Gossip-based computation of a Gaussian mixture model for distributed multimedia indexing

Afshin Nikseresht, Marc Gelgon

▶ To cite this version:

Afshin Nikseresht, Marc Gelgon. Gossip-based computation of a Gaussian mixture model for distributed multimedia indexing. IEEE Transactions on Multimedia, Institute of Electrical and Electronics Engineers, 2008, 10 (3), pp.385-392. <inria-00368854>

HAL Id: inria-00368854

https://hal.inria.fr/inria-00368854

Submitted on 17 Mar 2009

Gossip-based computation of a Gaussian mixture model for distributed multimedia indexing

Afshin Nikseresht and Marc Gelgon

Abstract— **The present paper deals with pattern recognition in a distributed computing context of the peer-to-peer type, that should be more and more interesting for multimedia data indexing and retrieval. Our goal is estimating of class-conditional probability densities, that take the form of Gaussian mixture models (GMM). Originally, we propagate GMMs in a decentralized fashion (gossip) in a network, and aggregate GMMs from various sources, through a technique that only involves little computation and that makes parcimonious usage of the network resource, as model parameters rather than data are transmitted. The aggregation is based on iterative optimization of an approximation of a KL divergence allowing closed-form computation between mixture models. Experimental results demonstrate the scheme to the case of speaker recognition.**

I. Introduction

The technical issue addressed by this paper is the distributed computation of a probability density, while the applicative motivation is multimedia document indexing, in the particular context of a decentralised, distributed system. In this section, we first argue for a vision, towards which the scheme we disclose afterwards only supplies a small brick, but we believe this foreword to be both stimulating and necessary to justify the applicative relevance and some technical aspects of the proposal.

A central and classical need expressed by content-based indexing of multimedia documents is the assignment of a symbolic class label to a document or portion thereof, such as identifying a face, a speaker or a spatio-temporal texture or event [9]. Supervised learning is the general task for inducing the class of unlabeled data from labeled examples. Building a search engine able to recognize very many kinds of such audiovisual classes is a formidable task, long rated as unrealistic by the computer vision community, that is currently reviving as one of the most stimulating visions for both research and applications in the field [18], the other major work direction being ability to find different very different views of the same physical scene [3]. Characterizing classes involves a careful design of media-specific observations from the raw data, but by and large, there should all the more features as there are more classes to be distinguished, which in turn increases the amount for training data needed and the computation power required. Briefly stated, the work direction is promising but very demanding.

There are, however, encouraging trends towards the perspective of automatic large-scale harvesting of training examples : (i) joint text/image analysis, which may be fed by a massive resource of web pages, (ii) recent advances in weakly supervised learning [20], [23], which enables learning a class from instances supplied in clutter (e.g. a face within a complex background), and semi-supervised learning [7], which can handle jointly class-labelled and unlabelled data in the training phase. This suggests that the (necessary huge) amount of training data would inherently be distributed on a large scale and provided by independent sources that may join or leave the network. Both for alleviating the computational cost of learning and for reducing the amount of data on the network, we examine the case where supervised learning itself is distributed and, more precisely, decentralized. A suitable organization for the above vision is a peer-to-peer architecture [15] which nodes would run a service providing supervised learning of a multimedia class and would possibly store some training data. Upon request, it could classify incoming data to the best of its current knowledge. A peer-to-peer organization of participant nodes seems relevant, since (i) resources are dynamic : data and learning/classification services can join or leave the network at any time; (ii) a node is both client and server : nodes can learn from one another; (iii) resources are aggregated : the quality of the global service is due to its collective aspect; (iv) the system is decentralized : each contributor can supply data or learning tools, without any central administration. Similar ideas are also being examined for collective learning from text data [21] and sensor networks [17].

As this is a broad perspective, we now restrict the paper to decentralized supervised learning of a class. We do not address herein important issues such as service and data localization, elaborate data placement schemes (examined in [16] for retrieval of similar images), the fact that class identifiers should conform to a standard, nor the query phase.

Let us consider *statistical supervised learning of a class*. To allow for a flexible evolution of the set of classes, we favour a generative approach, that characterizes the class in feature space, over a discriminant approach, that learns directly to distinguish it from other classes. This generative approach leads to more tractable solutions, as introduction of new classes into the system does not require any update to description

The authors are with Laboratoire d'Informatique Nantes-Atlantique (LINA FRE CNRS 2729) and INRIA Atlas project-team, Polytech'Nantes/Université de Nantes, France. Email : firstname.lastname@univ-nantes.fr

This work was for part funded by Region Pays-de-la-Loire (MILES project), and SFERE.

