# MODIFIED K-MEAN CLUSTERING METHOD OF HMM STATES FOR INITIALIZATION OF BAUM-WELCH TRAINING ALGORITHM

*Pauline Larue*[1], *Pierre Jallon*[1], *Bertrand Rivet*[2]

[1]CEA LETI - MINATEC Campus
Grenoble, France
email: pierre.jallon@cea.fr

[2]GIPSA-lab, CNRS-UMR5216
Grenoble University, F-38402 Grenoble Cedex, France
email: bertrand.rivet@gipsa-lab.grenoble-inp.fr

## ABSTRACT

Hidden Markov models are widely used for recognition algorithms (speech, writing, gesture, ...). In this paper, a classical set of models is considered: state space of hidden variable is discrete and observation probabilities are modeled as Gaussian distributions. The models parameters are generally estimated with training sequences and the Baum-Welch algorithm, i.e. an expectation maximization algorithm. However this kind of algorithm is well known to be sensitive to its initialization point. The problem of this initialization point choice is addressed in this paper: a model with a very large number of states which describe training sequences with accuracy is first constructed. The number of states is then reduced using a k-mean algorithm on the state. This algorithm is compared to other methods based on a k-mean algorithm on the data with numerical simulations.

## 1. INTRODUCTION

Hidden Markov models (HMM) are widely used for statistical modeling of signals having a temporal structure. They are based on two statistics processes: the state variable process and the observation process. The first one $X$ is the so-called state variable and is a Markov chain, assumed discrete in this paper. It is not observed but it is fully characterized by transitions and initialization probabilities. In many recognition algorithms (e.g., speech [1], writing recognition [2] , gesture recognition [3], ...), this variable is used to describe the temporal structure of the signals to model. In particular if these signals can be described as sequences of (shorter) stationary signals, a very particular set of models can be used: the left-right models; models whose transition probabilities satisfy the following constraints:

$$\forall(i_1,i_2),\ i_2<i_1,\quad p\left(X_t=i_2|X_{t-1}=i_1\right)=0,$$

which means that the state variable can only be a, increased sequence. The second process $Y$ is the so-called observation process and is assumed to be an independent process conditionally to $X$ [1]. For each state $n\in\{0,\cdots,N-1\}$ a probability density function (p.d.f.) $p\left(Y|X=n\right)$ is defined, where $N$ is the number of different values taken by $X$ or the number of hidden states. $Y$ is assumed to be a continuous variable and its p.d.f is modeled for each state as a Gaussian distribution.

The following notations will be used in the rest of this paper:

- the initial probability of state is denoted $\pi_i = p\left(X_0=i\right), \forall i\in\{0,\cdots,N-1\}$,
- the transition probability $a_{i_1,i_2}$ between states $i_1$ and $i_2$ is defined by $a_{i_1,i_2}=p\left(X_t=i_2|X_{t-1}=i_1\right)$, $\forall(i_1,i_2)\in\{0,\cdots,N-1\}\times\{0,\cdots,N-1\}$.

The observation probabilities are described, for state $\#i$ with two variables $\boldsymbol{\mu}_i$ (mean vector) and $\Sigma_i$ (covariance matrix). Finally the whole set of parameters of a HMM with $N$ states is denoted as $\lambda_N$:

$$\lambda_N=\Big\{\{\pi_i\}_i,\{a_{i_1,i_2}\}_{i_1,i_2},\{\boldsymbol{\mu}_i,\Sigma_i\}_i\Big\}.$$

In general $\lambda_N$ is estimated using training sequences. Given $K$ observation sequences of length $T_k$, $Y_{k,0:T_k-1}=\{Y_{k,0}\cdots,Y_{k,T_k-1}\}$, the optimal set of parameters $\lambda_N$ is defined as:

$$\hat{\lambda}_N=\underset{\lambda_N}{\operatorname{argmax}}\left\{\prod_{k=0}^{K-1}p\Big(Y_{k,0:T_k-1}|\lambda_N\Big)\right\}.$$

