RC23851 (W0601-121) January 27, 2006
Computer Science

IBM Research Report

In Interleaved HMM/DTW Approach to Robust Time Series Clustering

Jianying Hu, Bonnie Ray
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Lanshan Han
Renssalaer Polytechnic Institute
110 8th Street
Troy, NY 12180

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

An Interleaved HMM/DTW Approach to Robust Time Series Clustering

Jianying Hu Bonnie Ray
IBM T.J. Watson Research Center
1101 Kitchawan Road, Route 134 Yorktown Heights, NY 10598
{jyhu, bonnier}@us.ibm.com

Lanshan Han
Rensselaer Polytechnic Institute
110 8th Street, Troy, NY 12180
hanl3@rpi.edu

Abstract

We introduce an approach for model-based sequence clustering that addresses several drawbacks of existing algorithms. The approach uses a combination of Hidden Markov Modeling (HMM) for sequence estimation and Dynamic Time Warping (DTW) for hierarchical clustering, with interlocking steps of model selection, estimation and sequence grouping. We demonstrate experimentally that the algorithm can effectively handle sequences of widely varying lengths, unbalanced cluster sizes, as well as robustness to outliers.

1 Introduction

Cluster analysis is a way to derive structure from data by automatically partitioning the data samples into homogeneous groups. In model-based clustering, mathematical models are used to represent the cluster structure, with the models for each cluster selected to optimize the data fit. Model-based clustering has been widely used in many applications, especially those involving complex data. Compared to distance-based clustering, model-based methods can provide better interpretability [19] and richer representation of the data. Some models that have been found useful for model-based clustering of data sequences include Markov and Hidden Markov Models (HMMs), among others. HMMs are an extension of Markov models in observations are probabilistic functions of the states, but for which the states cannot be observed directly. HMMs are particularly attractive for the clustering of time series, or more generally, sequence data, for two reasons. First, they represent a formal probabilistic model with solid mathematical foundations, with available efficient and well-defined algorithms for inducing HMMs from a set of sequences [14]. Second, the hidden states in HMMs provide a compact and easy-to-interpret representation of the underlying "stages" in a dynamic process. Even though the exact sequence of states behind each generative process cannot be observed, it can be estimated by studying the observable output of the system. Because of these desirable properties, HMMs have been successfully used to model a wide variety of time series arising from real-world applications including speech recognition[14, 8], protein sequencing [11], computational molecular biology [1], handwriting recognition[7] and human gesture recognition [2].

However, while well known algorithms exist to induce HMMs from a set of time series data [14], these algorithms do not directly address the problems of clustering the time series: they simply attempt to fit a single model that best accounts for all of the data, regardless of whether it was generated by one or multiple underlying processes. Clustering time series data using HMMs is a very different and much more complex task: it involves the difficult tasks of estimating the number of homogeneous clusters, or underlying processes, as well as the number of states representing the process that generated the data in each cluster.

Early work on HMM based sequence clustering focused on speech recognition [15, 10, 5] and assumed that the number of states in the models is known before clustering (e.g., pre-defined by linguistic experts). Also, clustering results in most of these systems are evaluated and selected based on the amount of improvement in recognition accuracy achieved by the models. Other methods for cluster number selection were later proposed by Smyth [18], who used Monte-Carlo cross validation, Oates *et. al.* [13], who used an initialization process based on Dynamic Time Warping and hierarchical clustering, and Schliep *et. al.* [16], who used a splitting criterion based on cluster size.

However all of these systems still assume that the number of states is known beforehand and fixed for all clusters.

More recently,Li *et. al.*[12] proposed a more general clustering methodology called *Matryoshka*, which does not assume that the number of states in the HMMs are known beforehand or fixed for all clusters. The method is a top down approach which starts by assigning all sequences to one cluster fitted by a single state model, then iteratively increase the number of states as well as the number of clusters. Both the partition size (number of clusters) and model sizes (number of states) are determined using a Bayesian Information Criteria (BIC) based measure. The BIC measure is used to ensure that the HMMs generated are accurate representations of the data, and at the same time are meaningful abstracts that are easy to interpret, e.g., not overtly complex.