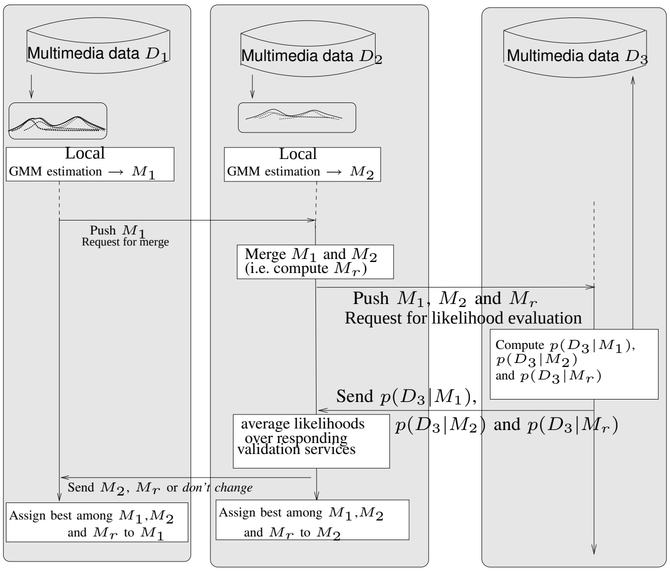


Fig. 1. This figure illustrates a single gossip cycle. Let us consider 3 nodes (the three columns of the figure) that all aim at estimating the class-conditional density of the same class. Each owns some training data, on which it computes it local estimate, of which only M_1 and M_2 are relevant in this figure. M_1 and M_2 are merged into M_r, the quality of which of evaluated over the data D_3 of the third node. The best performing model is then assigned to nodes 1 and 2.

of known classes. Consequently, the remainder of this paper will describe the technique for a single class. Practically, *we estimate the class-conditional probability density*. In this paper, the feature space is assumed common to all nodes in the network. While this leaves space for extending the work, this assumption does not contradict the ideas of the proposal and is applicable to the speaker recognition task on which we apply it.

We further focus on the *case where all densities are Gaussian mixture models*. This model form is of ubiquitous use in modelling multimedia data, for it has numerous good properties (density modelling accuracy, good behaviour in high dimension space, clean procedures for estimation and model complexity determination). They have widely been used to model audio classes [19], images [11] or motion-based spatio-temporal events in videos [9].

The mechanism we employ to propagate mixtures between cooperating nodes that participate in the scheme is *gossip*. Algorithm 1 defines its simplest version for our problem. Gossiping, here, is a non-ending background process in which acquainted nodes may share their models. Any node may then supply, at any time, an estimate of the model ; this estimate improves over time, thanks to the mechanisms proposed below.

Fig. 1 shows the procedure for a single gossip cycle. In this work, the distribution of computation and data is due to the applicative context, in which independent systems cooperate. The key goal of the system is to obtain an estimate which quality is close to what would have been estimated in a centralized version.

Despite its simplicity, this asynchronous, decentralized technique is very effective. Its good properties are extensively reviewed in [8], but may summed up for our problem as follows :

- *speed up* by implementing coarse-grain parallelism over the set of nodes. This occurs at two levels : (1) gossip-based parallelism of learning by merging (step 2 of Algorithm 1) and (2), for each step of (1), parallelism in the computation of the likelihoods for validation (step 3 of Algorithm 1);
- *robustness* both in the distributing computing and statistical estimation senses, since :
  - any node may *leave* during the gossiping without causing major degradation or *join* and obtain, with high probability, an effective estimate of what has been previously collectively estimated on the network,
  - a very poor estimate in a minority of nodes does not affect the collectively estimated model.

Efficiency of the proposed technique comes the two main following features:

- merging density estimates between nodes only involves transmission of, and computation on, mixture model parameters, rather than the generally large amount of multimedia data (or feature vectors that represent it). As a result:
  - the amount of information to be sent on the network is very low ;
  - computation on nodes remains low, relatively to estimation tasks that operate on the multimedia data or feature vectors,
- during the gossip-based model learning phase, the complexity of any mixture (i.e. the number of Gaussian components) keeps a constant order of magnitude. Let us underline that the distributed learning phase and the querying phase, can fully overlap, since mixture reduction keeps the class representation directly ready for query evaluation.

The key mechanism that enables these properties is a criteria and an algorithm related to merging two (or more) mixture models, which are exposed further down.

A work which goal is close to ours, i.e. gossip-based distributed estimation of the parameters of a Gaussian mixture, has been recently presented in [13]. Their approach consists in introducing parallelism in the EM algorithm, by gossiping the M-step, resorting to original data. In our case, in contrast, each contributing node is in charge of estimating its local Gaussian mixture model, and is free to use any mixture model parameter estimation technique for this. The latter point gives an interesting degree of freedom towards a completely decentralized system : only the mixture description need to be standardized, while the node may benefit from recent advances in mixture estimation techniques (e.g. variational Bayes [2], or versions suitable for large amounts of data [22]). Further, the averaging in [13] between the parameters to be merged is simply uniform. To our understanding, a more central difference is that their way of merging knowledge between mixture models does not (at least explicitely) address

Algorithm 1 A gossip cycle for merging-sharing Gaussian mixture models

1. Select at random two nodes in the network, which models are M_1 et M_2 (practically, nodes should autonomously select their partners in a dialogue)
2. Concatenate the components of M_1 and of M_2 to form a single model M_c, then reduce M_c to a merged model M_r with lower number of Gaussian components.
3. Evaluate which among M_1, M_2 and M_r better describes third party data from the data class. A key point is that generally, M_r will perform best.
4. Assign this best model to M_1 and to M_2

correspondence between components to be merged, and leaves open the issue of merging models with different number of components. More generally, we shall see that our technique is amenable to variation of the number of components in the mixture along the gossiping process.

