Distributed and Incremental Clustering Based on Weighted Affinity Propagation

Xiangliang Zhang, Cyril Furtlehner, Michèle Sebag

Distributed and Incremental Clustering Based on Weighted Affinity Propagation

Xiangliang ZHANG [1], Cyril FURTLEHNER and Michèle SEBAG
Laboratoire de Recherche en Informatique, CNRS UMR 8623 & INRIA Saclay
Bâtiment 490, University Paris Sud 11, 91405 - Orsay Cedex, France

Abstract. A new clustering algorithm Affinity Propagation (AP) is hindered by its quadratic complexity. The Weighted Affinity Propagation (WAP) proposed in this paper is used to eliminate this limitation, support two scalable algorithms. Distributed AP clustering handles large datasets by merging the exemplars learned from subsets. Incremental AP extends AP to online clustering of data streams. The paper validates all proposed algorithms on benchmark and on real-world datasets. Experimental results show that the proposed approaches offer a good trade-off between computational effort and performance.

Keywords. Data Clustering, Data Streaming, Affinity Propagation, K-centers

Introduction

Data Clustering, one major task in Unsupervised Learning, is concerned with structuring data items into clusters, enforcing the similarity of items belonging to a same cluster and their dissimilarity w.r.t. items in other clusters. While Unsupervised Learning has been acknowledged a core task of Machine Learning since the beginnings of the field, its theoretical foundations are less mature than those of Supervised Learning.

Many fundamental advances in Data Clustering however have been proposed since the mid 2000s. Ding et al. have highlighted the relationship between K-means and Principal Component Analysis [2]. Based on this relationship, Meila has proposed a stability criterion for assessing clusters and shown the uniqueness of good optima for K-means [14, 15]. In the meanwhile, various criteria have been proposed to set the number K of clusters, e.g. based on Information Theory [18], ROC curve [10] or Dynamic Local Search [11]. Simultaneously, the topic of distance learning has been considered along different perspectives, e.g. related to accurate K-nearest neighbors [19], or enforcing good margins [8].

The present paper is concerned with a new clustering approach, Affinity Propagation (AP) proposed by Frey and Dueck [5]. It is suited to domains where no artefact item (e.g. the barycenter of a set of molecules) can be constructed although a similarity or a distance function can be defined. In such spaces, data clustering is viewed as a combinatorial

[1]Corresponding Author: LRI, Bat.490, University Paris Sud 11, 91405, ORSAY, France; E-mail: xlzhang@lri.fr.

optimization problem: assuming the number K of clusters to be given, the goal is to select K items or *exemplars* from the N-item dataset, such that the average distance from an item to its nearest exemplar, is minimal. This combinatorial optimization problem is tackled using a message passing algorithm, akin to belief propagation, detailed in Section 1.

AP involves the acquisition of the similarity matrix, and the message passing algorithm. While the message passing algorithm converges with $N \log N$ complexity, the similarity matrix is computed with quadratic complexity, thus hindering the scalability of the approach. In [5], the similarity matrix is assumed to be given beforehand, or to involve a small fraction of the item pairs.

The goal of the paper is to address the limitation related to AP quadratic complexity. Firstly, AP is extended to handle duplicated items in a transparent way, resulting in the Weighted AP (WAP) algorithm. Secondly, WAP is used to achieve distributed AP, merging the exemplars independently learned from subsets of the whole dataset (Section 1). Thirdly, an incremental AP algorithm is defined, aimed to data stream clustering (Section 2). The proposed algorithms were validated on benchmark problems and a real-world application (Section 3).

1. Affinity propagation and scalable variants

For the sake of self-containedness, this section first describes the AP algorithm, referring the reader to [5] and [6] for a comprehensive introduction. Two AP extensions are thereafter described, respectively handling the case of weighted items, and the merge of partial solutions.

1.1. Affinity propagation

Let $\mathcal{E} = \{e_1, \ldots e_N\}$ define a set of items, and let $d(e_i, e_j)$ denote the distance or dissimilarity between items e_i and e_j. Letting K denote a positive integer, the K-center problem consists of finding K items in $\mathcal{E}$, referred to as exemplars and denoted $e_{i_1}, \ldots, e_{i_K}$, such that they minimize the sum over all items e_j, of the minimal squared distance between e_j and $e_{i_k}, k = 1 \ldots K$.

