Clustering non-stationary data streams and its applications

Amr Abdullatif

DIBRIS, University of Genoa, Italy

amr.abdullatif@unige.it

June 22th, 2016
INTRODUCTION

- Data streams are becoming a major paradigm in the realm of data science (Zliobaite et al, 2012; Lu et al, 2014; Pedrycz et al, 2015).
- They arise from seamlessly observed phenomena in an increasing number of fields, ranging from the web, to wearable sensors, to intelligent transportation systems, to smart homes and cities.
- The size of any collection of "big data" makes single-pass methods a necessity, turning these data effectively into a special case of streaming data.
Data streams are always related to time, although to different degrees:

- **Quasi-stationary phenomena**: longer-term variability (time series), e.g., changes in statistical distribution or a cyclical behavior.
- **Non-stationary phenomena**: given model is expected to be appropriate only in a time interval of where it has been validated/learned.
CONCEPT DRIFT V.S. CONCEPT SHIFT

Proposed terminology:

- **Concept drift**: data locally (in time) identically distributed, i.e., coming from an *underlying source which varies slowly* with respect to window size.

- **Concept shift**: the data distribution may switch from one operating condition to another in a short time, with a *change rate below the sampling rate*.

- **Outlier**: An observation that does not fit a given model is either called novelty.

Assumption: the data of the stream independently sampled.

Robust filtering approaches (Chen, 1993; Xie, 1994) developed for dependent samples (time series) not suitable in this case (Chen, 1993; Xie, 1994).
AIM OF THE PRESENT WORK

Fuzzy clustering for non-stationary data-streams: Clustering process able to adapt to streaming data, by implementing a continuous learning that exploits the input patterns as they arrive.
A set of $c$ fuzzy clusters is represented by means of $c$ centroids $y_j$ ($j \in \{1, \ldots, c\}$).

Each cluster has associated a fuzzy cluster indicator (or membership) function $u_j(x_l) \in [0, 1]$.

Memberships depend on distances between the centroids and the observations $x_l$ ($l \in \{1, \ldots, n\}$).

Many methods are derived as the iterative optimization of a constrained objective function (Bezdek, 1981) often obtained by adding different constraints and penalty terms to the main objective, the mean squared distance or expectation of the distortion:

$$ D = \frac{1}{n} \sum_{l=1}^{n} \sum_{j=1}^{c} u_{lj} \|x_l - y_j\|^2. $$

(1)
The definition of centroids obtained from the necessary minimum condition $\nabla D = 0$ turns out to be common to all methods:

$$y_j = \frac{\sum_{l=1}^{n} u_{lj} x_l}{\sum_{l=1}^{n} u_{lj}}. \quad (2)$$

One exception to this schema is FCM where $u_{lj}^m$ replaces $u_{lj}$.

Usually constraint are imposed on the sum of all membership for any given observation $x_l$,

$$\zeta_l = \sum_{j=1}^{c} u_{lj}. \quad (3)$$

The value $\zeta_l$ can be interpreted as the \textit{total membership mass to clusters} of observation $x_l$. 
MAXIMUM ENTROPY CLUSTERING (Rose et al, 1990)

- $\zeta_l = 1 \rightarrow$ memberships formally equivalent to probabilities ("probabilistic" case).
- Objective function is the Lagrangian

$$J_{ME} = D + H + U,$$

(4)

where

$$H = \beta \sum_{l=1}^{n} \sum_{j=1}^{c} u_{lj} \log u_{lj} \quad (\beta \text{ weight}),$$

(5)

$$U = \sum_{l=1}^{n} \lambda_l (1 - \zeta_l) \quad (\lambda_l \text{ Lagrange multipliers})$$

(6)
The necessary minimum condition $\nabla J_{ME} = 0$ yields

$$u_{ij} = \frac{v_{ij}}{Z_l},$$

where the free membership $v_{ij} = e^{-d_{ij}/\beta}$ is normalized by the Partition Function of observation $x_l$ defined as $Z_l = \sum_j v_{ij}$

The associated optimization procedure proposed for this model includes an “annealing” schedule on $\beta$ (Deterministic Annealing).

At each step of this procedure, previously overlapping clusters may separate when a sufficiently low $\beta$ is reached.

The value $\beta = 0$ corresponds to crisp clustering.
The Possibilistic c-Means (PCM) does not impose any constraint on $\zeta_l$, so memberships are not formally equivalent to probabilities; they represent degrees of typicality.