In the remainder of this paper, we detail the proposed approach for distributed model learning (section 2) and demonstrate it on the example of a speaker recognition task (section 3). Section 4 provides concluding remarks.

II. Parameter-level merging of Gaussian mixture models

This section justifies and details how mixture models may be merged using parameter-level rather than data-level computations : section 2.1 defines the optimality criterion aimed at of the merged model, while section 2.2 discusses an approach for conducting the corresponding optimization.

A. Optimality criterion

Let two nodes each carry different probabilistic Gaussian mixture models, denoted $M_1(x)$ and $M_2(x)$, associated to the same multimedia class and hence hidden density $p(x)$. The mixtures can be expressed as :

$$M_k(x) = \sum_{i=1}^{m_k} w_k^i N_k^i(x), \qquad k = 1, 2 \tag{1}$$

where $N_k^i(x)$ is a Gaussian component which mean is μ_k^i and covariance Σ_k^i and the w_k^i are scalar weights. Model M_k is estimated on a data set of size n_k located on node k. $p(x)$ can be estimated by concatenating incoming mixtures as follows :

$$M_c(x) = \frac{1}{n_1 + n_2}(n_1 \sum_{i=1}^{m_1} w_1^i N_1^i(x) + n_2 \sum_{i=1}^{m_2} w_2^i N_2^i(x)) \tag{2}$$

However, the $m_1 + m_2$ components in M_c are generally largely redundant, which implies a useless increase in evaluation cost of likelihoods for this density at query time, when merges are chained gy gossip. Consequently, scaling up the scheme requires transforming M_c into a reduced mixture $M_r = \sum_{i=1}^{m_r} w_r^i N_r^i(x)$ that preserves reasonnably well the density while only having the necessary number of components for this. The point of this policy is that the order of magnitude of the number of components is kept constant through propagation, although it may fluctuate to fit the complexity of the density.

The class models in the nodes would be used to classify new data, typically based on maximum likelihood or more elaborate criteria involving the likelihood. In order to preserve the likelihood as much as possible, we seek a mixture model M_r which maximizes the expected log-likelihood of data D assumed to be drawn from M_c, see (3). It is classically established [5] that this amounts to minimizing the Kullback-Leibler divergence $KL(M_c \| M_r)$, defined by (5), which, in short, measures the loss of information due to the approximation of M_c by M_r :

$$\hat{M_r} = \arg\max \quad E_{M_c} \; [\; ln \;\; p(D|M_r) \;] \tag{3}$$

$$\hat{M_r} = \arg\min \quad \left[\; - \int M_c(x) \; ln \; M_r(x) \; dx \; \right] \tag{4}$$

$$\hat{M_r} = \arg\min \quad \left[- \int M_c(x) \; ln \; \frac{M_r(x)}{M_c(x)} \; dx \; \right] \tag{5}$$

A major issue for the practical computation of (5) is the lack of closed form for this divergence, in the case of Gaussian mixtures, but we propose a bypass in the form of the following approximation. Linearity of the integral applied to (4) provides :

$$\hat{M_r} = \arg\min \quad \left[- \sum_i w_c^i \int N_c^i(x) \; ln \; M_r(x) \; dx \right] \tag{6}$$

In each term of the sum in (6), we approximate the mixture M_r by only one of its Gaussian components, selected as the best approximation to N_c^i, in the KL sense. This leads to the following similarity measure :

$$d(M_c, M_r) = \sum_{i=1}^{m_1+m_2} w_c^i \min_{j=1}^{m_r} KL(N_c^i \| N_r^j) \tag{7}$$

This similarity measure can easily be computed at low-cost, since the Kullback divergence between two Gaussians, which parameters are (μ_1, Σ_1) and (μ_2, Σ_2), benefits from the following closed-form expression :

$$\frac{1}{2}(log\frac{|\Sigma_2|}{|\Sigma_1|} + Tr(\Sigma_2^{-1}\Sigma_1) + (\mu_1 - \mu_2)^T \Sigma_2^{-1}(\mu_1 - \mu_2) - \delta) \tag{8}$$

where δ is the dimension of the feature space.

B. Optimization : an iterative scheme and its initialization

To gain insight into complexity, we assume $m = m_1 \approx m_2 \approx m_r$. The search space is of size $O(m^2)$ and typically cannot be searched exhaustively if there are more than 10 components, which is common when modelling multimedia classes. Hence, we optimize locally criterion (7) with an iterative scheme detailed in Algorithm 2 below, which is adapted (by several aspects) from a technique [10] proposed in the context of hierarchical clustering. The procedure bears analogy with the classical k-means algorithm, in that it operates local optimization by alternatively assigning elements to groups and re-computing group representatives. In our context, the elements are the components of M_c and the representatives those of M_r.

As often done with k-means, the initial assignements π^0 from which local optimization proceeds could be drawn randomly. Our context suggests a more effective initialization criteria in our context : since generally, Gaussian components coming from the same mixture are not redundant, we draw π^0 at random with the constraint that components arising from the same mixture are not initially grouped. The iterative scheme may still regroup them later, if the data drives it that way. As we draw multiple starting points to retain the best local optimum, this strategy improves sampling of the search space.