The Affinity Propagation approach proposes an equivalent formalization of the K-center problem, defined in terms of energy minimization. Let σ associate to each item e_i its nearest exemplar, then the goal is to find the mapping σ maximizing the functional $E[\sigma]$ defined as:

$$E[\sigma] = \sum_{i=1}^{N} S(e_i, \sigma(e_i)) - \sum_{i=1}^{N} \chi_i[\sigma] \tag{1}$$

where $S(e_i, e_j)$ is set to $-d(e_i, e_j)^2$ if $i \neq j$, and is set to a small constant $-s^*$, $s^* \geq 0$ called *preference* otherwise. The second term in the energy function represent a consistency constraint [2] that if e_i is an exemplar for others, it has to be its own exemplar,

[2] A soft-constraint AP(SCAP) was proposed by [13] to relax the hard constraint that the exemplar selected by other items has to be its self-exemplar. This SCAP algorithm unveils the distributed cluster structure in the data sets instead of regularly shaped clusters. The extension of the proposed algorithms to SCAP will be considered in further studies.

$$\chi_i[\sigma] = \begin{cases} \infty & \text{if } \sigma(\sigma(e_i)) \neq \sigma(e_i) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Aside from the consistency constraints, the energy function thus enforces a tradeoff between the distortion, i.e. the sum $d(e_i, \sigma(e_i))^2$, and the cost of the model, that is $s^* \times |\sigma|$ if $|\sigma|$ denotes the number of exemplars retained. Eq. (1) thus does not directly specify the number of exemplars to be found, as opposed to K-centers. Instead, it specifies the penalty s^* for allowing an item to become an exemplar.

The resolution of optimization problem defined by Eq. (1) is achieved by a message passing algorithm, considering two types of messages: availability messages $a(i,k)$ express the accumulated evidence for e_k to be selected as the exemplar for e_i; responsibility messages $r(i,k)$ express the fact that e_k is suitable to be the exemplar of e_i.

All availability and responsibility messages $a(i,k)$ and $r(i,k)$ are set to 0 initially. Their values are iteratively adjusted by setting:

$$r(i,k) = S(i,k) - \max_{k',k' \neq k} \{a(i,k') + S(i,k')\} \quad (3)$$

$$r(k,k) = S(k,k) - \max_{k',k' \neq k} \{S(k,k')\} \quad (4)$$

$$a(i,k) = min\{0, r(k,k) + \sum_{i',i' \neq i,k} \max\{0, r(i',k)\}\} \quad (5)$$

$$a(k,k) = \sum_{i',i' \neq k} \max\{0, r(i',k)\} \quad (6)$$

The index of exemplar $\sigma(e_i)$ associated to e_i is finally defined as:

$$argmax\{r(i,k) + a(i,k), k = 1 \ldots N\} \quad (7)$$

The algorithm is stopped after a maximal number of iterations or when the exemplars did not change for a given number of iterations.

As could have been expected, Affinity Propagation is not to be seen as a universally efficient data clustering approach. Firstly, as mentioned in the introduction, linear and robust algorithms such as K-means should be preferred to AP in domains where artefact items can be constructed. Secondly, if the desirable number K of clusters is small, then the combinatorial problem can be tackled by brute force (considering all N^K possible solutions). Lastly, and most importantly, AP suffers from a quadratic computational complexity in the number N of items ,hindering its direct use in large-scale applications. The next subsection aims to address this limitation.

1.2. Weighted and distributed AP

To reduce the computational complexity of AP, this paper proposed an algorithm based on a distributed extension of AP, splitting the whole dataset into subsets, then clustering the sets of exemplars extracted from these subsets.

1.2.1. Weighted AP

In order to do so, a preliminary step is to extend AP in order to deal with multiply-defined items. Let the dataset $\mathcal{E}$ be defined as in section 1.1, and let n_i be the number of copies

of item e_i (in the default case, $n_i = 1$ for all i). The S matrix involved in the energy criterion (Eq. (1)) is thus naturally modified as follows. With no difficulty, the penalty $S(i, j)$ of selecting e_j as exemplar of e_i is multiplied by n_i; as e_i actually represents a set of n_i identical copies, the penalty is n_i times the cost of selecting e_j as exemplar for each one of these copies.