While Matryoshka demonstrates a more general framework allowing the objective, data-driven determination of both model and partition sizes in HMM-based sequence clustering, there are several drawbacks in the methodology which hinder its application in many real-world situations. First. the method assumes that all sequences are of equal length. Second, to generate new clusters, the method uses a simple approach of initializing a new cluster using the sequence that is farthest from the existing models. Since the expectation-minimization (EM) style iterative procedure used to refine the models at each stage is highly sensitive to the initial condition, this simple approach makes the method unstable when the data contains unbalanced clusters (clusters of very different sizes), or outliers (singletons or very small clusters). Finally, the method provides no mechanism to handle outliers or noise in the data: it attempts to account for all the data with HMMs. Because of the latter two factors, Matryoshka tends to get "distracted" in the presence of highly unbalanced clusters or outliers, resulting in sub-optimal models.

In this paper we present a new algorithm for HMM-based sequence clustering designed to address these problems. While we also adopt a top-down iterative approach, our algorithm differs from Matryoshka in three important aspects. First, a normalized BIC measure is adopted to allow for sequences of varying lengths. Second, a mechanism called the *outlier pool* is introduced to dynamically identify and handle outliers throughout the clustering process. Finally, we provide a more sophisticated methodology for creating and initializing new clusters. Whenever the partition size is to be increased, the candidate cluster for splitting is identified based on a goodness-of-fit evaluation for all existing models. Then, Dynamic Time Warping (DTW) combined with hierarchical clustering is used to initialize the two new clusters.

DTW is a dynamic programming algorithm designed to minimize the dissimilarity/distance between any two given sequences, even those of very different lengths or stage progressions. It efficiently searches for the optimal mapping between the points in the two sequences that minimizes the accumulated point-wise distance and then returns this distance as the dissimilarity measure between the two sequences. In our algorithm, DTW is used to provide initial splits generating new clusters. HMM estimation is then used for model-based refinement of clusters, allowing for a pool of potential outliers that are not explicitly modeled. Experimental results demonstrate that this new methodology, combined with the outlier pool, effectively improves the robustness of the clustering results.

DTW was also used by Oates *et. al.* in sequence-based clustering [13]. However, in their system, DTW was applied once to identify all clusters, which were then adjusted and refined using HMMs with known number of states. In contrast, our method interleaves DTW into every step of a top-town, model-based clustering scheme that searches for the optimal number of clusters and optimal number of states for each cluster in an iterative manner guided by a normalized BIC measure.

The rest of this paper is organized as follows. In Section 2, we present details of our algorithm. Experimental results are presented in Section 3. Section 4 concludes.

2 The Interleaved HMM/DTW Algorithm

Suppose we have a set of N sequences (samples) of varying lengths: $X = (x_1, \cdots, x_N)$. We assume that a majority of the sequences were generated by an unknown number of HMMs, each representing a "dominant" underlying regime in the data. By "dominant" we mean it represents a significant number of sequences. However, the number of sequences represented by each dominant regime may vary widely (*i.e.*, the corresponding clusters maybe highly unbalanced). We further assume that the data may contain outliers, or sequences that do not belong to any of the dominant regimes. The goal of the clustering algorithm is to identify the clusters that correspond to these dominant regimes, along with the underlying models that characterize the sequences in each regime.

This clustering problem can be viewed as a model-fitting problem, where, given a set of data assumed to come from a mixture of models, we attempt to find the best estimate of the model parameters such that they lead to maximum likelihood of the data. The challenge is how to solve the nested problems of identifying the "right" number of clusters, and given a cluster, the "right" model size for the cluster. We adopt a top-down approach, where we start with the minimal size for both model and partition, and increment them in an estimation-maximization (EM)-like procedure until a certain "goodness" measure is reached. Here, we use a Bayesian Information Criterion [17] based measure

Table 1. Outline of the clustering algorithm

Assign all sequences to one cluster.
Apply **Model Construction** to the cluster.
Sample Reassignment/Outlier Detection.
Compute **normalized** partition BIC measure.
while BIC measure for current partition $>$ BIC measure of previous partition:
Partition Growing.
Apply **Model Construction** to each new cluster.
Sample Reassignment/Outlier Detection.
Compute **normalized** BIC for current partition.
end while
Accept the previous partition as the final partition.

