Clustering Time Series with Hidden Markov Models and Dynamic Time Warping

Tim Oates, Laura Firoiu and Paul R. Cohen
Computer Science Department, LGRC
University of Massachusetts, Box 34610
Amherst, MA 01003-4610
{oates,lfiroiu,cohen}@cs.umass.edu

1 Introduction

Given a source of time series data, such as the stock market or the monitors in an intensive care unit, there is often utility in determining whether there are qualitatively different regimes in the data and in characterizing those regimes. For example, one might like to know whether the various indicators of a patient's health measured over time are being produced by a patient who is likely to live or one that is likely to die. In this case, there is a priori knowledge of the number of regimes that exist in the data (two), and the regime to which any given time series belongs can be determined post hoc (by simply noting whether the patient lived or died). However, these two pieces of information are not always present.

Consider a system that produces multivariate, real-valued time series by selecting one of K hidden Markov models (HMMs), generating data according to that model, and periodically transitioning to a different HMM. Only the time series produced by the system are observable. In particular, K and the identity of the HMM generating any given time series are not observable. Given a set of time series produced by such a system, this paper presents a method for automatically determining K, the number of generating HMMs, and for learning the parameters of those HMMs.

An initial estimate of K is obtained by unsupervised clustering of the time series using dynamic time warping (DTW) to assess similarity. In addition to producing an estimate of K, this process yields an initial partitioning of the data. As later sections will explain, DTW is related to HMM training algorithms but is weaker in several respects. Therefore, the clusters based on DTW are likely to contain mistakes. These initial clusters serve as input to a process that trains one HMM on each cluster and iteratively moves time series between clusters based on their likelihoods given the various HMMs. Ultimately the process converges to a final clustering of the data and a generative model (the HMMs) for each of the clusters.

The remainder of the paper is organized as follows. Section 2 briefly reviews HMMs and the algorithms used to induce HMMs from data. Section 3 describes dynamic time warping and its use as a distance measure between multivariate time series for the purpose of unsupervised clustering. Section 4 describes an algorithm that uses DTW to bootstrap the process of fitting HMMs to data containing multiple regimes. Section 5 presents the results of experiments with this approach using artificial data, and section 6 concludes and points to future work.

2 Hidden Markov Models

A discrete hidden Markov model is defined by a set of states and an alphabet of output symbols (Rabiner 1989). Each state is characterized by two probability distributions: the transition distribution over states and the emission distribution over the output symbols. A random source described by such a model generates a sequence of output symbols as follows: at each time step the source is in one state, and after emitting an output symbol according to the emission distribution of the current state, the source jumps to a next state according to the transition distribution of its current state. Since the activity of the source is observed indirectly, through the sequence of output symbols, and the sequence of states is not directly observable, the states are said to be hidden. A continuous HMM is different from a discrete one in that the output symbols are emitted from a probability density instead of a distribution. We prefer the discrete models because they are simpler. For both discrete and continuous HMMs dynamic programming algorithms exist for:

- computing the probability of observing a sequence, given a model
- finding the state sequence that maximizes the probability of the given sequence, when the model is known(the Viterbi algorithm)
- inducing the HMM that maximizes (locally) the probability of the given sequence (the Baum-Welch algorithm, an expectation-maximization algorithm)

The HMM model definition can be readily extended to the multidimensional case, where a vector of symbols is emitted at each step, instead of a single symbol. The assumption that allows this immediate extension is conditional independence of variables given the state. While hidden Markov models are richer representations of time series (they add a vocabulary of states), the induction algorithm has two major weaknesses:

- the number of states must be set in advance, i.e. the structure of the model is not fit to the data, but given a priori
- the algorithm converges to a local maximum only

These two problems led us to using the dynamic time warping technique for alleviating them, as presented in section 4.

3 Dynamic Time Warping

This section describes dynamic time warping, an algorithm that is less well known than Baum-Welch and Viterbi, and its use as a measure of similarity for unsupervised clustering of time series. Let S denote a multivariate time series spanning n time steps such that $S = \{s_t | 1 \leq t \leq n\}$. The s_i are vectors of values containing one element for the value of each of the component univariate time series at time i. Given a set of m multivariate time series, we want to obtain, in an unsupervised manner, a partition of these time series into subsets such that each subset corresponds to a qualitatively different regime.

If an appropriate measure of the similarity of two time series is available, clustering followed by prototype extraction is a suitable unsupervised learning method for this problem. Finding such a measure of similarity is difficult because time series that are qualitatively the same may be quantitatively different in at least two ways. First, they may be of different lengths, making it difficult or impossible to embed the time series in a metric space and use, for example, Euclidean distance to determine similarity. Second, within a single time series, the rate at which progress is made can vary non-linearly. The same pattern may evolve slowly at first and then speed up, or it may begin quickly and then slow down. Such differences in rate make similarity measures such as cross-correlation unusable.

DTW is a generalization of classical algorithms for comparing discrete sequences (e.g. minimum string edit distance (Corman, Leiserson, & Rivest 1990)) to sequences of continuous values (Sankoff & Kruskall 1983). It was used extensively in speech recognition, a domain in which the time series are notoriously complex and noisy, until the advent of Hidden Markov Models which offered a unified probabilistic framework for the entire recognition process (Jelinek 1997).