Likewise by consistency with Eq. (1), the penalty $S(i, i)$ of selecting e_i as exemplar for itself is set to $s^* + (n_i - 1)\varepsilon_i$. Indeed, let item e_i be unfolded as a set of n_i (almost) identical copies $\{e_{i_1}, \ldots, e_{i_{n_i}}\}$, and let us assume that one of them, say e_{i_1} is selected as exemplar. One thus pays the *preference* penalty s^*, plus the sum of the dissimilarities between e_{i_1} and the other copies in e_i, modelled as $(n_i - 1)\varepsilon_i$. Constant ε_i thus models the average dissimilarity among the n_i copies of e_i.

Formally, let $\mathcal{E}' = \{(e_1, n_i), \ldots, (e_L, n_L)\}$, and define S' as:

$$S'(i,j) = \begin{cases} -n_i d^2(i,j) & \text{if } i \neq j \\ s^* + (n_i - 1) \times \varepsilon_i & \text{otherwise} \end{cases}$$

It is straightforward to show that the combinatorial optimization problem defined as: find σ minimizing

$$E'[\sigma] = \sum_{i=1}^{L} S'(i, \sigma(i)) - \sum_{i=1}^{L} \chi_i[\sigma]$$

is equivalent, for $\varepsilon_i = 0$, to the optimization problem defined by Eq. (1) for $\mathcal{E}$ made of the union of n_i copies of e_i, for $i = 1 \ldots L$.

1.2.2. Distributed AP

The WAP algorithm above is then used to cluster the exemplars constructed from disjoint subsets of the whole dataset, referred to as primary exemplars. Formally, let $\mathcal{E}$ be divided into $\sqrt{N}$ subsets of equal size, noted $\mathcal{E}_i, i = 1 \ldots \sqrt{N}$. Let $\{e_{i_1}, \ldots e_{i_{K_i}}\}$ be the primary exemplars extracted from $\mathcal{E}_i$, with n_{i_j} the number of items in $\mathcal{E}_i$ having e_{i_j} as nearest exemplar. Consider the weighted AP problem defined from $\mathcal{E}' = \{(e_{i_j}, n_{i_j}), i = 1 \ldots \sqrt{N}, j = 1 \ldots K_i\}$.

Note that the construction of $\mathcal{E}'$ is in $\mathcal{O}(N^{\frac{3}{2}})$. Letting K be an upper bound on the number of exemplars learned from every subset $\mathcal{E}_i$, WAP thus achieves the distributed clustering of the exemplars extracted from all $\mathcal{E}_i$ with complexity $\mathcal{O}(N^{\frac{1}{2}} \times K^2)$. The global complexity then is $\mathcal{O}(N \times K^2 + N^{\frac{3}{2}})$.

2. Incremental AP and Data Streaming

This section describes the proposed extension from AP and Weighted AP to Data Stream, one of the hottest topics in Data Mining [4, 1, 7]. It aims to provide a compact description of the data flow [16] and/or the frequent patterns or anomalies thereof. It imposes an additional constraint on Data Mining techniques, the fact that each data item can be seen only once due to the fast rate of acquisition.

The general schema proposed to extend AP to Data Stream (called STRAP , Alg. 1) involves four main steps besides the initialization.

1. The first bunch of data is used by AP to compute the first exemplars.
2. Each new item is compared to the exemplars; if the best fit between the new item and the exemplars is deemed insufficient (section 2.1), the item is put in the reservoir.
3. The restart criterion is triggered if the reservoir size exceeds some threshold, or if some drift in the data distribution is detected (section 2.2).
4. If it is triggered, WAP is restarted with the current exemplars and the reservoir; new exemplars are thus obtained and the associated model is computed (section 2.3).
5. The process goes to step 2.

At every time step, the current model of the data flow is represented by the exemplars and their distribution. The performance of the process, measured from the average distortion and the overall size of the model, is detailed in section 2.4.