Table 2. The Partition Growing Module

Identify the cluster with smallest normalized likelihood.
Compute DTW-based distance matrix, D.
Split the cluster based on D.

Table 3. The Sample Reassignment/Outlier Detection Module

repeat
Compute the acceptance threshold for each model using Monte Carlo simulation
For each sequence x_i, identify model j having maximum likelihood;
If likelihood $>$ acceptance threshold then assign x_i to cluster j
Otherwise assign x_i to outlier pool.
Apply **Model Construction** to each cluster with changed membership
until no more change of cluster membership

to assess model goodness, which provides a nice trade-off between model parsimony and maximization of the likelihood.

Tables 1–3 give a high-level outline of our clustering algorithm. In the following sections we explain in detail the three key components of this algorithm: model and partition size selection, partition growing, and outlier handling.

2.1 Normalized BIC for state and partition size selection

The Bayesian Information Criterion (BIC) was first proposed as a criterion for model selection when fitting a mixture model in a Bayesian framework. It was derived from an asymptotic approximation formula proposed by Schwarz in 1978 [17]. The basic definition of the BIC measure given a mixture model M and data set X is:

$$BIC(M, X) = \log\{P(X|M, \hat{\theta})\} - \frac{d}{2}\log(N), \quad (1)$$

where $\hat{\theta}$ denotes the maximum likelihood estimate of the model parameters, d is the number of free parameters in the model, and N is the number of data samples in X. The first term in the formula is the likelihood term, which tends to favor larger and more detailed models, while the second term is the model complexity penalty term, which favors simpler models. Thus BIC has the effect of selecting a good, yet parsimonious model for the data by trading off the contributions of these two terms. terms. Over the years, various forms of the BIC measure have been used successfully in many different clustering applications [6, 3, 4], however most of these applications have involved clustering of static data points as opposed to sequences.

Li *et.al.*[12] adopted the BIC measure for sequence clustering and demonstrated that it can be effectively used to determine both the number of clusters and the number of hidden states represented by the sequences in each cluster. However, their formulation of the BIC measure failed to accommodate for differences in sequence lengths, as the likelihood function of each sequence was used directly in the likelihood term of the BIC measure, without any normalization.

While this straightforward adaptation works well for sequences of fixed length, as demonstrated in [12], it is problematic when the sequences are of widely varying lengths. Because of its cumulative nature, the likelihood of a sequence tends to be lower for longer sequences. Thus BIC measures using likelihoods not normalized to account for sequence length are biased towards longer sequences.

To correct for this bias, we normalize the BIC measure by dividing the first term by the the length of the sequence, and adding a regularization factor α (roughly the reverse of the average sequence length) to the penalty term, resulting in what we call a *normalized* BIC measure.

For model λ_k with parameters $\hat{\theta_k}$ estimated from cluster X_k, the normalized model BIC measure is defined as:

$$BIC(X_k, \lambda_k) = \sum_{j=1}^{N_k} \frac{\log P(x_{kj}|\lambda_k, \hat{\theta_k})}{|x_{kj}|} - \alpha \times \frac{d_k}{2}\log N_k, \quad (2)$$

where x_{kj} is the jth sequence in cluster X_k, $|x_{kj}|$ is its

length, and $P(x_{kj}|\lambda_k, \hat{\theta_k})$ is the likelihood of the sequence.

Similarly, for a given partition M containing K clusters, the partition BIC measure is defined as:

$$BIC(X, M) = \sum_{i=1}^{N}\sum_{k=1}^{K} P_{ik}\frac{logP(x_i|\lambda_k, \hat{\theta_k})}{|x_i|} - \alpha \times \frac{K + \sum_{k=1}^{K} d_k}{2} \log N_k \quad (3)$$

where P_{ik} is 1 if sample x_i is in cluster k and 0 otherwise.

The process of state size selection is embedded in the **Model Construction** algorithm referred to in Table 1. The algorithm starts with a single state and increases the number of states by one at each iteration. until the BIC measure begins to decrease.

Similarly, to choose the number of clusters, the algorithm starts from one cluster and keeps increasing the number of clusters until the partition BIC measure computed by equation 3 begins to decrease (as shown in Table 1).

