Rival Penalized Competitive Learning for Model-Based Sequence Clustering

Martin H. Law
Department of Computer Science
Hong Kong Baptist University
Kowloon Tong, Hong Kong
martin@comp.hkbu.edu.hk

James T. Kwok
Department of Computer Science
Hong Kong Baptist University
Kowloon Tong, Hong Kong
jamesk@comp.hkbu.edu.hk

Abstract

In this paper, we propose a model-based, competitive learning procedure for the clustering of variable-length sequences. Hidden Markov models (HMMs) are used as representations for the cluster centers, and rival penalized competitive learning (RPCL), originally developed for domains with static, fixed-dimensional features, is extended. State merging operations are also incorporated to favor the discovery of smaller HMMs. Simulation results show that our extended version of RPCL can produce a more accurate cluster structure than k-means clustering.

1. Introduction

Clustering aims at dividing a set of observations into different groups so that members of the same group are more alike than members of different groups. Knowledge of this cluster structure is usually very important in understanding data with unknown distributions. Traditional clustering algorithms work only with data having static, fixed-dimensional features. However, with the increasingly widespread use of information system technologies and the Internet, there is now an explosive growth of many different varieties of databases. One example is the GenBank DNA sequence database, which contains approximately 3,400,000,000 bases in 4,610,000 sequence records as of August 1999. These databases contain sequential data, and pose a challenge to existing clustering algorithms.

One approach to cater for these sequential data is to first define a similarity measure between sequences (such as those based on trigram statistics or dynamic time warping) and then apply standard similarity-based clustering algorithms [4, 5]. However, this usually results in a loss of information and is restricted to fixed-length sequences. Experimental results also showed that only a rough clustering can be obtained. Another paradigm is to use model-based clustering. Here, clusters are represented as models, instead of represented as data prototypes in similarity-based methods. For the processing of sequences, a natural choice for the model is the hidden Markov model (HMM) [7]. This allows easy handling of variable-length sequences. Moreover, the dynamics of sequence generation in HMM has also been proved to be a reasonable approximation in many real-world problems. Previous attempts on using HMMs for sequence clustering [2, 6, 8] usually employ a mixture of HMMs, and then use EM algorithm to estimate the clusters.

However, an important problem in many clustering algorithms is on how to determine the number of clusters. In general, model selection criteria such as AIC, BIC or MDL may be used. As an example, in using HMMs for sequence clustering, Li and Biswas [3] run a number of trials with different numbers of HMMs, and then select the configuration with the highest partition mutual information. The major drawback is that they can be very time-consuming.

Another issue is the complexities of the cluster centers. As mentioned above, traditional clustering algorithms represent the centers as data prototypes and so all are equally complex. Here, however, HMMs are used and larger HMMs can, in general, yield higher likelihoods than smaller ones. Hence, the resultant HMMs discovered in the cluster structure may be larger than the generating models.

In this paper, we propose the combination of HMMs and rival penalized competitive learning (RPCL) [10] with state merging for sequence clustering. The rest of this paper is organized as follows. Brief introductions to the HMM and RPCL are given in Sections 2 and 3 respectively. Section 4 discusses how these two can be integrated. Simulation results are presented in Section 5, and the last section gives some concluding remarks.

2. Hidden Markov Model

A discrete HMM has a set of states $\{1, \ldots N\}$ and an alphabet of output symbols $\{v_1, \ldots, v_W\}$. Each state is characterized by a state-transition probability $a_{ij} = P(q_{t+1} = j | q_t = i)$ and an emission probability $b_j(s) = P(x_t =$

$v_s|q_t = j)$ for the sequence $x = \{x_1, x_2, \ldots, x_T\}$. Define $\xi_t(i,j|x,\lambda) = P(q_t = i, q_{t+1} = j|x,\lambda)$ and $\gamma_t(i|x,\lambda) = \sum_{j=1}^{N} \xi_t(i,j|x,\lambda)$. Then, given a set of sequences[1] $\{x^{(i)}\}_{i=1}^{k}$, the re-estimation formulas for a_{ij} and b_j can be found from the Baum-Welch method [7]:

$$\bar{a}_{ij}^{k} = \frac{\sum_{l=1}^{k} \sum_{t=1}^{T^{(l)}-1} \xi_t(i,j|x^{(l)},\lambda)}{\sum_{l=1}^{k} \sum_{t=1}^{T^{(l)}-1} \gamma_t(i|x^{(l)},\lambda)}, \quad (1)$$

$$\bar{b}_{j}^{k}(s) = \frac{\sum_{l=1}^{k} \sum_{t=1 \& x_t^{(l)}=v_s}^{T^{(l)}} \gamma_t(j|x^{(l)},\lambda)}{\sum_{l=1}^{k} \sum_{t=1}^{T^{(l)}} \gamma_t(j|x^{(l)},\lambda)}. \quad (2)$$