Algorithm 1 WAP-based Data Streaming

Datastream $e_1, \ldots e_t, \ldots$**; fit threshold** ϵ
Init
  AP$(e_1, \ldots, e_T) \rightarrow$ Exemplar-based Model
  Reservoir = $\{\}$
for $t > T$ **do**
  Compute Fit(e_t, current model) — section 2.1
  if $Fit < \epsilon$ **then**
    Update model — section 2.1
  else
    Reservoir $\leftarrow e_t$
  end if
  — section 2.2
  if Restart criterion **then**
    Rebuild model by WAP — section 2.3
    Empty reservoir
  end if
end for

2.1. WAP-based Model and Update

In STRAP additional information is needed to see whether a new item should be allocated to an exemplar or considered to be an outlier. The proposed model, inspired from DbScan [3], characterizes each exemplar e_i from a 3-tuple (e_i, n_i, Σ_i), where: n_i is the number of items associated so far to exemplar e_i; Σ_i is the sum of the squared distances between these items and e_i.

This exemplar model enables an additive, computationally efficient update when a new item is associated to any exemplar. The relevancy between current item e_t and current model C_t is defined as $Fit(e_t, C_t) = min_i d(e_t, e_i)$. If $Fit(e_t, C_t)$ is larger than a threshold ϵ, item e_t is put into reservoir. Otherwise, e_t is associated to the nearest exemplar e^*, where $e^* = argmin_i d(e_t, e_i)$. In this case, the selected model (e^*, n^*, Σ^*) is most simply updated by incrementing n^* and adding $d(e_t, e^*)^2$ to Σ^*.

2.2. Restart criterion

The core difficulty in Data Streaming is to deal with outliers and detect the changes in the generative process underlying the stream, referred to as drift. In case of drift, the stream model must be updated. In many application domains, e.g., continuous spaces, the model update can be smoothly achieved through updating the clusters and their centers. AP-relevant domains require the definition of new exemplars. Therefore the data streaming process needs a restart criterion, in order to decide whether the construction of new exemplars from current ones and reservoir should be launched.

Two restart criteria have been considered. The first one is most simply based on the size of the reservoir criterion. When the reservoir is filled with items, the construction of new exemplars based on the current exemplars and the items in the reservoir is launched.

The second criterion is based on a change point detection test. Let us consider the flow of items e_t, and the sequence $p_t = Fit(e_t, C_t)$ of their relevancy measure wrt the current exemplars. If the item generative process is drifting, then sequence p_t should display some change; the restart criterion is triggered upon detecting such a change.

The so-called Page-Hinkley change-point-detection test [17, 9] has been selected as it minimizes the expected detection time for a prescribed false alarm rate. Formally, the PH test is controlled after a detection threshold λ and tolerance δ, as follows:

$$\begin{array}{ll} \bar{p}_t = \frac{1}{t}\sum_{\ell=1}^{t} p_\ell & m_t = \sum_{\ell=1}^{t} (p_\ell - \bar{p}_\ell + \delta) \\ M_t = max\{m_\ell, \ell = 1...t\} & PH_t = (M_t - m_t) > \lambda \end{array}$$

In this latter case, it might happen that the reservoir is filled before the restart criterion is triggered. In such a case, the new item put in the reservoir replaces the oldest one; a counter keeping track of the number of removed reservoir items.

2.3. Model Rebuild

Upon triggering of the restart criterion, Weighted AP is launched on $\mathcal{E} = \{(e_i, n_i)\} \cup \{(e'_i, 1)\}$, where $\{e_i\}$ are the current exemplars together with their size $\{n_i\}$; $\{e'_i\}$ are the items in reservoir with size equal to 1. The question is how to adjust penalties $S(e_i, e_i)$ and $S(e_i, e'_j)$ in order to prevent the number of final exemplars from increasing beyond control, and to avoid sacrificing relevant exemplars to many outliers.

After section 1.2.1, one has:

$$\begin{array}{ll} S(e_i, e_i) = s^* + \Sigma_i & S(e'_j, e'_j) = s^* \\ S(e_i, e_j) = -n_i d(e_i, e_j)^2 & S(e_i, e'_j) = -n_i d(e_i, e'_j)^2 \\ S(e'_j, e_i) = -d(e_i, e'_j)^2 & \end{array}$$

After reconstruct the exemplars by WAP, we need to set the associated model based on the previous model and the reservoir, granted that the items originally involved in the extraction of exemplars are no longer available. Formally, let f be a new exemplar, let $e_1, \ldots e_m$ (respectively $e'_1, \ldots, e'_{m'}$) be previous exemplars (resp. reservoir items) associated to f. With no difficulty, the number n of items associated to f is set to $n_1 + \ldots + n_m + m'$.