2.2 Outlier handling

Outliers in the context of model-based clustering refer to objects that do not belong to any of the dominant underlying clusters. Most likely these objects have been generated due to system anomaly or noise and therefore are not of primary interest.

Outliers are very common in real world data and can cause serious difficulty in model based clustering. First, mixing outliers in a "legitimate" cluster leads to the "contamination" of the model. Second, even if the algorithm is capable of isolating the outliers, they lead to a diversion of the model parameter resource. Thus very often, when there are outliers, a model based clustering algorithm that attempts to account for all data with models will only identify the outliers at the expense of failing to isolate some of the dominant regimes.

To resolve this problem, we introduce a mechanism called *the outlier pool*, detailed in Table 3. **Sample Reassignment/Outlier Detection** module in Table 3. Instead of attempting to account for all sequences with HMM models, we allow each model to reject a sequence whose likelihood is too low. A sequence that is rejected by all current models is placed in the outlier pool. The outlier pool is a special "garbage" cluster which is not modeled. To be more specific, no model is estimated from the outlier pool, and members of the outlier pool do not enter into the BIC measures. Note that this outlier pool is dynamic: objects can enter into or exit from the outlier pool as the clustering algorithm proceeds.

The threshold used to determine whether a sequence should be accepted or rejected by a model is selected based on the expected likelihood of each model, estimated using Monte Carlo simulation. For each model, 500 sequences are generated according to the model parameters. Then the normalized likelihood of each sequence against the given model is computed and the average is taken as the expected likelihood of the model.

2.3 Partition growing using DTW

As shown in Table 1, the algorithm starts with one cluster and incrementally grows the number of clusters until the partition BIC measure reaches a maximum point. For each given number of clusters, the initial set of clusters and models are adjusted using an EM procedure as outlined in Table 3, **Sample Reassignment/Outlier Detection**. Since the EM algorithm will only converge to a local optimal point, its outcome greatly depends on the initial partition. Thus a crucial step in a top-town model based clustering algorithm is the initialization of a new cluster from an existing set of clusters.

One possible strategy is to seed the new cluster with the data sample that is "least fit", i.e., farthest away from all current models (e.g., the method adopted in [12]). While this works reasonably well for clean data, it is sensitive to outliers. When there are outliers in the data, the data sample that is farthest away from all models is very likely an outlier. Thus using this strategy the cluster growing process tends to be dominated by outliers.

We have adopted a more robust alternative. Instead of evaluating each individual sequence for fitness, each cluster is evaluated as a whole. The cluster with the lowest average likelihood is identified as the candidate for splitting. The assumption is that the cluster with the lowest likelihood is most likely to be a composite cluster whose model is an average of the true underlying models.

3. Experiments

Synthesized data were generated to systematically evaluate our algorithm and compare it's performance with that of other methods, in particular for clusters of discrete-values, left-to-right HMMs estimated using segmental k-means training. Discrete HMMs constrained to transition only from left to right have proven to be particularly suitable for many real world applications [14] and segmental k-means training is a standard estimation technique. It should be noted, however, that the approach is not predicated on these choices: the algorithm and the analysis apply to more general HMMs and different choices of HMM training techniques as well.

3.1 Data description

To generate a synthesized data set, we specify the parameters of the HMM represented in each cluster, along with the number of clusters and their sizes and generate the desired number of sequences generated. Singleton clusters or clusters with a small number of samples are used to simulate outliers.

Clearly the level of difficulty in clustering a synthesized data set depends on the pair-wise distances between the generating HMMs. A number of distance measures have been developed to compare different HMMs. Here we use a slightly modified version of the symmetrized similarity measure (SSM) proposed by Juang and Rabiner [9]. Given two HMM's λ_1 and λ_2, the SSM between these two models is defined as:

$$D(\lambda_1, \lambda_2) = \frac{(L(O_2|\lambda_1) - L(O_1|\lambda_2)) + (L(O_1|\lambda_2) - L(O_2|\lambda_2))}{2}, \quad (4)$$

where O_i is a set of observation sequences generated by model λ_i and $L(O_i|\lambda_j)$ is the average normalized log-likelihood of sequences in O_i given model λ_j. It is essentially a measure of how well each model matches data generated by the other model compared to data generated by itself, and is always negative, with a larger number indicating more similar models.