C. Complexity of the reduced model

An important point in the proposed approach is the determination of the number of Gaussian components in the reduced model M_r. The seminal study reported in [1] showed that estimating the Kullback divergence is in fact affected by a bias that grows with the number of parameters to be estimated, i.e. with the number of components. It also supplies a first-order approximation of this correction, which we apply here to the definition of $d(M_c, M_r)$, which hence becomes :

$$d(M_c, M_r) = \sum_{i=1}^{m_1+m_2} w_c^i \min_{j=1}^{m_r} KL(N_c^i \| N_r^j) \quad + \quad \nu_{M_r} \quad (13)$$

where ν_{M_r} is the number of independent parameters in the mixture. Our experimental results back the application of this approximation : the number of components obtained in practice appears very similar to that obtained by usual (AIC,BIC) model selection criteria on the model computed directly on all the data (i.e. discarding the distributed aspect of the learning process). We evaluate exhaustively from 1 to m_1+m_2 the performance of each possible number of components in M_r, in independent trials. A faster alternative would be to compute this recursively downwards from $m_1 + m_2$ to 1, but experimental results suggest this can excessily prune the search space at early stages.

D. Validation of a merge operation

In this section, we discuss the need to validate the ability of M_r to generalize to the complete data over the network. Generally speaking, estimation of statistics by gossiping may be shown to converge in some cases (e.g. computing a means and quantiles [12]) but in our problem, the lack of a global view on the data occasionnally leads to a situation where $\hat{M}_r$ is a better model than M_1 and M_2 for local data $D_1 \cup D_2$, but worse on the complete data over the network.

We thus introduce in the scheme a step to validate M_r on third-party data. As described in algorithm 3, it consists in sending $\hat{M}_r, M_1$ and M_2 to a sufficient number of randomly selected acquaintance nodes, each of which make these models compete on its local data and returns the corresponding three likelihoods. When the requesting node has received a sufficient number of such responses (4 in our experiments, further work could make this adaptive), it takes the decision to validate M_r or reverse to M_1 and M_2.

This phase loads the network with more messages, but (i) these messages are very short and no multimedia data nor feature vectors are transmitted (ii) computation of likelihoods is inexpensive.

While exchanges between nodes in this validation step may be implemented in a variety a ways, limited depth network flooding is an interesting one, offering the following perspective of extension on the present work : likelihood information being collected and aggregated through the flooding may be useful not only to the node emitting the request, but also to other nodes, since at no extra cost, they can learn from it about the quality of their own model, relatively to others. In other words, a ranking of the nodes may be learned in a decentralized way, which could help route queries to more effective models.

III. Experimental results

The example of distributed speaker recognition is taken throughout this section, as it is a representative case where Gaussian mixtures are very popular. The technique however directly applies to a wide range of audiovisual classes. We first focus on the merging operation, i.e. at local scale (section III-A). We then observe global performance, in the context of gossip-based mixture propagation (section III-B). Throughout these experimental results, the figure of merit is the quality of the class-conditional pdf estimate, in particular with respect to a conventional, centralized approach, rather than ability of the scheme to classify new data correctly. The latter however derives direcly from the former in a Bayesian decision rule.

A. Detailled view on one or two merge operations

In the first experiment, each of three nodes has learnt a class-conditional density for a speaker in a common 13-dimension mel-cepstral feature space [19]. The three corresponding mixtures merge simultaneously into a single mixture (straightforward since (1) generalizes to merging more than two mixtures). Each node was provided with different training data from the same speaker and the duration of audio recordings was between 7 to 16 seconds, depending on node. Each node provides a mixture estimate from local data, and is free to choose the precise technique used for this. For our experiments, the Expectation-Maximization local optimization algorithm is employed, with some enhancements [4] to limit poor local minima. Each mixture also autonomously and automatically determines its number of components (in practice, the common BIC criterion was used). All covariance matrices in the mixture are full (rather than spherical or diagonal).

The three incoming nodes respectively have 4,4 and 5 components. Their concatenation into M_c supplies a 13-component model, which should be reduced to a lower number of components, to be determined. Fig. 2(a) displays, in the example case of the second feature vector, the three incoming densities, the concatenated density and the density after mixture reduction.