Without additional assumptions, this problem can not be solved analytically. The mainly used technique to estimate $\lambda_N$ is the expectation maximization (EM) algorithm [4] trough the forward-backward method so-called Baum-Welch algorithm [5]. Starting from a preliminary set of parameters $\lambda_N^{(0)}$, the algorithm estimates the set of parameters in an iterative manner - denoted as $\lambda_N^{(s)}$ at step $\#s$ - such as $\prod_{k=0}^{K-1}p\left(Y_{k,0:T_k-1}|\lambda_N^{(s)}\right)$ increases with respect to $s$. $\hat{\lambda}_N$ is then estimated as:

$$\hat{\lambda}_N=\lim_{s\to\infty}\lambda_N^{(s)}.$$

However convergence to a global maxima is not ensured and this algorithm is well known to be very sensitive to its initialization point $\lambda_N^{(0)}$ [6]. Several initialization methods have hence been proposed in the literature. A k-mean algorithm on the data can be used to cluster observations $Y_{k,0:T_k-1}$ [1, 7] or several sets of parameters to perform the training can be used [8]. For left-right models, constrained clustering techniques can also be used with a k-mean algorithm on the observation data [9].

In this paper, an alternative approach is proposed to estimate a first set $\lambda_N^{(0)}$. A set of parameters is first estimated with a very large number of states ($M \gg N$) to describe all training sequences in a very accurate way. However, this set of parameters can not be used in practice due to over-learning problems and computational time issues. As a consequence, the number of states has to be reduced to a much smaller number to overcome these difficulties. An unsupervised clustering algorithm (based on the k-mean algorithm) on the observation p.d.f is hence proposed to do this operation while keeping the training signals description accuracy. Transition and initialization probabilities remain to be estimated with the Baum-Welch algorithm. The proposed method is derived in two cases: unconstrained and left-right models.

The paper is structured as follows. Section 2 describes the mainlines of the algorithm. The k-mean algorithm operations are detailed in section 3. The proposed method performance is compared to other methods's based on simulations in section 4. Finally, Section 5 concludes this paper.

## 2. INITIALIZATION ALGORITHM DESCRIPTION

The aim of the algorithm is to provide an initialization set of parameters $\lambda_N^{(0)}$ close enough to $\hat{\lambda}_N$ to ensure a good convergence of the learning algorithm.

It only focuses on the observation probabilities parameters, which aims at finding $N$ sets of observation probabilities parameters $\{\boldsymbol{\mu}_i^{(c)}, \Sigma_i^{(c)}\}_{i\in\{1,\cdots,N\}}$, i.e. $N$ Gaussian distributions, to describe in an accurate way the training sequences. It is worth noting that transition probabilities $a_{i_1,i_2}$ and initialization probabilities $\pi_i$ are not estimated (although constrained for the left-right models), this operation being done by the Baum-Welch algorithm. The following values are hence used:

- for unconstrained models, transition probabilities are set to $a_{i,i} = 0.8, \forall i \in \{0,\cdots,N-1\}$ and $\forall i_1 \neq i_2$, $a_{i_1,i_2} = 0.2/(N-1)$, and initialization probabilities $\pi_i = 1/N$, $\forall i \in \{0,\cdots,N-1\}$;
- for left-right models, transition probabilities are set to $a_{i,i} = 0.8$, $\forall i \in \{0,\cdots,N-2\}$ and $a_{N-1,N-1} = 1$, $a_{i,i+1} = 0.2$ and other values are set to 0, finally initialization probabilities $\pi_0 = 1$ and $\forall i \in \{1,\cdots,N-1\}, \pi_i = 0$.

Concerning the $N$ p.d.f. of hidden states, the estimation is performed in two steps: first $M$ p.d.f. are estimated which accurately describe the signal (Subsection 2.1). These $M$ distributions are then reduced to $N$ (Subsection 2.2).