The objective is:

$$J_{\text{PCM}} = D + W.$$  \hspace{1cm} (8)

In the version presented in 1996 (PCM-II)

$$W = \sum_{l=1}^{n} \sum_{j=1}^{c} \beta_j \left( u_{lj} \log u_{lj} - u_{lj} \right).$$  \hspace{1cm} (9)

The condition $\nabla J_{\text{PCM}} = 0$ yields

$$u_{lj} = e^{-d_{lj}/\beta_j}.$$  \hspace{1cm} (10)
POSSIBILISTIC CLUSTERING (Krishnapuram et al, 1993, 1996)

- Coefficients $\beta_j$ are model parameters (different values for each centroid) playing a role in the representation of data as cluster widths, while in the ME approach the unique coefficient $\beta$ is an optimization parameter, which matters mainly during model building (learning).
- Two heuristics for setting $\beta_j$ are suggested
- An appropriate initialization is required in PCM-II, since it is extremely prone to falling into local minima.
(Masulli & Rovetta, 2006) generalized the membership function:

\[ u_{lj} = \frac{v_{lj}}{Z_l}, \quad (11) \]

where the *free membership* \( v_{lj} = e^{-d_{lj}/\beta_j} \) is normalized by \( Z_l \) is not necessarily equal to \( \zeta_l \).
GRADED POSSIBILISTIC CLUSTERING

\[ Z_l = \left( \sum_{j=1}^{c} v_{lj} \right)^{\alpha}, \quad \alpha \in [0, 1] \subset \mathbb{R} \tag{12} \]

- \( \alpha \) controls the “possibility level” of the model:
  - *Totally probabilistic* model (\( \alpha = 1 \)): representation properties of the method coincide with those of ME.
  - *Totally possibilistic* model (\( \alpha = 0 \)): representation properties of the method equivalent to those of PCM-II.
  - *Intermediate cases* (\( 0 < \alpha < 1 \)): as soon as \( \alpha > 0 \), there is a degree of competition between clusters, as in probabilistic models, but memberships eventually vanish for points sufficiently far away from the centroids (outliers rejection), as in the possibilistic case.
Effect of $\alpha$ in the case of two one-dimensional centroids. The plots show the membership of a point, whose position $x$ varies from $-1$ to $+1$, to each cluster.

Possibilistic ($\alpha = 0$)  
Graded ($\alpha = 0.8$)  
Probabilistic ($\alpha = 1$).
Clustering process that should adapt to streaming data: learning continuously from the input patterns as they arrive.

Data stored in a sliding window $W$ updated every $s$ observations deleting the $s$ oldest patterns and adding $s$ new ones (so at each time, the size of $W$ ($|W| = w$) remains constant and $W$ has an overlap of $w - s$ patterns with the previous one).

Before to incorporating the $s$ incoming observations in the window, we collect them in a test set $S$ we will use for outlier analysis.

Two cases:
- $w > 1$, $s > 1$ (batch learning, batch density estimate), adequate for concept drift
- $w = s = 1$ (online learning, online density estimate), adequate for concept shift
OUTLIERNESS INDEX $\Omega$

- **Outlierness**: membership of an observation to the concept of “being an outlier”.
- The graded possibilistic model provides a direct measure of outlierness $\Omega$ of an observation defined with respect to the whole clustering model:

$$\Omega(x_i) = \max \{0, (1 - \zeta_i)\} \tag{13}$$

where $\zeta_i = \sum_j u_{ij} \in (0, c)$ is the total mass of membership to clusters (in graded possibilistic model it does not necessarily equal 1)

- $\Omega$ measures how much the total mass of membership $\zeta_i$ to clusters is less than one.
OUTLIERNESS INDEX $\Omega$

- Outlierness is modulated by the parameter $\alpha$ ($\alpha \in [0, 1] \subset \mathbb{R}$) controlling the "possibility level":
  - low values of $\alpha \Rightarrow$ possibilistic model: sharper outlier rejection
  - higher values of $\alpha \Rightarrow$ wider cluster regions (and therefore lower rejection)
  - $\alpha = 1 \Rightarrow$ probabilistic model: no able to identify or reject outliers.