Algorithm 2 Iterative optimization algorithm for estimating the reduced model M_r (criterion (7))

```
for $m_r$ : from 1 to m1+m2 do
  Start from a constrained random initialization $\hat{\pi}^0$ (or given, if available)
  $it = 0$
  repeat
    1. Re-fit mixture $M_r$ :
    given the current component clustering $\hat{\pi}^{it}$, set initially or computed at the previous iteration, update mixture model parameters as follows :
        $\hat{M_r}^{it} = \arg\min_{M_r \in \mathcal{M}_{m_r}} d(M_c, M_r, \hat{\pi}^{it})$   (9)
    where $\mathcal{M}_{m_r}$ is the space of all mixture with $m_r$ components that may be formed by grouping components of $M_c$. This re-estimation in fact amounts to updating each component of $M_r$ as follows. For component $j$, algebra leads to the following expressions :
        $\hat{w}_r^j = \sum_{i \in \pi^{-1}(j)} w_c^i, \quad \hat{\mu}_r^j = \frac{\sum_{i \in \pi^{-1}(j)} w_c^i \mu_c^i}{\hat{w}_r^j}, \quad \hat{\Sigma}_r^j = \frac{\sum_{i \in \pi^{-1}(j)} w_c^i (\Sigma_c^i + (\mu_c^i - \hat{\mu}_r^j)(\mu_c^i - \hat{\mu}_r^j)^T)}{\hat{w}_r^j}$   (10)
    where $\pi^{-1}(j)$ is a light notation for $\hat{\pi}^{-1,it}(j)$, the set of $M_c$ that project onto component $j$ in $M_r$. Let us note that $\hat{\Sigma}_r^j$ is generally non-diagonal, even if the components being grouped have diagonal covariance matrices, such as is often the case with decorrelated features used in e.g. speech or speaker recognition.
    2. Grouping components :
    for mixture $\hat{M}_r^{it}$ obtained in Step 1, we seek the mapping $\pi^{it+1}$, defined from $\{1, \ldots, m_1 + m_2\}$ into $\{1, \ldots, m_r\}$, which best groups components of $M_c$ to build components of $\hat{M}_r^{it}$, in the following sense :
        $\hat{\pi}^{it+1} = \arg\min_{\pi} d(M_c, \hat{M}_r, \pi)$   (11)
    In other words, each component $i$ of $M_c$ projects onto the closest component $j$ of $\hat{M}_r^{it}$, according to their Kullback divergence ((12) below). In this phase, we resort to exhaustive search among 'source' components, which has a low-cost, thanks to the availability of (8).
        $\pi^{it+1}(i) = \arg\min_j KL(N_c^i || N_r^j)$   (12)
    3.it=it+1
  until convergence (i.e. $\pi^{it+1} = \pi^{it}$)
  compute $d(M_c, \hat{M}_r) = \sum_{i=1}^{m_1+m_2} w_i \min_{j=1}^{m_r} KL(N_c^i \| N_r^j) \quad + \quad \nu_{M_r}$
end for
Retain model $\hat{M}_r$ which minimizes $d(M_c, \hat{M}_r)$ over the set of candidate mixture complexities explored.
```

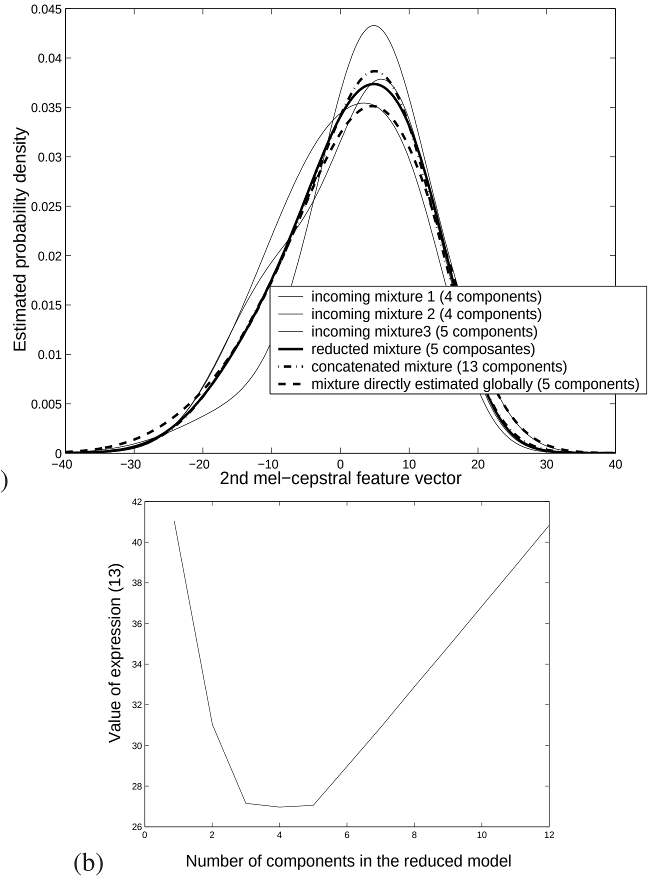
Algorithm 3 Validation of a merge

```
for enough times do
  Draw node $k$ at random among acquaintance nodes
  Sends $\hat{M}_r, M_1, M_2$ to node $k$ running a GMM evaluation service
  Node $k$ computes $p(D_k|\hat{M}_r), p(D_k|M_1), p(D_k|M_2)$ and sends them back to current node.
end for
if $p(D_k|\hat{M}_r) > p(D_k|M_1)$ and $p(D_k|\hat{M}_r) > p(D_k|M_2)$ then
  Validate the merge operation (proceed as in Algorithm 1)
else
  ignore it and keep $M_1$ and $M_2$
end if
```

The mixture estimated (again enhanced EM) on the whole the data over the network is also plotted. While the main point of this paper is to propose a decentralized alternative to this, this direct model (denoted M_d) serves herein as a reference density against which we evaluate the loss due to distribution of the data and computations. Fig. 2(b) shows that criterion (13) chooses a reduction from 13 to 4 components. To evaluate the effectiveness of the mixture reduction, Fig. 2(c) provides numerical evidence in terms of Kullback-Leibler loss between reference mixtures (M_d and M_c) and approximating mixtures. KL divergence is used (rather than its approximation proposed in (7)). We evaluate it by a Monte-Carlo procedure with N=10^8 samples, as follows :

$$KL(p, \tilde{p}) \;\approx \frac{1}{N}\sum_{i=1}^{N} log\frac{p(x)}{\tilde{p}(x)} \tag{14}$$

where p and $\tilde{p}$ respectively denote an ideal model and its approximation. While this should be closer to the true loss than (7)), its computational cost forbids its usage in the scheme, it is only used here for external assessement.