The sum of squared distances of the items to f is estimated after an Euclidean model as follows. Let e be an item associated to e_1. After the Euclidean model, e is viewed as

a random item $e_1 + X\vec{v}$, where $\vec{v}$ is a random vector in the unit ball, and X is a random variable with distribution $\mathcal{N}(\mu_1, \sigma_1)$. One has:

$$\begin{aligned} ||f - e||^2 &= ||f - e_1||^2 + ||e_1 - e||^2 - 2\langle f - e_1, X\vec{v}\rangle \\ &= d(f, e_1)^2 + d(e_1, e)^2 - 2X\langle f - e_1, \vec{v}\rangle \end{aligned}$$

Taking the expectation, it comes $E[d(f, e)^2] = d(f, e_1)^2 + \frac{1}{n_1}\Sigma_1$. Accordingly,

$$\Sigma = \sum_{i=1}^{m} \left(n_i d(f, e_i)^2 + \Sigma_i\right) + \sum_{i=1}^{m'} d(f, e'_i)^2$$

2.4. Evaluation criterion

The distortion D of STRAP is computed as follows: i) If some new item e is associated to exemplar e_i, D is incremented by $d(e, e_i)^2$; ii) Otherwise, e is put in the reservoir; after the next restart, the average square distance $\bar{d^2}$ of the reservoir items to the new exemplars is computed, and D is incremented by $\bar{d^2}$ times the number of items put in the reservoir since the last restart[3].

3. Experimental Validation and Discussion

This section firstly presents a comparative validation of distributed clustering with batch clustering on benchmark data set. AP, WAP and K-centers are used on both clustering frameworks. The validation of STRAP compared with distributed WAP on a real world data set is then presented. The size of the real data forbids the use of batch clustering.

3.1. Distributed Clustering Setting

The distributed clustering validation process is as follows:

- Formally, letting N be the total size of dataset $\mathcal{E}$, $\mathcal{E}$ is partitioned into $\sqrt{N}$ subsets of equal size noted $\mathcal{E}_i$.
- Distributed AP (WAP) clustering
  1. On each subset $\mathcal{E}_i$, the preference s_i^* is set to the median of the pair similarities in the subset. AP (WAP) is launched and defines a set of K_i exemplars noted e_{i_j}. Let $\bar{K}$ denote the average of K_i over $i = 1 \ldots \sqrt{N}$.
  2. AP (WAP) then is launched on the primary exemplars $\{e_{i_j}\}$ ($\{(e_{i_j}, n_{i_j})\}$) with preference s^* ranging from minimum to median of the pair similarities.
- Distributed K-centers clustering
  1. Simultaneously, on each subset, K-centers is launched 120 times with $K = \bar{K}$. The best result in terms of distortion is kept. The corresponding exemplars are gathered in $\mathcal{C}$.

[3] This procedure is meant to handle the case of items removed from the reservoir, when the restart criterion is based on the change point detection test, section 2.2.

2. Thereafter, for K set to the number of exemplars obtained by Distributed AP, K-centers is independently applied to $\mathcal{C}$ 20 times. The best results are reported enforcing a fair comparison with AP and WAP (same computation cost).

- Finally, the curves $(K, \text{distortion}(K))$ obtained by AP, WAP and K-centers are compared.

3.2. Validation on benchmarks

13 benchmark datasets kindly provided by E. Keogh have been considered [12], ranging over diverse application domains, e.g. images, videos, texts. On each data set, the distance considered is the Euclidean one.