The HMMs used to generate our synthesized data are discrete, left to right models as specified in section 2.1. Each model has two or three states. To generate different emission probability distributions, we first generated 6 normal distributions with deviation of 0.5 and means of $i * 2.0, 1 \leq i \leq 6$. We then calculated the probabilities for each 1.0 interval between 0.5 and 12.5 for each distribution, arriving at 6 distinct discrete probability distributions for 13 symbols. The emission probability distributions for all generating HMMs are selected from these 6 distributions. The distance between any two HMMs is controlled by the number of states that that have shared emission probabilities and the self-transition probabilities of these states.

Instead of using a fixed length for all sequences as in previous methods [18, 13, 12], we allow the length of the sequences to vary, to more closely simulate the situation in most applications. Since the HMMs are left-to-right models with a forced initial and final state, the expected sequence length for each model is essentially determined by the self-transition probabilities for the states. We adjusted these transition probabilities such that all models have an expected sequence length of 50, and allowed individual sequence lengths to vary between 30 and 100.

Two synthetic data sets were used in our evaluations, generated using 10 HMMs. Models 1 to 5 (referred to as *major models*) were used to generate dominant regimes and models 6 to 10 (referred to as *noise models*) were used to simulate outliers. Each pair of HMMs has 0 to 2 shared states, and the SSM measure ranges from -2.1 for models that are very close to each other, to -5.8 for models that are further apart. For both data sets, the sizes for the major clusters are 100,60,30,30,10 respectively. For outliers, the first data set has 5 singletons while the second one has 5 minor clusters with sizes 4,3,2,1,1.

3.2 Performance Measures

We use two performance measures to quantitatively measure the performance of our clustering algorithm. The first is the Partition Misclassification Count (PMC), proposed by Li*at.al.*[12]. This measure is a weighted sum of all different types of object misclassifications that occur in the derived partition. The smaller the count, the closer the derived partition is to the true partition and thus more accurate the clustering algorithm. While this measure can provide a good comparison between two different algorithms, it is somewhat difficult to interpret.

We thus propose another performance measure, the Difference of Concordance Matrix (DCM), which measures the mismatch between the true and derived partitions. Given a set of N objects, the concordance matrix C is a 0-1 $N \times N$ matrix where $c_{ij} = 1$ if the ith and jth objects are in the same cluster and $c_{ij} = 0$ otherwise. The DCM measure is then defined as:

$$DCM = \frac{e^T(|C_t - C_d|)e}{e^T C_t e} \quad (5)$$

where e is vector of ones, C_t and C_d are concordance matrices for the true and derived partitions, respectively, and $|\bullet|$ denotes the component-wise absolute value of a matrix. Values of DCM range from 0 to 1 with 0 indicating a perfect match and 1 indicating a complete mismatch.

3.3 Experimental results

We evaluated our algorithm using the two synthetic data sets described in Section 3.1, and compared the results to those obtained using the Matryoshka algorithm developed by Li*et.al.* [12]. Table 4 shows the performance measures of both algorithms. As can be seen from the table, our algorithm significantly outperforms the Matryoshka algorithm in both measures for both data sets.

Looking in more detail for data set 2, we found that 100% of sequences from Models 1 and 5 were clustered correctly, while 5 sequences (of 10 total) from Models 6, 9, and 10 (the outlier clusters) were mixed in with major clusters from Models 2,4,and 5, respectively. The remaining outlier sequences were identified correctly.

Table 4. Performance comparison

	PMC measure		DCM measure	
	Matryoshka	HMM/DWT	Matryoshka	HMM/DWT
Set 1	126	16	0.478	0.083
Set 2	122	10	0.413	0.026

In contrast, the Matryoshka algorithm produced a total of 8 clusters, yet failed to isolate all of the major groups: the algorithm was unable to distinguish sequences generated by Models 2 and 3, grouping 55 (of 60 total) sequences from Model 2 together with the 60 sequences from Model 3. Another identified cluster was spurious, consisting sequences from Models 2 and plus four outlier sequences.