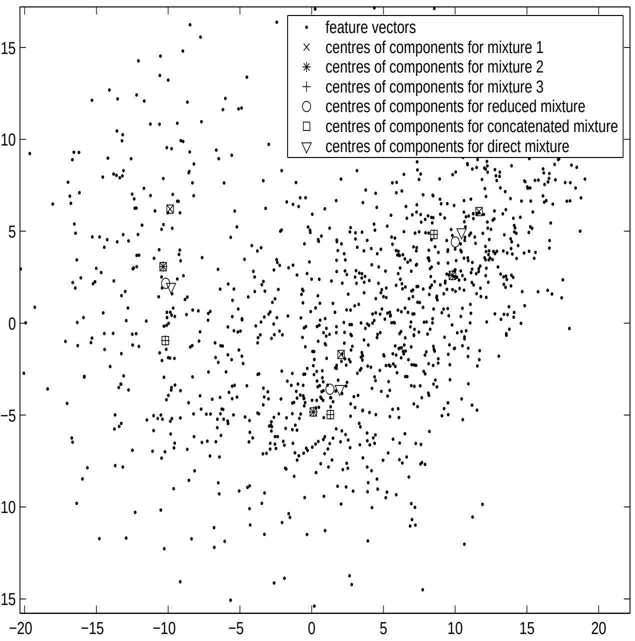


(a)

(b)

(c)

	KL(concatenated ‖...)	KL(direct‖...)
mixture 1	0.0241	0.0306
mixture 2	0.0030	0.0219
mixture 3	0.0101	0.0309
reduced mixture	0.0005	0.0087



(a)

(b)

	KL(concatenated ‖...)	KL(direct‖...)
mixture 1	0.125	0.112
mixture 2	0.109	0.117
mixture 3	0.075	0.082
reduced model	0.015	0.020

Fig. 3. Three mixtures merge (not simulatenously, see main text). (a) shows the feature vectors and the centres of the Gaussian components (for incoming, concatenated, reduced models, as well as, for reference, the mixture that could be directly estimated over the whole data set). (b) Evaluation of the KL loss between reference densities (concatenated, direct) and models coming in and out of the merge operation.

It can be observed that the direct mixture is much better approximated by the reduced mixture than by any of the incoming mixtures. This does not come at the expense of mixture complexity, since the reduced mixture has 4 components, in fact the same is estimated by a BIC criterion for the direct model.

We report a second experiment, applied to a different speaker. It again involves three nodes but, in contrast to the previous experiment, two nodes are merged, and then a third node is merged to their reduced mixture to form a final reduced mixture. The experiment is conducted in a 2-dimension space, for the sake of clarity of fig. III-A(a). Its purpose is more an illustration value than a demonstration of large scale effectiveness. The centres of the incoming mixtures, as well as the centres of the reduced and direct mixtures are superimposed to the feature vectors, and the two latter are clearly very close.

B. Gossip-based estimation

This section reports the performance of the proposed technique in the gossip setting. Each node owns different data from the same speaker and independently estimates its own model. Practically, EM with multiple starts is employed in our experiments for this purpose but, as stated before, other techniques may be used.

We evaluate the capability of each mixture on the network to model data D from the class of interest (here, a speaker) with the classical marginal likelihood [14]. Data D here is the union of the data dispatched over all nodes, which is never gathered when the practical system runs, but is a relevant figure of merit for external observation.

We carry out the practical computation of the marginal likelihood of the data with the BIC criterion :

$$BIC(D|M) = -log\ p(D|\hat{\theta}) + \frac{\nu\ log(\sharp D)}{2} \qquad (15)$$

where mixture M is defined by a parameter vector θ, $p(D|\theta)$ is the likehood of the data for this model, ν is the number of independent parameters in the mixture and $\sharp D$ the size of the data set (the data set does not need to propagate in the network, but its size should propagate and cumulate in n_1 and n_2)

Fig. 4 depicts, after each gossip cycle and on each node, the evolution of criterion (15), which should be minimized. The following observations can be made. The process stabilizes around a "collective model". Convergence cannot be established, as illustrated in the zoom into Fig. 4, due to the lack of an optimization criterion global to the network, which is the case in the prototypal example of computation of a mean [13], [6]. From a practical viewpoint, however, *all nodes are rapidely assigned a mixture that is better (slightly or largely) than any of the original mixture, which later implies improvement in recognition rates when the system is queried.* In this experiment, the effectiveness of the collective model is significantly better than that of a single mixture model that could have been estimated directly on the whole data (the performance of which is represented by a dashed horizonal line). This latter advantage however reduces when the size of the feature space is large compared to the amount of training data (dimensionality curse). Overall, however, this example, which is representative of many other obtained, suggests that the proposed scheme provides promising results on three points : quality in model estimation, the flexibility of a decentralized system, and speed up thanks to parallel computing.