Table 1. Comparison of K-centers (best of 20 runs) and AP on batch clustering, and comparison of K-centers, AP and WAP on distributed clustering. K depends on AP

Data	N	D	K_AP	Distortion		K_DAP	Distortion of distributed clustering		
				KC	AP		KC	AP	WAP
1	600	60	35	18528	**17522**	/	/	/	/
2	200	150	12	858	**813**	/	/	/	/
3	930	128	47	44088	**42593**	/	/	/	/
4	2250	131	168	100420	**88282** (128 sec)	39	172359	164175 (3 sec)	**160415**
5	442	427	41	90798	**83795**	/	/	/	/
6	1125	128	100	12682	**9965** (21 sec)	23	21525	**20992** (1.4 sec)	21077
7	905	270	62	87426	**78996**	/	/	/	/
8	200	275	9	4529	4651	/	/	/	/
9	112	350	13	15315	**14662**	/	/	/	/
10	121	637	17	37826	**35466**	/	/	/	/
11	143	319	16	20480	**19602**	/	/	/	/
12	200	96	14	2254	**2172**	/	/	/	/
13	781	176	70	412	**216**	/	/	/	/

In Table 1, N is the N. of items, D is the dimension. The subsequent columns report the distortion of batch clustering using AP and K-centers (best out of 20 independent runs). In this batch clustering case, the preference s^* of AP is set to the median similarity among item pairs and K of K-centers is set to the number of exemplars thus obtained with AP, K_AP. The right part of Table 1 shows the performance of distributed clustering using AP, WAP and K-centers on the two largest data sets. K_DAP is the number of exemplars obtained by distributed AP when preference s^* is set to the median similarity. The distributed WAP is forced to produce the same number K_DAP of exemplars by tuning the preference at the second clustering step. The K of K-centers in the exemplars clustering is also set to K_DAP.

These results suggest that AP is more appropriate for complex datasets, where the underlying structure of the domain involves many clusters. In distributed clustering, the clustering of primary exemplars is better performed by WAP than by AP, as the size of exemplars is taken into account. WAP merges the exemplars considering their potential

ability of being a bigger exemplar by passing weighted messages. AP, by contrast, fairly groups the exemplars. Distributed AP significantly decreases the clustering computation time compared with batch clustering, slightly increasing the distortion as a counter part.

Table 2. Comparison of K-centers (best of 20 runs) and AP on batch clustering, and comparison of K-centers, AP and WAP on distributed clustering. K is set to the number of classes and preference s^* is tuned to let AP and WAP have the same number K of clusters

Data	K	N	D	Distortion		Distortion of Hierarchical clustering		
				KC	AP	KC	AP	WAP
4	14	2250	131	189370	**183265**	198658	190496	**189383**
6	15	1125	128	20220	**19079**	20731	20248	**20181**

Note that we cannot compare the distortion of batch clustering with that of distributed clustering in Table 1, because they have different number of clusters. Table 2 shows the results when the number of clusters, K, is fixed to the given number of classes. As could have been expected, distributed AP entails a slightly higher distortion explained by the fact that it uses less information than batch clustering.

3.3. Validation on real-world data

This validation considers a real-world dataset, the set of jobs submitted to the EGEE grid system[4], which will be described first.

3.3.1. Job stream

The considered dataset describes the states of the arrived jobs from 2006-03-14 to 2007-02-06, including 237,087 jobs. After data preprocessing, each job is described by five attributes:

1. the duration of waiting time in a queue;
2. the duration of execution;
3. the number of jobs waiting in the queue when the current job arrived;
4. the number of jobs being executed after the transition of this queue when the current job arrived;
5. the identifier of queue by which the job was transited.

This representation makes it impossible to consider job artefact; the behavior might be significantly different from one queue to another and the expert is willing to extract representative actual jobs as opposed to virtual ones (e.g. executed on queue 1 with weight .3 and on queue 2 with weight .7). The dissimilarity of two jobs x_i and x_j is the sum of the Euclidean distance between the numerical description of x_i and x_j, plus a weight w_q if x_i and x_j are not executed on the same queue. Note that there are around 30% duplicated jobs in the real-world data.

3.3.2. Validation of Distributed AP

Firstly, distributed AP and distributed WAP clustering are validated on this real-world dataset. The whole data, is divided into 486 subsets. Each subset then includes 486 jobs.

