionally independent points. More explicitly, since at every time instant the available data points belong to the same cluster, the strong law of large numbers implies a.s. convergence to the true centroid of the cluster. Reoccurrence of the cluster at later time instants makes no difference in the limit $M \to \infty$. The aforementioned coincidence of the generated centroids with the true means also holds for $n = 1$ with $n-1$ corresponding to the initial mean selection. Additionally, by assumption $\mathbf{m}^n_{\text{CRI}}$ minimizes $\sum_{i=1}^{\infty} d\left(\mathbf{x}^{(i)}_n, \mathbf{m}^n_k\right)$, where $\mathbf{m}^n_k$ is one of the K true cluster means due to the limit $M \to \infty$. Focusing on the CHTD algorithm, its corresponding selected mean $\mathbf{m}^n_{\text{CHTD}}$ minimizes the same distance function at time n. ■

Proposition 3: The clustering decisions up to time $n-1$ from the (local) $K-$means algorithms implemented by each learner at time n are the same as the final clustering decisions of the EMRI algorithm a.s. as $M \to \infty$.

Sketch of Proof: The EMRI algorithm requires that the learners perform local $K-$means algorithms at every time instant over their own data to cluster the datum at time n. Then they exchange their votes and update their local clustering models. Consider now the temporal sequence of decisions made by a local $K-$means algorithm and those made by the majority-voting scheme. As $M \to \infty$, the common initial means tend to the true cluster means w.p.1 due to the strong law of large numbers. This implies that all incoming data to any learner are correctly clustered by a local $K-$means algorithm, since in this context no other (true) mean is closest to each datum. This also implies that the majority voting scheme almost always selects the same (true) cluster mean. ■

Proposition 4: The EMRI and EMHTD solutions converge a.s. to that of the centralized batch $K-$means algorithm for any n as $M \to \infty$.

Sketch of Proof: The proof can be easily derived along the lines of the proofs of Propositions 2 and 3. ■

Remarks:

1) In the simulation section, we demonstrate that the desired M in the aforementioned propositions can actually be very small given the form and the separation of the clusters, as well as the dimensionality of the ambient space $\mathbb{R}^d$.
2) By Propositions 1 and 2 we deduce that the CHTD solution converges a.s. to that of the centralized batch $K-$means as $M \to \infty$.
3) Note the bidirectional implications between Propositions 3 and 4.

VI. Simulations

In this section we provide numerical examples to verify the theoretical guarantees and to demonstrate the operational requirements of the proposed schemes in the previous sections. We assume equiprobable clusters with means separated by $\Delta = 5\sqrt{d}$ distance. The cluster points are selected as $\mathcal{N}(\mathbf{m}_j, 64\mathbf{I}_d)$, where $\mathbf{m}_j$ corresponds to the true cluster mean and the variance 64 corresponds to a value that is large enough to create intersections between the clusters given Δ and the choice of d in each case. The first initialization algorithm for the means in Section IV has been used in both figures.

In Fig. 1, we assume that $N = 100, K = 6, d = 3$. N is selected small so that the behavior of the algorithms against M is highlighted. The vertical axis corresponds to the number of errors made by the CHTD, EMRI and EMHTD algorithms against the centralized batch $K-$means algorithm implemented by the fusion center. We can see that with approximately 60 assumed learners, the CHTD converges to

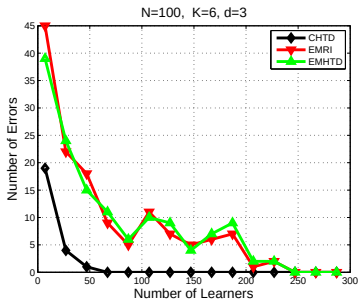

Fig. 1. $N = 100, K = 6, d = 3$: Total number of errors for the CHTD, EMRI and EMHTD algorithms against the central batch $K-$means algorithm.

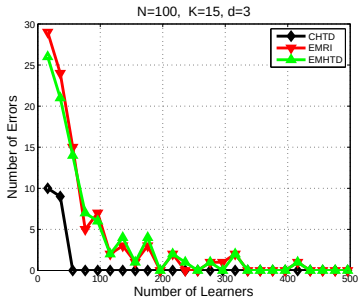

Fig. 2. $N = 100, K = 15, d = 3$: Total number of errors for the CHTD, EMRI and EMHTD algorithms against the central batch $K-$means algorithm.

the solution of the centralized batch $K-$means algorithm. The CRI algorithm with cluster decision updates delivers the same solution at $n = N$ as the centralized batch $K-$means algorithm. Fig. 1 implicitly verifies Propositions 2 and 4. Moreover, the CHTD converges to the centralized batch solution much faster in the number of agents, M, compared to the EMRI and EMHTD algorithms. For the final value of M, i.e., $M = 287$ agents, the execution time of the centralized batch $K-$means algorithm was 0.6020s and for the CHTD algorithm 0.2847s. The execution times of the ensemble schemes (per agent) were 0.1088s and 0.0023s for the EMRI and EMHTD algorithms, respectively.