It should also be underlined that the horizontal axis only indicates time order and is non-linearly related to time, since gossiping is strongly parallel.

As the result depends on the order in which the nodes are merged and on the random initializations involved in the merging algorithm, we draw in fig. 5 statistics to average out those effects. Fig. 5 indicates how variable the likehood is over the set of nodes (this variability is averaged over 50 independent runs of the complete gossip).Variability decreases very fast ; the transient phase with higher variability (up to the 18th gossip cycle) corresponds to not all the nodes having yet participated in the gossip process.

As illustrated in fig. 6, the scheme can easily handle a node that joins the network. In this example involving 20 nodes, an additional node joins after 50 cycles. Soon after it joins, it benefits from the previous exchanges. Indeed, the amount of data available for mixture estimation ((n_1 and n_2) in eq. (2)) cumulates as gossip progresses.

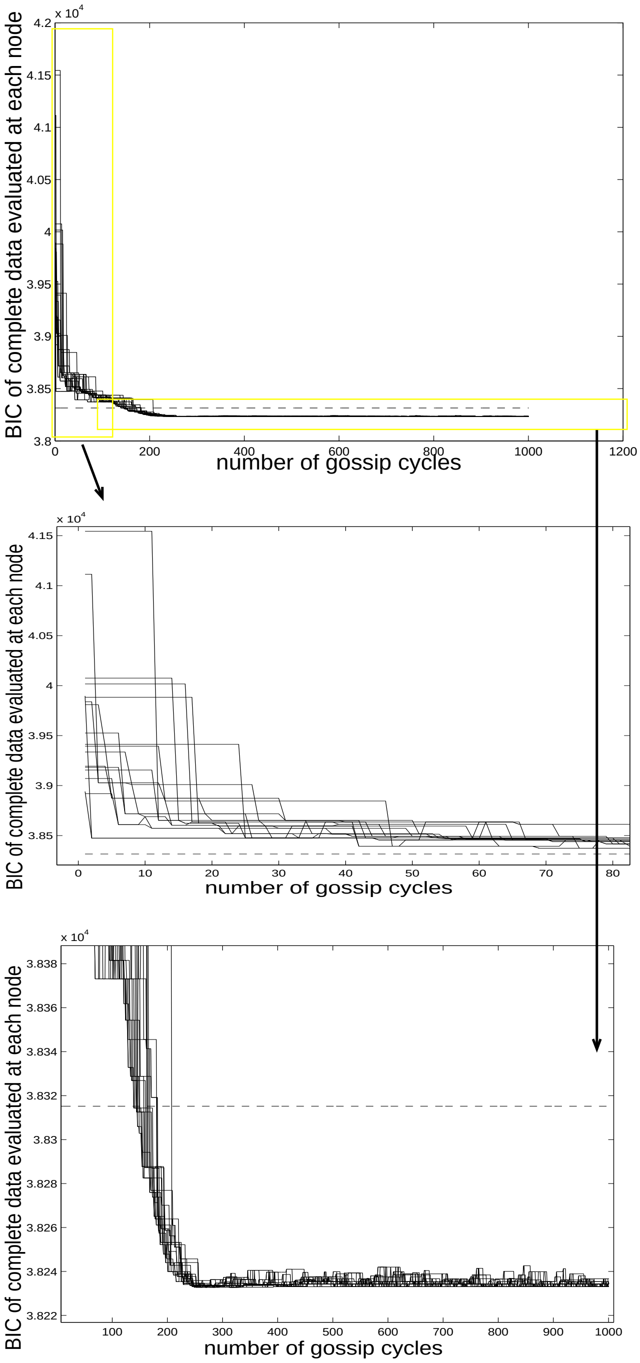


Fig. 4. All graphs measure the BIC criterion over time (this criterion evaluates the ability of the Gaussian mixture models to generalize to all data from the class being learnt; it should be minimized). Top : this evolution is shown for the 20 nodes participating in the experiment. As the 20 graphs are somewhat superimposed, Medium and Bottom figures are zooms on the top figure, for clarity sake.

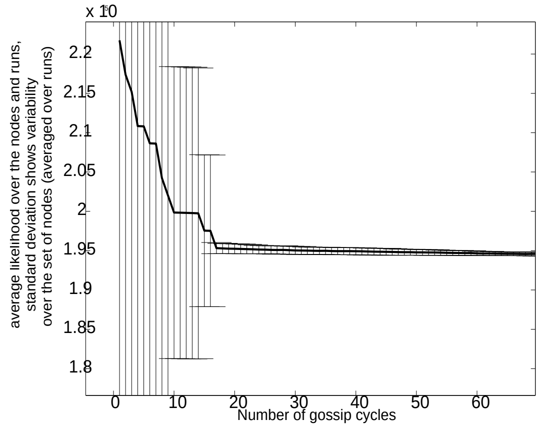


Fig. 5. This graph shows the statistical behaviour of the system : the variability of the likelihood over the set of nodes participating in the gossip. This variability is averaged over 50 runs