[4] http://www.eu-egee.org/

Table 3. Parameters and running time of subset clustering on real-world jobs

Algorithm	parameter	running time	N. of exemplars
K-centers	K = 15	10 mins	7290
AP	s^* = median(S)	26 mins	8444
WAP	s^* = median(S)	10 mins	7531

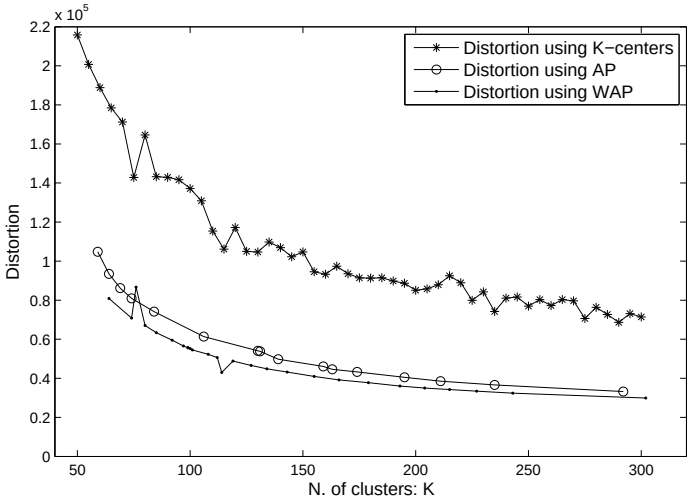

Figure 1. Distortion of distributed clustering using AP, WAP and K-centers on real-world jobs

K-centers, AP and WAP were respectively used on each subset to get exemplars. The parameters, the number of exemplars learned from subsets and running time are shown in Table 3[5]. K-centers is independently launched 120 times to make its running time comparable with WAP. The results with lowest distortion are reported. All the experiments were conducted on a Intel 2.66GHz Dual-Core PC with 2 GB memory by Matlab codes. WAP improves on AP, wrt the number of exemplars and the computation time, due to the duplications in the dataset.

K-centers, AP and WAP are then applied on the primary exemplars learned from the subsets. Distortions on different number K of clusters (Fig. 1) shows that WAP-based distributed clustering has lower distortion than AP-based and K-centers based distributed clustering. The proposed approach scales down the computation complexity of large-size data with roughly one third of the distortion when compared with K-centers.

3.3.3. Validation of STRAP

The job data stream was also used for the validation of STRAP , including a sensitivity analysis wrt threshold ϵ. Fig. 2 displays the overall distortion (defined in Section 2.4) for various values of ϵ. We compared the two restart criteria, the maximum size of reservoir

[5] K of K-centers on each subset was set to 15 such that the total number of exemplars (15×486) is similar to that of WAP (7531)

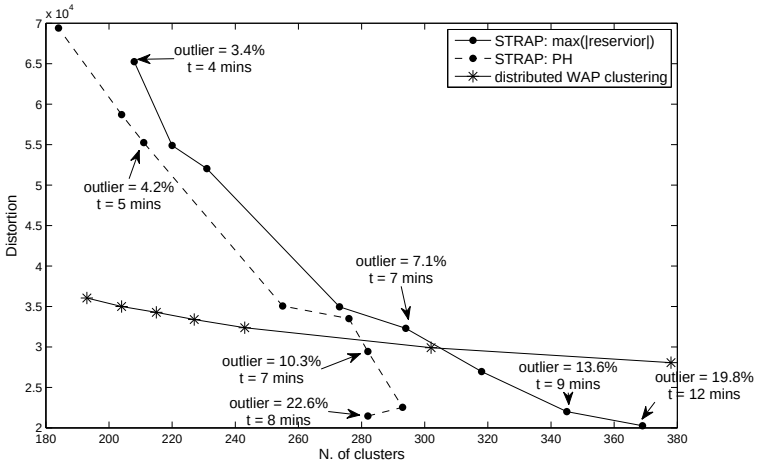

Figure 2. Distortion of STRAP and distributed AP on real-world jobs

(300) and Page-Hinkley (λ=100, δ=0.01). The percentage of jobs put in reservoir and time-cost of STRAP are given.

The distortion of distributed WAP clustering is also shown in Fig. 2. When K ranges in $[205\ 300]$, STRAP distortion is higher than that of distributed WAP. STRAP is also faster (around 7 mins) than distributed WAP (10 mins). When there are more than 10% of stream items put into reservoir, STRAP distortion is much lower because more exemplars are generated. Regarding the restart criteria in STRAP algorithm, Page-Hinkley improves on the other criterion, maximum size of reservoir. It can be explained as there is a higher percentage of outlier in reservoir.