### 2.1 Initialization step

Given a set of $K$ training sequences, $Y_{k,0:T_k-1}$, $k \in \{1,\cdots,K\}$, $M$ Gaussian distributions which accurately describe data can be estimated as follows. Each training sequence $Y_k$ is split into several segments of length $P$, overlapping or not and covering its time support. Each segment is then modeled as a Gaussian signal which parameters $(\boldsymbol{\mu}_{k,j}, \Sigma_{k,j})$ are estimated using classical methods: $\boldsymbol{\mu}_{k,j}$ being the mean of the related signal segment and $\Sigma_{k,j}$ its covariance matrix. Without restriction, it is assumed that the distributions are sorted with respect to the $Y_k$ segment delay. For instance, with non-overlapping segments, each couple of values ($0 \leq k \leq K-1$ and $0 \leq j < T_k/P$) can be estimated as:

$$\begin{aligned}\boldsymbol{\mu}_{k,j} &= \frac{1}{P}\sum_{p=0}^{P-1} Y_{k,jP+p},\\ \Sigma_{k,j} &= \frac{1}{P}\sum_{p=0}^{P-1}\left(Y_{k,jP+p}-\boldsymbol{\mu}_{k,j}\right)\left(Y_{k,jP+p}-\boldsymbol{\mu}_{k,j}\right)^T,\end{aligned}$$

where $.^T$ is the transpose operator. $P$ should be chosen small enough so that each signal segment can be considered as stationary, and large enough to ensure a good estimation of mean vector and covariance matrix.

It is possible after this step to collect all these values to build the initial set of parameters $\lambda_M^{(0)}$. It is worth noting that this initial set $\lambda_M^{(0)}$ describes in a very accurate way the training sequences: indeed, each training signal segment being described by one of the $M$ states. However, and as already mentioned in the introduction section, this model can not be used in practice because of over-learning problems and of computational time issues.

### 2.2 Clustering step

The second step of the algorithm is hence to approximate these $M$ Gaussian distributions with $N$ ones ($N \ll M$). This operation is performed using a k-mean algorithm on these Gaussian distributions.

Let $\mathcal{N}(\boldsymbol{\mu}_{k,j}, \Sigma_{k,j})$ denotes the Gaussian distribution p.d.f. with mean vector $\boldsymbol{\mu}_{k,j}$ and covariance matrix $\Sigma_{k,j}$. Moreover, let us refer as 'center' the distribution which characterizes a cluster, of which parameters are labelled by $(c)$: $\mathcal{N}(\boldsymbol{\mu}_i^{(c)}, \Sigma_i^{(c)})$.

The k-mean algorithm works iteratively as follows. Given an initial set of $N$ centers $\left\{\mathcal{N}\left(\boldsymbol{\mu}_{i,0}^{(c)}, \Sigma_{i,0}^{(c)}\right)\right\}_{1\leq i\leq N}$, the following two steps are performed at each iteration $\#s$:

1. Association: each distribution $\mathcal{N}\left(\boldsymbol{\mu}_{k,j}, \Sigma_{k,j}\right)$ is associated to a center in the set $\left\{\mathcal{N}\left(\boldsymbol{\mu}_{i,s}^{(c)}, \Sigma_{i,s}^{(c)}\right)\right\}_{1\leq i\leq N}$;
2. Center update: based on the previous association, updated centers are estimated $\left\{\mathcal{N}\left(\boldsymbol{\mu}_{i,s+1}^{(c)}, \Sigma_{i,s+1}^{(c)}\right)\right\}_{1\leq i\leq N}$.

The $N$ centers are finally estimated as:

$$\forall i\ ,\ \mathcal{N}\left(\boldsymbol{\mu}_i^{(c)}, \Sigma_i^{(c)}\right) = \lim_{s\to\infty}\mathcal{N}\left(\boldsymbol{\mu}_{i,s}^{(c)}, \Sigma_{i,s}^{(c)}\right).$$

The initial set of $N$ centers depends on the type of HMM (unconstrained or left-right models). For unconstrained models, the $M$ distributions are sorted according to the first component of their mean vector. The set of $M$ distributions is split into $N$ consecutive sets and a center is estimated for each set according to the k-mean algorithm second step as described above. For

left-right models, each training sequence distributions are split into $N$ disjointed segments. First segment is then associated to center #0, second one to center #1, and so on. Initialization centers are also estimated using the k-mean algorithm second step.

It is worth noting that the first step requires to adapt the Euclidian distance to associate each distribution to a cluster. This point is discussed in Section 3.1 for unconstrained models and in Section 3.2 for left-right models. Furthermore, the second step is described in Section 3.3.