3. Rival Penalized Competitive Learning

Rival-penalized competitive learning (RPCL) [10] is an improvement over traditional competitive learning procedures by trying to determine the correct number of clusters during learning. The basic idea is that for each input pattern, not only is its winner adapted towards this input, but its rival (i.e., the first runner-up) is also moved away from this pattern (*delearned*). In the following, denote the weight vector associated with the cluster unit i by w_i. RPCL then consists of the following two steps:

1. Randomly sample a pattern x from the data. Find the winner $c = \arg\min_j \nu_j \|x - w_j\|^2$ and its rival $r = \arg\min_{j \neq c} \nu_j \|x - w_j\|^2$. Here, ν_j is called the *conscience* for unit j [10].

2. Update w_c by $\Delta w_c = \alpha_c(x - w_c)$ and w_r by $\Delta w_r = -\alpha_r(x - w_r)$, where α_c, α_r are the learning and delearning rates respectively (with $0 < \alpha_r < \alpha_c < 1$).

Experimentally, RPCL outperforms frequency sensitive competitive learning [10]. Recently, it has also been successfully used for financial time series forecasting [1].

4. Combining RPCL and HMM

4.1. The Basic Algorithm

In RPCL, as for other traditional clustering algorithms, the cluster centers are of the same type as the patterns. In this paper, however, we represent each cluster center by an HMM λ, and so the clusters are of a different type from the patterns (which are sequences). The original RPCL algorithm is thus modified as:

1. At the k-iteration, randomly sample a sequence $x^{(k)}$. Find the winner $c = \arg\min_j \nu_j d(x^{(k)}|\lambda_j)$ and its rival $r = \arg\min_{j \neq c} \nu_j d(x^{(k)}|\lambda_j)$. Here,

$$d(x^{(k)}|\lambda) = -\log P(x^{(k)}|\lambda), \quad (3)$$

and can be computed by the forward algorithm or approximated by the Viterbi algorithm [7].

2. Update the HMM parameters for the winner λ_c and its rival λ_r, such that λ_c is moved closer to the sequence $x^{(k)}$, whereas λ_r is moved further away.

[1] In the sequel, variables corresponding to the i-th sequence will be denoted using the superscript (i).

To perform step 2 above, we first define, for each λ,

$$u^{(i)} = \begin{cases} 1 & \text{if } \lambda \text{ is the winner of } x^{(i)}, \\ -\eta & \text{if } \lambda \text{ is the first runner-up for } x^{(i)}, \\ 0 & \text{otherwise,} \end{cases}$$

where $0 < \eta < 1$. For a particular λ, consider its associated subset of sequences in $\{x^{(i)}\}_{i=1}^{k}$ that λ has either been its winner or first runner-up. Consider maximizing the following weighted sum of log likelihoods:

$$P = \sum_{1 \leq i \leq k} \alpha^{k-i} u^{(i)} \log P(x^{(i)}|\lambda), \quad (4)$$

with respect to the parameters of λ. Here, $0 < \alpha < 1$ is a decaying factor which puts a stronger emphasize on the more recent sequences (as is typical for online learning algorithms). When λ is a winner, the contribution of $x^{(i)}$ on the learning of λ is positive, and λ will be moved closer to $x^{(i)}$. On the contrary, when λ is the first runner-up, the effect of $x^{(i)}$ on λ will be discounted and λ will subsequently be moved away from $x^{(i)}$. Moreover, $0 < \eta < 1$ means that the delearning rate will be smaller than the learning rate, which is in line with the original RPCL algorithm.

The standard Baum-Welch algorithm can be used to optimize (4), with the M-step in (1) and (2) modified to

$$\bar{a}_{ij}^{k} = \frac{\sum_{l=1}^{k} \alpha^{k-l} u^{(l)} \sum_{t=1}^{T^{(l)}-1} \xi_t(i,j|x^{(l)},\lambda)}{\sum_{l=1}^{k} \alpha^{k-l} u^{(l)} \sum_{t=1}^{T^{(l)}-1} \gamma_t(i|x^{(l)},\lambda)},$$

$$\bar{b}_{j}^{k}(s) = \frac{\sum_{l=1}^{k} \alpha^{k-l} u^{(l)} \sum_{t=1 \& x_t^{(l)}=v_s}^{T_l} \gamma_t(j|x^{(l)},\lambda)}{\sum_{l=1}^{k} \alpha^{k-l} u^{(l)} \sum_{t=1}^{T^{(l)}} \gamma_t(j|x^{(l)},\lambda)}.$$