4. Conclusion and Perspectives

In this paper we have extended Affinity Propagation to perform online clustering of data stream. [5] have shown that AP performs better than K-centers clustering especially on sufficiently complex problems. Considering the huge size of data stream (e.g. job error detection in grid computing), the main step is to adapt the scalability of AP.

To overcome the quadratic complexity of AP (caused by the computation of the similarity matrix), we firstly proposed the Weighted AP by aggregating the similar items into one single item.

The second algorithm achieves distributed clustering, by building exemplars from subsets of the initial dataset and aggregating them using WAP. Experimental validation demonstrates that distributed AP is competitive with K-centers on large datasets.

The third proposed algorithm, STRAP , achieves data streaming based on distributed AP. The experimental validation of STRAP shows that it reaches a similar distortion at a significantly lower computation cost. Further work is concerned with bounding the distortion loss due to the distributed computing of exemplars from different subsets.

References

[1] C. Aggarwal, J. Han, J. Wang, and P.S. Yu. A framework for clustering evolving data streams. In *Proceedings of the International Conference on Very Large Data Bases(VLDB)*, pages 81–92, 2003.

[2] C. Ding and X. He. K-means clustering via principal component analysis. In *International Conference on Machine learning (ICML)*, pages 225–232, 2004.

[3] M. Ester. A density-based algorithm for discovering clusters in large spatial databases with noisethe uniqueness of a good optimum for k-means. In *Proceedings of Second International Conference on Knowledge Discovery and Data Mining(KDD)*, pages 226–231, 1996.

[4] W. Fan, H. Wang, and P.S. Yu. Active mining of data streams. In *SIAM International Conference on data mining SDM'2004*, 2004.

[5] B. Frey and D. Dueck. Clustering by passing messages between data points. In *Science*, volume 315, pages 972–976, 2007.

[6] B. Frey and D. Dueck. Supporting online material of clustering by passing messages between data points. In *Science*, volume 315, http://www.sciencemag.org/cgi/content/full/1136800/DC1, 2007.

[7] S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams. In *IEEE Symposium on Foundations of Computer Science*, pages 359–366, 2000.

[8] T. Hertz, A. Bar-Hillel, and D. Weinshall. Boosting margin based distance functions for clustering. In *International Conference on Machine learning (ICML)*, pages 50–58, 2004.

[9] D. Hinkley. Inference about the change-point from cumulative sum tests. In *Biometrika*, volume 58, pages 509–523, 1971.

[10] H. Jahanian, H.S. Zadeh, G.A. Hossein-Zadeh, and M.R. Siadat. Roc-based determination of number of clusters for fmri activation detection. In *Proceedings of SPIE Medical Imaging*, volume 5370, pages 577–586, 2004.

[11] I. Karkkainen and P. Franti. Dynamic local search for clustering with unknown number of clusters. In *16th International Conference on Pattern Recognition (ICPR)*, volume 2, pages 240–243, 2002.

[12] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana. The ucr time series classification/clustering homepage: www.cs.ucr.edu/ eamonn/time_series_data/., 2006.

[13] Michele Leone, , Sumedha, and Martin Weigt. Clustering by soft-constraint affinity propagation: Applications to gene-expression data. *Bioinformatics*, 23:2708, 2007.

[14] M. Meila. Comparing clustering - an axiomatic view. In *International Conference on Machine learning (ICML)*, pages 577–584, 2005.

[15] M. Meila. The uniqueness of a good optimum for k-means. In *International Conference on Machine learning (ICML)*, pages 625–632, 2006.

[16] S. Muthukrishnan. Data streams: Algorithms and applications. In *Found. Trends Theor. Comput. Sci.*, volume 1, pages 117–236. Now Publishers Inc., 2005.

[17] E. Page. Continuous inspection schemes. In *Biometrika*, volume 41, pages 100–115, 1954.

[18] C.A. Sugar and G.M. James. Finding the number of clusters in a dataset: An information-theoretic approach. In *Journal of the American Statistical Association*, volume 98, pages 750–763, 2003.

[19] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, pages 1473–1480. MCambridge, MA: MIT Press, 2005.