## 3. MODIFIED K-MEAN ALGORITHM

This section details the practical proposed considerations of the proposed modified k-mean clustering algorithm.

### 3.1 State to center distance computation for unconstrained models

The unconstrained model case is first considered: each Gaussian distribution is associated to a center in an independent way. As both distribution and center are Gaussian distributions, the Kullback-Leibler divergence is proposed to estimate the distance between both p.d.f.. This divergence is defined as:

$$D_{KL}\Big(p_1(u)\|p_2(u)\Big) = \int p_1(u)\log\left(\frac{p_1(u)}{p_2(u)}\right)du.$$

For Gaussian distributions, this latter expression is proportional to

$$D_{KL}\left(\mathcal{N}\Big(\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\Big)\middle\|\mathcal{N}\Big(\boldsymbol{\mu}_i^{(c)},\Sigma_i^{(c)}\Big)\right) \propto \ln\frac{\det\Sigma_i^{(c)}}{\det\Sigma_{k,j}} + \mathrm{Tr}\left(\Big(\Sigma_i^{(c)}\Big)^{-1}\Sigma_{k,j}\right) + \Big(\boldsymbol{\mu}_i^{(c)}-\boldsymbol{\mu}_{k,j}\Big)^T\Big(\Sigma_i^{(c)}\Big)^{-1}\Big(\boldsymbol{\mu}_i^{(c)}-\boldsymbol{\mu}_{k,j}\Big), \quad (1)$$

where Tr is the trace operator and det is classically the determinant. Finally, $\forall(k,j)$, the related distribution $\mathcal{N}\big(\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\big)$ is associated to the closest center $i*$ which satisfies:

$$i_* = \underset{i}{\mathrm{argmin}}\, D_{KL}\left(\mathcal{N}\Big(\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\Big)\middle\|\mathcal{N}\Big(\boldsymbol{\mu}_i^{(c)},\Sigma_i^{(c)}\Big)\right).$$

### 3.2 State to center distance computation for left-right models

Compared to previous unconstrained model, left-right assumption sets an additional constraint. For all $k$, if distribution $\mathcal{N}\big(\boldsymbol{\mu}_{k,j_0},\Sigma_{k,j_0}\big)$ is associated with center $\#i_0$, then for all $j_1 > j_0$, $\mathcal{N}\big(\boldsymbol{\mu}_{k,j_1},\Sigma_{k,j_1}\big)$ cannot be associated to any center $\#i$, $i < i_0$. In other words, the distribution – center association has to be jointly done for each training sequence (rather than in an independent way).

For sequence $\#k$, let consider therefore the set of $N+1$ break indices $I_k(0),\cdots,I_k(N)$, defining the $N$ states. This set must verify due to the left-right constrain that $I_k(0)=0$, $I_k(N)=M-1$ and for two states $i_1$ and $i_2$, $\forall\, i_1<i_2\in\{0,\cdots,N\}\times\{0,\cdots,N\}$, $I_k(i_1)<I_k(i_2)$. Finally, $\forall\,(k,j)$, the related distribution $\mathcal{N}\big(\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\big)$ is associated to the closest center $i_*$ which satisfies:

$$i_* = i \;\Big|\; I_k(i)\le j < I_k(i+1).$$

The set of break indices is previously estimated as

$$\Big\{\hat{I}_k(1),\cdots,\hat{I}_k(N-1)\Big\} = \underset{I_k(1),\cdots,I_k(N-1)}{\mathrm{argmin}}\; D\big(I_k(1),\cdots,I_k(N-1)\big),$$

where

$$D\big(I_k(1),\cdots,I_k(N-1)\big) = \sum_{i=0}^{N-1}\sum_{j\in\mathcal{I}_k(i)} D_{KL}\left(\mathcal{N}\Big(\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\Big)\middle\|\mathcal{N}\Big(\boldsymbol{\mu}_i^{(c)},\Sigma_i^{(c)}\Big)\right),$$

with $\mathcal{I}_k(i)=\{I_k(i),\cdots,I_k(i+1)-1\}$ and $D_{KL}(\cdot\|\cdot)$ defined by (1). Note that the two extrema $I_k(0)$ and $I_k(N)$ do not require to be optimized since $I_k(0)=0$ and $I_k(N)=M-1$.