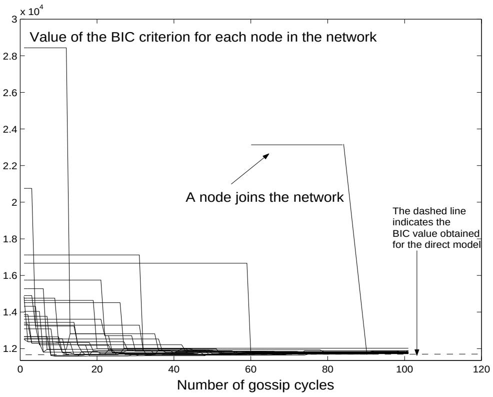


Fig. 6. This experiment illustrates fast integration of a node joining a distributed learning process involving 20 nodes. Right from its first contact (cycle 73), the joining node strongly improves by catching the central trend of the network.

IV. Conclusions

This work fits into a vision towards a multimedia indexing and retrieval system, which would be decentralized and deployed on a large scale. In this setting, algorithmic components are required, that induce low computational cost, incrementality and only require a little amount of bits to transit between nodes.

This paper proposed a novel scheme for this purpose, dedicated to Gaussian mixtures models, which are one of the most useful representations of a multimedia class. The proposal wraps a parcimonous mixture model merging technique into a gossip framework, demonstrating that it can efficiently propagate and collectively improve estimates over time. The point of the gossip framework is that it is well suited to dynamic, decentralized computing environments.

More generally, crossing pattern recognition and large-scale distributed computing is a promising direction in content-based multimedia indexing, since the first ingredient can greatly enhance services offered to users, far beyond file sharing, while the second provides data, computation and algorithmic resources.

References

[1] H. Akaike. A new look at the statistical model identification problem. *IEEE Trans. on Automatic Control*, (19):716–723, 1974.

[2] H. Attias. A variational Bayesian framework for graphical models. In *Neural Information Processing Systems (NIPS) Conference*, Denver, USA, November 1999. MIT Press.

[3] S.A. Berrani, L. Amsaleg, and P. Gros. Robust content-based image searches for copyright protection. In *Proc. of ACM workshop on Multimedia databases*, pages 70–77, New Orleans, USA, November 2003.

[4] C. Biernacki, G. Celeux, and G. Govaert. Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics and Data Analysis*, 41:561–575, 2003.

[5] C. Bishop. *Neural networks for Pattern Recognition*. Oxford University Press, 1995.

[6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. on Information theory*, June 2006.

[7] F. Cozman, M. Cirelo, T.S. Huang, I. Cohen, and N. Sebe. Semisupervised learning of classifiers : theory, algorithms and their applications to human-computer interaction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(12):1553–1567, December 2004.

[8] P.T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massoulié. From epidemics to distributed computing. *IEEE Computer*, 37(5), 2003.

[9] R. Fablet, P. Bouthemy, and P. Perez. Non parametric motion characterization using causal probabilistic models for video indexing and retrieval. *IEEE Trans. on Image Processing*, 11(4):393–407, April 2001.

[10] J. Goldberger and S. Roweis. Hierarchical clustering of a mixture model. In *Proc. of Neural Information Processing Systems (NIPS'2004)*, pages 505–512, 2004.

[11] R. Hammoud and R. Mohr. Gaussian mixture densities for video object recognition. In *Proc. on Int. Conf. on Pattern Recognition (ICPR'2000)*, pages 71–75, Barcelona, Spain, August 2000.

[12] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *IEEE symp. on foundations of computer science*, Cambridge, MA, USA, October 2003.

[13] W. Kowalczyk and N. Vlassis. Newscast EM. In MIT Press, editor, *Proc. of Neural Information Processing Systems (NIPS) 17*, 2005.

[14] D. MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[15] D. Milojicic, V. Kalogeraki, R. Lukose, L. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. Technical Report 2002-57, Hewlett-Packard, 2002.

[16] W.T. Muller, M. Eisenhardt, and A. Henrich. Efficient content-based P2P image retrieval using peer content descriptions. In *Proc. of SPIE Internet Imaging V*, volume 5304, pages 57–68, December 2003.

[17] R. Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE Trans. on Signal Processing*, 51(8), August 2003.

[18] J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, editors. *Towards category-level object recognition*. Springer, 2006.

[19] D. Reynolds. Speaker identification and verification using gaussian speaker models. *Speech communication*, 17:91–108, 1995.

[20] C. Schmid. Weakly supervised learning of visual models and its application to content-based retrieval. *Int. Journal of Computer Vision*, 1(56):7–16, 2004.

[21] C. Tang, S. Dwarkadas, and Z. Xu. On scaling latent semantic indexing on large peer-to-peer systems. In *Proc. of ACM SIGIR (SIG on Information Retrieval*, pages 145–153, Sheffield, U.K., July 2004.

[22] J.J. Verbeek, J.R.J. Nunnink, and N. Vlassis. Accelerated EM-based clustering of large datasets. *Data Mining and Knowledge Discovery*, 2006. in press.

[23] L. Xie and P. Perez. Slightly supervised learning of part-based appearance models. In *Proc. of IEEE Workshop of learning in computer vision and pattern recognition*, Washington, DC, USA., June 2004.