To examine the behavior of the proposed schemes with an increasing number of clusters, in Fig. 2 we assume that $N = 100, K = 15, d = 3$. The axes are as before. At approximately 70 learners we have convergence of the CHTD solution to that of the centralized batch $K-$means algorithm. For the ensemble schemes, this convergence practically occurs for $M \geq 330$. Fig. 2 implicitly verifies Propositions 2 and 4 as well. Moreover, for the final value of M, i.e., $M = 496$ agents, the execution time of the centralized batch $K-$means algorithm was 2.3358s and for the CHTD algorithm 1.1450s. The execution times of the ensemble schemes (per agent) were 0.2179s and 0.0032s for the EMRI and EMHTD algorithms, respectively.

Finally, Fig. 3 presents the average number of errors made by the EMRI and EMHTD algorithms against the centralized batch $K-$means scheme, when the probability of each learner to receive a datum at n in the same cluster with the

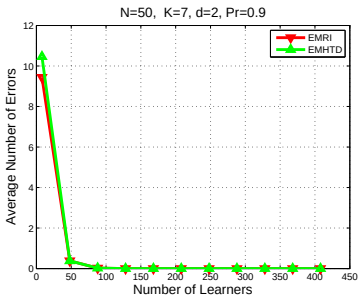

Fig. 3. $N = 50, K = 7, d = 2, \Pr = 0.9$: Average total number of errors for the EMRI and EMHTD against the central batch $K-$means algorithm with probabilistic cluster selection.

others is 0.9. Here, the cluster points are selected according to $\mathcal{N}(\mathbf{m}_j, 25\mathbf{I}_d)$. This plot demonstrates the robustness of the EMRI and EMHTD algorithms against a mismodeling assumption, namely, that the data at each time instant belong to the same cluster. Notice that this mismodeling also affects the centralized $K-$means algorithm. The good performance depends heavily on the parameter choices and on the selected probability.

VII. Conclusions

In this paper, various distributed algorithms for the clustering problem have been proposed, when the local learners are assumed to receive incoming streams correlated on the cluster index. We have demonstrated that a majority voting rule can be used to reduce the communication cost among learners, while delivering the same performance as the corresponding centralized batch clustering algorithm with an increasing number of learners. The results in this paper exhibit the gains that can be achieved by having multiple learners learning together as opposed to isolated learners.

References

[1] R. Agrawal, J. Gehrke, D. Gunopoulos, P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications", *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1998, pp. 94–105.

[2] H. Akaike, "A new look at the statistical model identification", *IEEE Trans. on Aut. Control*, vol. AC-19, no. 6, pp. 716–722, Dec. 1974.

[3] S. Basagni, "Distributed Clustering for Ad Hoc Networks", *Proc. ISPAN*, Fremantle, Australia, 23–25 June, 1999.

[4] V. Cherkassky, F. Mulier, *Learning from Data: Concepts, Theory and Methods*, Wiley, NY, 1998.

[5] Z. Huang, "Extensions to the $k-$means Algorithm for Clustering Large Data Sets with Categorical Data", *Data Mining and Knowledge Discovery*, 2, pp. 283–304, 1998.

[6] J. R. Norris, *Markov Chains*, Cambridge University Press, 1998.

[7] J. B. Predd, S. R. Kulkarni, H. V. Poor, "Distributed Learning in Wireless Sensor Networks", *IEEE Sig. Proc. Mag.*, vol. 23, no. 4, pp. 56–69, July 2006.

[8] K. Rose, "Deterministic Annealing for Clustering, Compression, Classification, Regression and Related Optimization Problems", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2210–2239, Nov. 1998.

[9] A. H. Sayed, S-Y. Tu, J. Chen, X. Zhao, Z. J. Towfic, "Diffusion Strategies for Adaptation and Learning over Networks", *IEEE Sig. Proc. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[10] P. Sharma, S. M. Salapaka, C. L. Beck, "Entropy-Based Framework for Dynamic Coverage and Clustering Problems", *IEEE Trans. on Automatic Control*, vol. 57, no. 1, pp. 135–150, Jan. 2012.

[11] R. Xu, D. Wunsch II, "Survey of Clustering Algorithms", *IEEE Trans. on Neural Networks*, vol. 16, no. 3, pp. 645–678, May 2005.