### 3.3 Center estimation

The third and last issue to solve is the computation of the updated center parameters once each distributions have been associated to one cluster. Consider therefore the center $\#i$ and $I_i^{(c)}$ the associated set of distributions. A random variable $Z$ in this set can be written as:

$$Z = \sum_{(k,j)\in I_i^{(c)}} \delta(H-(k,j))N_{k,j}$$

where $N_{k,j}$ is a Gaussian random variable with mean and variance $\{\boldsymbol{\mu}_{k,j},\Sigma_{k,j}\}$, $\delta(u)$ the Dirac delta function and $H$ is a hidden variable equal to $(k,j)$ if $Z$ shares $N_{k,j}$ distribution.

The i-th center parameters $\big\{\boldsymbol{\mu}_i^{(c)},\Sigma_i^{(c)}\big\}$ are estimated as mean and covariance of $Z$. It is straightforward to check that:

$$\boldsymbol{\mu}_i^{(c)} = \frac{1}{\mathrm{card}\Big(I_i^{(c)}\Big)}\sum_{(k,j)\in I_i^{(c)}}\boldsymbol{\mu}_{k,j}$$

and

$$\Sigma_i^{(c)} = \frac{1}{\mathrm{card}\Big(I_i^{(c)}\Big)}\sum_{(k,j)\in I_i^{(c)}}\Big(\Sigma_{k,j}+\boldsymbol{\mu}_{k,j}\boldsymbol{\mu}_{k,j}^T\Big) - \boldsymbol{\mu}_i^{(c)}\;\boldsymbol{\mu}_i^{(c)\,T}$$

where $\mathrm{card}\big(I_i^{(c)}\big)$ is the number of elements in set $I_i^{(c)}$.

## 4. NUMERICAL RESULTS

This section presents the results achieved by the proposed method with several kind of configurations: toy example and simulated signals.

| Scenario | $\mu_0$, $\mathbf{\Sigma_0}$ | $\mu_1$, $\mathbf{\Sigma_1}$ | $\mu_2$, $\mathbf{\Sigma_2}$ | $\mu_3$, $\mathbf{\Sigma_3}$ |
|---|---|---|---|---|
| #1 | $\left[\begin{smallmatrix}2\\2\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&2\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}6\\6\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}2&0\\0&1\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&2\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}5\\5\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}2&0\\0&1\end{smallmatrix}\right]$ |
| #2 | $\left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}5&0\\0&5\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}3\\3\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}6\\6\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}5&0\\0&5\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}5\\5\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$ |
| #3 | $\left[\begin{smallmatrix}1\\1\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}5&0\\0&5\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}2\\2\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}3\\3\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}5&0\\0&5\end{smallmatrix}\right]$ | $\left[\begin{smallmatrix}4\\4\end{smallmatrix}\right]$, $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$ |

Table 1: Simulation scenarios.

### 4.1 Evaluation method

The proposed algorithm performance (denoted "k-mean on Gaussian laws") have been estimated using numerical simulations and are compared to the performance of the classical k-mean algorithm applied directly on data (denoted "k-mean on data"). Moreover, the influence of the left-right constrain is investigated (referred as "no constrained" or "left-right"). Nevertheless, constrained k-mean on data for left-right models has not been tested because the number of sets of break indices to test is prohibitive leading thus to an excessive computational time. As mentioned in the introduction, the aim of these algorithms is to estimate a set of parameters $\hat{\lambda}_N^{(0)}$ so that the convergence of the Baum-Welch algorithm is improved. This latter algorithm is an iterative algorithm computing in an iterative manner the set of parameters $\lambda_N^{(s)}$ at step $\#s$ such as $\prod_{k=0}^{K-1} p\left(Y_{k,0:T_k-1}|\lambda_N^{(s)}\right)$ increases with respect to $s$.