These, in turn, can be rewritten as $\bar{a}_{ij}^{k} = \mu_{ij}^{(k)} / \sum_{j=1}^{N} \mu_{ij}^{(k)}$ and $\bar{b}_{j}^{k}(s) = {\nu_{js}}^{(k)} / \sum_{s=1}^{W} {\nu_{js}}^{(k)}$, with $\mu_{ij}^{(k)}$ and ${\nu_{js}}^{(k)}$ defined recursively as

$$\mu_{ij}^{(k)} = \alpha \mu_{ij}^{(k-1)} + \sum_{t=1}^{T^{(k)}-1} \xi_t(i,j|x^{(k)},\lambda), \quad (5)$$

$${\nu_{js}}^{(k)} = \alpha {\nu_{js}}^{(k-1)} + \sum_{t=1 \& x_t^{(k)}=v_s}^{T^{(k)}} \gamma_t(j|x^{(k)},\lambda). \quad (6)$$

However, notice that calculation of both $\xi_t(i,j|x^{(k)},\lambda)$ and $\gamma_t(j|x^{(k)},\lambda)$ in (5) and (6) are based on the same λ. Thus, in an online learning setting, we approximate (5) and (6) by

$$\begin{aligned}\mu_{ij}^{(k)} &= \alpha\mu_{ij}^{(k-1)} + \sum_{t=1}^{T^{(k)}-1} \xi_t(i,j|x^{(k)},\lambda^{(k-1)}),\\ \nu_{js}{}^{(k)} &= \alpha\nu_{js}{}^{(k-1)} + \sum_{t=1\&x_t^{(k)}=v_s}^{T^{(k)}} \gamma_t(j|x^{(k)},\lambda^{(k-1)}).\end{aligned}$$

4.2. Encouraging a Smaller Number of Clusters

To further encourage a smaller number of clusters in the final cluster structure, we adopt the heuristic of favoring clusters (which are HMMs) that have already been winning frequently. In other words, instead of using (3), we have

$$d(x^{(k)}|\lambda_j) = -\rho\log w_j^{(k)} - \log P(x^{(k)}|\lambda_j), \qquad (7)$$

where ρ is a user-defined parameter, and $w_j^{(k)}$ approximates the winning rate of λ_j in the past (i.e. before the k-th iteration). It is online updated as:

$$w_j^{(k+1)} = \begin{cases} \frac{M-1}{M}w_j^{(k)} + \frac{1}{M} & \text{if } \lambda_j \text{ is the winner for } x^{(k)},\\ \frac{M-1}{M}w_j^{(k)} & \text{otherwise.}\end{cases}$$

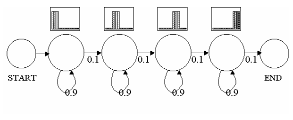
Here, M corresponds to a window size for the estimation of $w_j^{(k)}$.

4.3. Encouraging Smaller HMMs

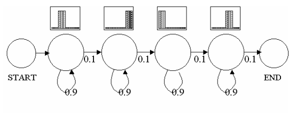
Larger HMMs can yield higher likelihoods than smaller HMMs, and so are more likely to win. To alleviate this problem, we incorporate the idea of state merging [3, 9] into RPCL. For each HMM in the clustering process, two adjacent states i and $i+1$ with closest emission probabilities (as measured by the symmetric divergence) are considered for merging. The emission probabilities of the merged state is the mean of those of i and $i+1$, while the self-looping probability is set such that the expected time to stay in the merged state remains the same as before. This new candidate HMM is then adapted for a certain period of time and, at the end, its likelihood compared to that of the original HMM. If the change is small, the original HMM will be replaced, and another round of state merging repeated.

5. Simulation

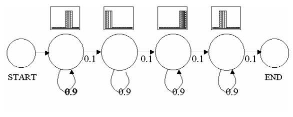
Experiments are performed on a synthetic problem with four generating HMMs and 16 symbols (Figure 1). They are not well separated, and the Bayes error is about 15%. 250 variable-length sequences are generated from each HMM. The clustering process starts with 35 HMMs as possible cluster centers. Among these 35 HMMs, 5 are of 2-state, another 5 are of 3-state, and so on. The experiment is repeated 10 times, each presenting the 1000 sequences in a different random order. The resultant non-empty clusters for a representative run by the four methods (k-means[2], k-means with conscience, RPCL and RPCL with state merging) are shown in Figure 2 Here, the top row corresponds to the 1-state HMMs, the second row corresponds to the 2-state HMMs, and so on. The size of each pie is proportional to the number of sequences won by the corresponding cluster, and the four colors correspond to sequences from the four generating HMMs. The ideal results, obtained by using only the four generating HMMs as possible cluster centers, are also shown for comparison. As can seen, RPCL locates exactly four clusters. With the incorporation of state merging, the clusters' complexities can also be correctly determined. On the other hand, both k-means procedures yield much larger numbers of clusters with varying complexities.