- $\zeta_l = \sum_j u_{lj} \in (0, c)$:
  - $\zeta_l \approx 1$ are typical of regions well covered by centroids;
  - $\zeta_l \gg 1$ is very unlikely for good clustering solutions without many overlapping clusters;
  - $\zeta_l \ll 1$ characterizes regions not covered by centroids, and any observation occurring there is an outlier.
Outlier density $\rho \in [0, 1)$ or degree of anomaly of outliers that accounts for the frequency and intensity at which outliers occur. It can be defined in several ways.

$$\rho_{\text{RMS}} = \frac{1}{|S|} \sqrt{\sum_{l \in S} (\Omega(x_l))^2}$$ (14)
LEARNING REGIMES

Characterization of the test set $S$:

- **Concept drift**
  - The source is stationary or changing smoothly and slowly.
  - From pointwise evaluation, based on a single pattern, the variation is indistinguishable from normal data fluctuation, and generally the outlierness value $\Omega = 0$.
  - *Action to be taken*: The model should be incrementally updated to track possible variations. We can call this the baseline learning regime.

- **Outliers**
  - One or few isolated observations are clearly not explained by the model, which means that they have outlierness $\Omega > 0$, but most other observations are well fitted by the current clustering, so that the outlierness density $\rho$ is low.
  - *Action to be taken*: Incremental learning should be paused to avoid skewing clusters with atypical observations.
LEARNING REGIMES

- **Concept shift**
  - Several observations have *outlierness* $\Omega > 0$, and *outlierness density* $\rho$ high.
  - The current clustering is not adequate, i.e., there has been an abrupt change in the source.
  - *Action to be taken*: The old clustering should be replaced by a new one. This is the *re-learning regime*.

- Learning regimes are not clear cut; they are fuzzy concepts.

- We introduce a function $\theta(\rho)$ to control the learning regimes so that the three learning regimes (baseline, no learning and re-learning) can be implemented:
  - $\theta \approx 0$ stability (model stays the same)
  - $\theta \approx 1$ plasticity (model changes completely)
\[ \theta = 1 + \theta_0 \exp \left( -\frac{\rho}{\tau_1} \right) - \exp \left( - \left( \frac{\rho}{\tau_2} \right)^\gamma \right) \] (15)

- \( \theta_0 \) **baseline** value of \( \theta \), used when new data are well explained by the current model (baseline learning regime).
- \( \tau_1 \) scale constant determining the **range of values for which the baseline learning regime should hold** (alike to a time constant in linear dynamical system eigenfunctions).
- \( \tau_2 \) scale constant determining the **range of values for which the re-learning regime should hold**.
- \( \gamma \) exponential gain controlling **how quick the relearning regime should go to saturation**, i.e., to \( \theta \approx 1 \).
The degree of possibility $\alpha$ is assumed to be fixed. $\alpha$ incorporates the a-priori knowledge about the amount of outlier sensitivity desired by the user.

The batch approach is adequate in the many situations where the data generation rate of the source is much higher than its rate of change (concept drift).

Updates to the model are made every $s$ observations, i.e., at times/steps of the form $t = Ns$ ($N = 1, 2, 3, \ldots$).

Initialized on first sample $W(0)$ with complete deterministic annealing optimization (annealing schedule $B = \{\beta_1, \ldots, \beta_b\}$).
BATCH LEARNING

At each subsequent time $t = 1 + Ns$:

1. The clustering model is trained on a training set (window) $W(t)$ (of fixed size $w$).
2. Outlierness density $\rho$ is evaluated on the next $s$ observations (set $S(t)$).
3. Computing $W(t + 1)$ for the next step by removing the $s$ oldest observations and adding $S(t)$ (so that the training set size remains constant).
4. Performing a complete deterministic annealing optimization, with the annealing schedule controlled by the value of $\theta(\rho) \in [0, 1]$:
   - $\theta = 1 \Rightarrow$ the complete $B$ is used ($|B| = b$ optimization steps);
   - $\theta = 0 \Rightarrow$ no training is performed (0 steps);
   - $0 \leq \theta \leq 1 \Rightarrow$ a corresponding fraction $B_\theta$ of the schedule $B$ is used, starting from step number $\lceil b \cdot \theta \rceil$ up to $\beta_b$ (that is, $\lfloor b \cdot (1 - \theta) \rfloor$ steps in total).
Updating rule for a generic centroid (non-stationary data streams):

\[ y_j(t+1) = y_j(t) + \eta \frac{\sum_{l=1}^{W} u_{lj} (x_l - y_j(t))}{\sum_{l=1}^{W} u_{lj}} \]  \hspace{1cm} (16)

\[ = (1 - \eta)y_j(t) + \eta \frac{\sum_{l=1}^{W} u_{lj} x_l}{\sum_{l=1}^{W} u_{lj}} \]  \hspace{1cm} (17)

For \textit{stationary data streams} the distribution of any sample \( W(t) \) is constant w.r.t. \( t \), and therefore its weighted mean \( \frac{\sum_{l=1}^{W} u_{lj} x_l}{\sum_{l=1}^{W} u_{lj}} \) is also constant. In this case

\[ y_j(t \rightarrow \infty) \rightarrow \frac{\sum_{l=1}^{W} u_{lj} x_l}{\sum_{l=1}^{W} u_{lj}} . \]  \hspace{1cm} (18)

With fixed \( \eta \), Eq. (16) computes an exponentially discounted moving average.
Window length $w$ should be

- large enough to have a statistically significant number of observations,
- not as large as to extend outside a region where the source cannot be approximated as (locally) stationary.