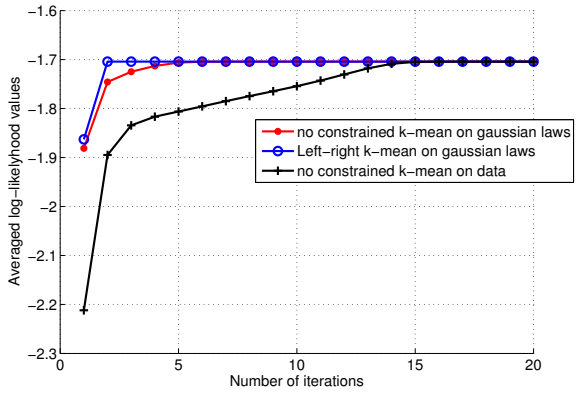
For a given set of training sequences $\left\{Y_{k,0:T_k-1}\right\}_k$, the sets of parameters $\hat{\lambda}_N^{(0)}$ estimated with the different initialization algorithms are hence compared with the value reached and the convergence rate of the following likelihood function

$$\ell(s) = \log\left(\prod_{k=0}^{K-1} p\left(Y_{k,0:T_k-1}|\lambda_N^{(s)}\right)\right).$$

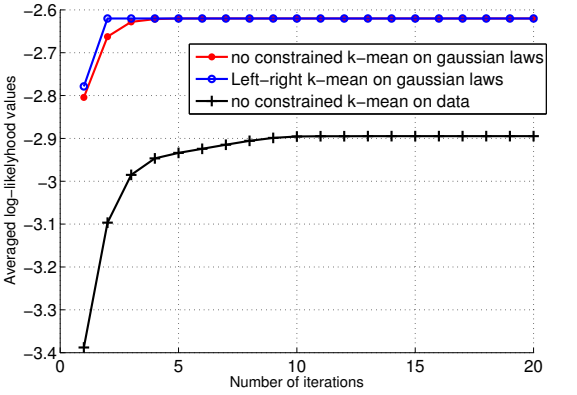
### 4.2 Toy example

Signals have been generated as the concatenation of 4 segments signals so that $Y_{0:T} = \{Y^{(0)}_{0:T_0-1}, Y^{(1)}_{0:T_1-1}, Y^{(2)}_{0:T_2-1}, Y^{(3)}_{0:T_3-1}\}$. Each of these four signals is generated using a Gaussian distribution: for the $p-th$ signal, with mean vector $\boldsymbol{\mu}_p$ and covariance matrix $\Sigma_p$. Their time lengths $T_p$, $\forall p$ are randomly chosen using a truncated Gaussian distribution of mean 500 and standard deviation 100 (keeping only positive value).

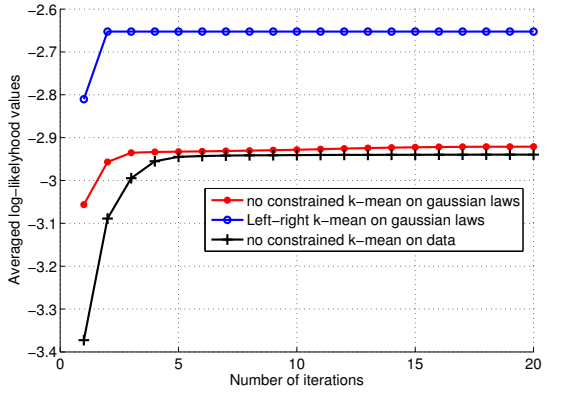
Three sets of parameters $\{\boldsymbol{\mu}_p, \Sigma_p\}_p$ have been tested corresponding to 3 scenarios resumed in Table 1. As one can see, these three scenario vary from these difficulty. Indeed, scenario #1 is the easiest since the 4 states are quite separated. Scenario #2 is a little bit more complex than scenario #1 since states 1 and 2, and 3 and 4 overlap. Finally, scenario #3 is the more confusing model since all the state largely overlap. For each scenario, 100 realizations have been generated, and for each realization 5 sequences have been used for the initialization and the training algorithms. $P = 20$ has been used for estimation of $M$ Gaussian distributions prior to k-mean algorithm.




(a) Scenario #1




(b) Scenario #2




(c) Scenario #3

Figure 1: Performance evaluation achieved on the toy example.