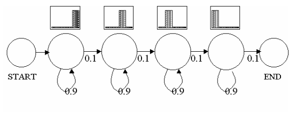
(a) HMM 1



(b) HMM 2



(c) HMM 3



(d) HMM 4

Figure 1. The four generating HMMs.

Figure 3 shows the distances (as measured by symmetric divergence) between the HMMs obtained and the four generating HMMs. Here, each row corresponds to one generating HMM, and each column corresponds to one HMM in Figure 2. The top, leftmost HMM in Figure 2 corresponds to the leftmost column in Figure 3. The remaining HMMs are then shown in the order from left-to-right, up-to-down. The whiter a particular square, the closer are the two corresponding HMMs. As can be seen, both RPCL procedures produce an unique white square in each row, indicating that each generating HMM is close to one and only one cluster. The k-means procedures, on the other hand, yield much inferior results.

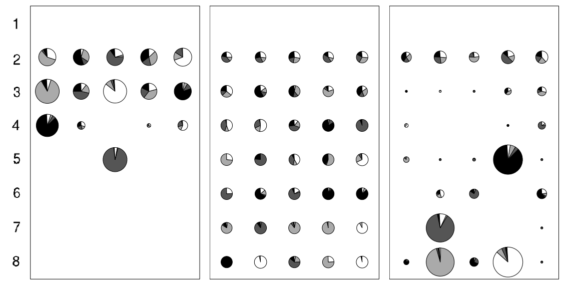
[2]Traditional k-means clustering is modified for variable-length sequence clustering in a way similar to RPCL as described in Section 4.1.

6. Conclusion

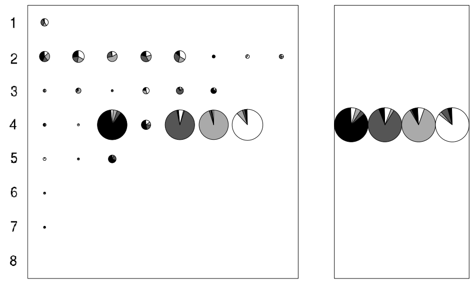
In this paper, we extend RPCL for the clustering of variable-length sequences. As in traditional domains with static, fixed-dimensional features, RPCL produces a more accurate estimation of the number of clusters. Besides, the incorporation of state merging further allows the complexities of the HMMs to be correctly determined. More experiments on real world data will be performed in the future.

Acknowledgments

This research has been partially supported by the Research Grants Council of the Hong Kong Special Administrative Region under grant HKBU2063/98E.

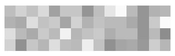


(a) k-means (b) k-means + conscience (c) RPCL

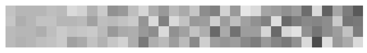


(d) RPCL + state merging (e) Ideal

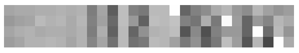
Figure 2. Clustering results.



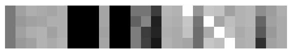
(a) k-means



(b) k-means + conscience



(c) RPCL



(d) RPCL + state merging

Figure 3. Symmetric divergence between the HMMs obtained and the generating HMMs.

References

[1] Y. Cheung, H. Lai, and L. Xu. Adaptive rival penalized competitive learning and combined linear predictor with application to financial investment. *International Journal of Neural Systems*, 8(5):141–147, 1997.

[2] A. Krogh, M. Brown, I. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, Feb. 1994.

[3] C. Li and G. Biswas. Temporal pattern generation using hidden Markov model based unsupervised classification. In *Proceedings of the Third International Symposium on Intelligent Data Analysis*, 1999.

[4] A. Manning, A. Brass, C. Goble, and J. Keane. Clustering techniques in biological sequence analysis. In *First European Symposium on Principles of Data Mining and Knowledge Discovery*, pages 315–322, 1997.

[5] T. Oates, L. Firoiu, and P. Cohen. Clustering time series with hidden Markov models and dynamic time warping. In *Proceedings of the IJCAI Workshop on Neural, Symbolic and Reinforcement Learning Methods for Sequence Learning*, 1999.

[6] L. Owsley, L. Atlas, and G. Bernard. Self-organizing feature maps and hidden Markov models for machine-tool monitoring. *IEEE Transactions on Signal Processing*, 45(11):2787–2798, 1997.

[7] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb. 1989.

[8] P. Smyth. Clustering sequences with hidden Markov models. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 648–654, San Mateo, CA, 1997. Morgan Kaufmann.

[9] A. Stolcke and S. Omohundro. Best-first model merging for hidden Markov model induction. Technical Report TR-94-003, International Computer Science Institute, University of California, Berkeley, 1994.

[10] L. Xu, A. Krzyzak, and E. Oja. Rival penalized competitive learning for clustering analysis, RBF net, and curve detection. *IEEE Transactions on Neural Networks*, 4(4):636–648, July 1993.