4 Conclusions and Future Work

In this paper, we have introduced refinements to existing HMM-based clustering schemes to address important shortcomings. In particular, we interleave clustering based on a DTW-based distance measure with an HMM model-based approach. Our model-based approach allows for objective selection of both the number of clusters and the number of HMM states represented by sequences within a cluster using a normalized BIC measure, which can accommodate sequences of widely varying lengths and clusters of widely varying sizes. Our approach also allows for identification of outlier sequences in such a way that they do not detract from the identification of the major clusters. Experimental results with synthetic data show that these adaptations provide dramatic improvement in the cluster performance measures, such as the Partition Misclassification Count (PMC) and the Difference of Concordance Matrix.

Several open questions remain. First, what is the impact of the particular HMM model fitting algorithm on the cluster results? Here, we have applied a Viterbi algorithm for model fitting, but the Baum-Welch algorithm could also be used. Second, does the improved performance carry over to the case of continuous HMM and/or unconstrained HMM models? Finally, we would like to understand how explicit modeling of the state durations impacts the final cluster results, compared to characterizing durations implicitly through the HMM transition probabilities. Better understanding of these issues will aid in identification of applications where the proposed technique will be most useful.

References

[1] P. Baldi, Y. Chauvin, T. Hunkapiller, and M. McClure. Hidden Markov models of biological primary sequence information. *Proceedings of the National Academy of Sciences*, 91(3):1059–1063, 1994.

[2] C. Bregler. Learning and recognizing human dynamics in video sequences. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 568–574, 1997.

[3] S. Chen and P. Gopalakrishnan. Clustering via the bayesian information criterion with applications in speech recognition. In *Proc. IEEE Int. Conf. Acoustics, Speech,Signal Processing*, 1998.

[4] W. Chou and R. Reichl. Decision tree state tying based on penalized Bayesian information criterion. In *Proc. IEEE Int. Conf. Acoustics, Speech,Signal Processing*, pages 345–348, 1999.

[5] E. Dermatas and G. Kokkinakis. Algorithm for clustering continuous density HMM by recognition error. *IEEE Trans. Speech Audio Processing*, 4(3):231–234, May 1996.

[6] D. Heckerman, D. Geiger, and D. Chickering. A tutorial on learning with Bayesian networks. *Machine Learning*, 20:197–243, 1995.

[7] J. Hu, M. Brown, and W. Turin. Hmm based on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(10), October 1996.

[8] F. Jelinek. *Statistical methods for speech recognition*. MIT Press, 1997.

[9] B. Juang and L. Rabiner. A probabilistic distance measure for hidden Markov models. *AT&T Techinical Journal*, 64(2):391–408, February 1985.

[10] T. Kosaka, S. Masunaga, and M. Kurasaoka. Apeaker-independent phone modeling based on speaker-dependent Hmm's composition and clustering. In *Proc. IEEE Int. Conf. Acoustics, Speech,Signal Processing*, pages 441–444, 1995.

[11] A. Krog, M. Groen, I. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: applications of protein modeling. *Journal of Molecular Biology*, 235:1501–1531, February 1994.

[12] C. Li, G. Biswas, M. Dale, and pat Dale. Matryoshka: A hmm based temporal data clustering methology for modeling system dynamics. *Intelligent Data Analysis*, pages 281–308, June 2002.

[13] T. Oates, L. Firoiu, and P. Cohen. Using dynamic time warping to bootstrap hmm-based clustering of time series. In R. Sun and C. Giles, editors, *Sequence Learning, LNAI 1828*.

[14] L. Rabiner and B. Juang. *Fundations of speech recognition*. Prentice Hall, 1993.

[15] L. Rabiner, C. Lee, B. Juang, and J. Wilpon. Hmm clustering for connected work recognition. In *Proc. IEEE Int. Conf. Acoustics, Speech,Signal Processing*, 1989.

[16] A. Schliep, A. Schonhuth, and C. Steinhoff. Using hidden markov models to analyze gene exppression time course data. *Bioinformatics*, 19, 2003.

[17] G. Schwarz. Estimating the dimension of a model. *Annuals of Statistics*, 6:461–464, 1978.

[18] P. Smyth. Clustering sequences with hidden Markov models. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing*, page 648. The MIT Press, 1997.

[19] S. Zhong and J. Ghosh. A unified framework for model-based clustering. *Journal of Machine Learning Research*, (4):1001–1037, 2003.