Fig. 1 presents the achieved results as the averaged values of $\ell(s)$ over the 100 realizations. For the first scenario (Fig. 1(a)), the 3 tested algorithms ('left-right k-mean on Gaussian distributions', 'unconstrained k-mean on Gaussian distributions' and 'on data') converge to the same value, but k-mean on Gaussian distributions

| | Conf. #1 | Conf. #2 |
|---|---|---|
| DTW | 77.1 | 59.5 |
| HMM | 88.1 | 69.9 |

Table 2: Classification accuracy in percentage [%].

algorithms converge much faster than classical k-mean initialization. For the second scenario (Fig. 1(b)), the two proposed k-mean on Gaussian distributions algorithms (left-right and unconstrained) converge to the same point with a few iterations whereas the classical k-mean on data algorithm fails into a local maxima. Finally for the most complex scenario (Fig. 1(c)), the constrained left-right proposed k-mean algorithm on Gaussian distribution has better performance than both other algorithms.

The proposed method hence improves the estimation of the set of parameters used to initiate the Baum Welch algorithm: a better and a faster convergence is shown by simulations.

### 4.3 Recognition problem

In a second set of numerical experiments (Tab. 2) classification accuracy (CA) of the proposed method is compared to CA achieved by the reference 'dynamic time warping' method (DTW) [10]. Two configurations are compared, and for each of them 50 bi-dimensional observations ($Y_{k,i} \in \mathbb{R}^2$) are generated from 25 different models. Each model is composed from the concatenation of 2 to 4 states (Configuration #1) and of 4 to 8 states (Configuration #2). In all this experiment (configurations #1 and #2), a 5-state left-right HMM with the proposed initialization procedure is considered (referred as HMM). As one can see, the proposed method outperforms the classical DTW by about 10% of classification accuracy computed in a 10-fold cross validation procedure: i.e. the 50 observations sequences are partitioned into 10 sets and each of them is sequentially used as the test database while the other 9 sets are used to train the HMM parameters. It is worth noting that the good behavior of proposed method is achieved without optimization of the number of states. As a consequence it is not surprising that the best CA is obtained on the simplest configuration (Conf. #1), however the gap between the two detection method remains of about 10% even with the more complex configuration (Conf. #2).

## 5. CONCLUSION

In this study, an initialization of the Baum-Welch training algorithm based on a modified k-mean clustering method of HMM states is presented. The proposed procedure differs from classical implementations by a clustering of the states rather than on the training data. The simulations results have shown that the proposed method improves the initialization of the Baum-Welch algorithm since the value of the log-likelihood achieved by our method is higher than value achieved by the classical initialization. Moreover, in a recognition problem, the proposed method outperforms the reference dynamic time warping method showing its good behavior.

Future works will include a deeper analysis of this method as well as an automatic procedure to adjust jointly the number of hidden states.

## REFERENCES


[1] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, February 1989.

[2] Jianying Hu, M.K. Brown, and W. Turin, "HMM based online handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1039–1045, October 1996.

[3] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, "A hidden markov model-based continuous gesture recognition system for hand motion trajectory," in *Proc. 19th International Conference on Pattern Recognition (ICPR)*, December 2008, pp. 1–4.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-likelihood from incomplete data via the EM algorithm," *J. Royal Statist. Soc. Ser. B.*, vol. 39, pp. 1–38, 1977.

[5] Olivier Cappé, Eric Moulines, and Tobias Rydén, Eds., *Inference in Hidden Markov Model*, Springer Series in Statistics, 2005.

[6] L. R. Rabiner, B. H. Juang, S. E. Levinson, and M. M. Sondhi, "Some properties of continuous hidden markov model representations," *AT&T technical journal*, 1985.

[7] K. Nathan, A. Senior, and J. Subrahmonia, "Initialization of hidden markov models for unconstrained on-line handwriting recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, May 1996, vol. 6, pp. 3502–3505.

[8] Md. Huda, Ranadhir Ghosh, and John Yearwood, "A variable initialization approach to the EM algorithm for better estimation of the parameters of hidden markov model based acoustic modeling of speech signals," in *Advances in Data Mining*, Lecture Notes in Computer Science. 2006.

[9] S. Huda, J. Yearwood, and R. Togneri, "A constraint-based evolutionary learning approach to the expectation maximization for optimal estimation of the hidden markov model for speech signal modeling," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 1, pp. 182–197, February 2009.

[10] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.