The source process is memoryless, so the window size does not need to match the intrinsic dimensionality of any state space.
ONLINE LEARNING

- **Concept drift**: the data generation rate of the source is not too large than its rate of change (e.g., real-time learning in robotics)
- Limit case of the batch method.
- Initialized by taking a first sample $W(0)$ and performing a complete deterministic annealing optimization on it with an annealing schedule $B = \{\beta_1, \ldots, \beta_b\}$.
- At each subsequent time $t$:
  1. The clustering model is trained on a training “set” $W(t)$ of size 1 (i.e., one observation)
  2. $\rho$ is evaluated on the next $s = 1$ observation forming the “set” $S(t)$.
  3. Computing $W(t+1)$ for the next step by replacing the single observation with that in $S(t)$. 
ONLINE LEARNING

- Stochastic approximation procedure (Robbins et al, 1951; Kiefer et al, 1952). Iterative update equations:

\[ y_j(t + 1) = y_j(t) + \eta_t u_{lj}(x_l - y_j) \]  (19)

for each centroid, \( j = 1, \ldots, c \), with learning step size \( \eta_t \).

- The update equation for the memberships is still given by Eqs. (11) and (12).
After the initialization step stochastic annealing step size $\eta_t$

- $\eta_t$ for convergence in the *stationary case* (Robbins et al, 1951):
  \[
  \sum_{t=1}^{\infty} \eta_t = \infty \quad \text{and} \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty \quad (20)
  \]

- However, these conditions obviously do not hold in the *nonstationary case*. 
Strategy adopted in this work: step size $\eta$ directly proportional to $\rho = \Omega$, i.e.,

$$\eta = \eta_0 \cdot \rho$$  \hspace{1cm} (21)

with $\eta_0 \geq 1$ user-selected positive constant.

After initialization, the intensity of updates depends on the degree of outlierness of the current observation. (averaging effect through the stochastic iterative updates)
Possibility degree \( \alpha \) made dependent on \( \rho \), so as to increase centroid coverage when outliers are detected:

\[
\alpha = \alpha_{\min} + \rho (\alpha_{\max} - \alpha_{\min}) = \alpha_{\min}(1 - \rho) + \rho
\]

as it is reasonable to set \( \alpha_{\max} = 1 \)

Eq.22 avoids premature convergence
Parameters $\beta_j$ are model parameters.

To adopt an annealing schedule, we decompose them as follows

$$\beta_j = \beta b_j,$$

where

- $\beta$ optimization parameter for the deterministic annealing procedure;
- $b_j$ relative cluster scale (model parameter)
SYNTHETIC PROBLEM

- Two-dimensional stream of artificial data was generated using the Matlab program `ConceptDriftData.m` (Ditzler et al, 2013)
- Gaussians problem including *concept drift*: four Gaussians in the plane with the same dispersion and gradually moving centers. - The generated stream includes 2400 observations.
- *Concept shift* was added by introducing two discontinuities in the sequence:
  - The data were obtained by deleting data from a longer stream and joining the two halves at 50% (i.e., at pattern 1200).
  - The last 25% of the stream was shifted by adding an offset of 0.2 to both components of the data, so that centers move abruptly to the north-east after pattern 1800.
### Synthetic Problem

#### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Batch</th>
<th>Online</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training window size</td>
<td>( w )</td>
<td>200</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Test window size</td>
<td>( s )</td>
<td>30</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Possibility degree</td>
<td>( \alpha )</td>
<td>0.7</td>
<td>0.7</td>
<td>(1)</td>
</tr>
<tr>
<td>Num. annealing steps</td>
<td>( b )</td>
<td>20</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>( \beta ) schedule</td>
<td></td>
<td>linear</td>
<td>linear</td>
<td>(2)</td>
</tr>
<tr>
<td>Starting value for annealing</td>
<td>( \beta_1 )</td>
<td>0.05</td>
<td>0.05</td>
<td>(2)</td>
</tr>
<tr>
<td>Ending value for annealing</td>
<td>( \beta_b )</td>
<td>0.002</td>
<td>0.002</td>
<td>(2)</td>
</tr>
<tr>
<td>Density estimation function</td>
<td>( \rho(\Omega) )</td>
<td>( \rho_{RMS} )</td>
<td>( \Omega )</td>
<td></td>
</tr>
<tr>
<td>Coefficient for discounted avg.</td>
<td>( \lambda )</td>
<td>—</td>
<td>0.01</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( \theta_0 )</td>
<td>0.3</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( \tau_1 )</td>
<td>0.01</td>
<td>0.01</td>
<td></td>
</tr>
<tr>
<td>Parameters for computing ( \theta )</td>
<td>( \tau_2 )</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( \gamma )</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

(1) For online: minimum value, maximum is 1.
(2) For online: only in the batch initialization phase.
SYNTHETIC PROBLEM
Tracking error: Absolute difference between distortion w.r.t. true centroids

- **batch**
- **online**
- **k-means**
- **TRAC-STREAMS (Nasraoui et al, 2006)**

Introduction
Central Clustering Models
Proposed Approach
Experimental Validation

Clustering non-stationary data streams
Layered Ensemble Model for Short Term Traffic Flow Forecasting

- Short-term urban traffic forecasting: Given observations of relevant traffic indicators on an urban road network, forecast the observation in the near future.
- Two different data sets were used for experimentation (UK motorway, and Genova Urban area of the city of Genoa (north-west of Italy)).
- Traffic parameters were obtained from actual observations and several days of traffic were estimated by using the SUMO open source traffic simulator (Krajzewicz et al, 2012).
- LEM model combines Artificial Neural Networks and Graded Possibilistic Clustering obtaining an accurate forecast of the traffic flow rates with outlier detection.
Area of interest and the graph used to model it
For any forecaster model there are some issues that affect its performance:

**Lag period:** Proper selection of the lag period which is crucial because it affects the correct representation of the traffic flow in time (Redundancy vs Irrelevance).

**Historical period:** This refers to the number of observation patterns that will be used to train the forecasters.

**Outliers:** measuring the degree of outlierness of each pattern to improve the forecaster accuracy.
Training stage of the LEM model

Flow chart of the training stage in the proposed model
Test stage of the LEM model

Flow chart of the test stage in the proposed model:

1. Test patterns
2. Test input pattern $i$
3. Graded Possibilistic Clustering (GPCM)
4. $\zeta_i < \min(\zeta_i)$
   - no: Forecaster$e$
   - yes: Flag pattern $i$ as an outlier
5. Forecaster$_1$
6. Forecaster$_2$
7. Forecaster$_3$
8. $y_i = (y_{1u1} + y_{2u2} + y_{3u3} + ... + y_{eu}) / \zeta_i$
## LEM Model Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>UK Data Set</th>
<th>Genoa Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lag period</td>
<td>1 day (96 observations)</td>
<td>25 min (5 observations from 3 road links)</td>
</tr>
<tr>
<td>Observation period</td>
<td>15 min</td>
<td>5 min</td>
</tr>
<tr>
<td>Historical period for training</td>
<td>9 months</td>
<td>6 hours</td>
</tr>
<tr>
<td>Test period (forecasting period)</td>
<td>3 months</td>
<td>3 hours</td>
</tr>
<tr>
<td>Validation</td>
<td>10-fold cross validation</td>
<td>10-fold cross validation</td>
</tr>
<tr>
<td>Forecaster</td>
<td>Artificial Neural Network</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>Number of layers</td>
<td>3 (input, hidden, output)</td>
<td>3</td>
</tr>
<tr>
<td>Neural Network architecture</td>
<td>95-10-1</td>
<td>4-10-1</td>
</tr>
<tr>
<td>Number of clusters</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>$0 &lt; \alpha &lt; 1$</td>
<td>$0 &lt; \alpha &lt; 1$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>.1</td>
<td>.01</td>
</tr>
</tbody>
</table>
Results - Genova data-set

![Graph showing traffic flow over time with target and output markers.](image_url)
Results - UK data-set

UK data set: R=0.98006 (drop rate=0)

Output = 0.95*Target + 0.011

Target

Data
Fit
Y = T
The results show that we were able to control the degree of outliers and improve the accuracy rate by choosing an appropriate value of $\alpha$.

\[ \text{droprate} = 1 - \left( \frac{\text{Number of output patterns}}{\text{Number of target patterns}} \right). \]  

(24)