Given two time series, S_1 and S_2, DTW finds the warping of the time dimension in S_1 that minimizes the difference between the two series. Consider the two univariate time series shown in Figure 1. Imagine that the time axis of S_1 is an elastic string, and that you can grab that string at any point corresponding to a time at which a value was recorded for the time series. Warping of the time dimension consists of grabbing one of those points and moving it to a new location. As the point moves, the elastic string (the time dimension) compresses in the direction of motion and expands in the other direction. Consider the middle column in Figure 1. Moving the point at the third time step from its original location to the seventh time step causes all of the points to its right to compress into the remaining available space, and all of the points to its left to fill the newly created space. Of course, much more complicated warpings of the time dimension are possible, as with the third column in Figure 1 in which four points are moved.

Given a warping of the time dimension in S_1, yielding a time series that we will denote S_1', one can compare the similarity of S_1' and S_2 by determining the area between the two curves. That area is shown in gray in the bottom row of Figure 1. Note that the first warping of S_1 in which a single point was moved results in a poor match, one with a large area between the curves. However, the fit given by the second, more complex warping is quite good. In general, there are exponentially many ways to warp the time dimension of S_1. DTW uses dynamic programming to find the warping that minimizes the area between the curve in time that is a low order polynomial of the lengths of S_1 and S_2, i.e. $O(|S_1||S_2|)$.

DTW returns the optimal warping of S_1, the one that minimizes the area between S_1' and S_2, and the area associated with that warping. The area is used as a measure of similarity between the two time series. Note that this measure of similarity handles nonlinearities in the rates at which experiences progress and is not affected by differences in the lengths of experiences. In general, the area between S_1' and S_2 may not be the same as the area between S_2' into S_1. We use a symmetrized version of DTW that essentially computes the average of those two areas based on a single warping (Kruskall & Liberman 1983). Although a straightforward implementation of DTW is more expensive than computing Euclidean distance or cross-correlation, there are numerous speedups that both improve the properties of DTW as a distance metric and make its computation nearly linear in the length of the time series with a small constant.

Given m time series, we can construct a complete pairwise distance matrix by invoking DTW $m(m-1)/2$ times (the factor of 2 is due to the use of symmetrized DTW). We then apply a standard hierarchical, agglomerative clustering algorithm that starts with one cluster for each time series and merges the pair of clusters with the minimum average intercluster distance (Everitt 1993). Without a stopping criterion, merging will continue until there is a single cluster containing all m experiences. To avoid that situation, we do not merge clusters for which the mean intercluster distance is significantly different from the mean intracluster distance as measured by a t-test. The number of clusters remaining when this process terminates is K, the number of regimes in the time series.

Finally, for each cluster we select a prototype. Two methods commonly used are to choose the cluster member that minimizes the distance to all other members of the cluster, and to simply average the members of the cluster. The advantage of the latter method is that it smooths out noise that may be present in any individual data item. Unfortunately, it is only workable when the cluster elements are embedded in a metric space (e.g.

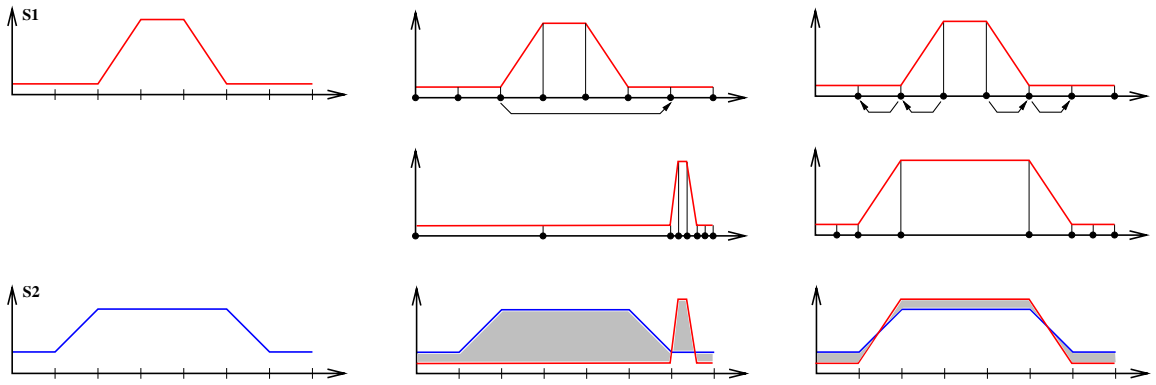

Figure 1: Two time series, S_1 and S_2, (the leftmost column) and two possible warpings of S_1 into S_2 (the middle and rightmost columns).

Cartesian space). Although we cannot embed experiences in a metric space, DTW allows us to use a combination of the two methods as follows. First, we select the time series that minimizes distance to all other time series in a given cluster. Then we warp all other patterns into that centroid, resulting in a set of patterns that are all on the same time scale. It is then a simple matter to take the average value at each time point over all of the series and use the result as the cluster prototype.

4 Clustering with DTW and HMMs

This section presents an algorithm for clustering time series using only HMMs, and then shows how the utility of that algorithm is greatly enhanced by the information obtained by first clustering the time series with DTW.

4.1 Clustering with HMMs

The assumption underlying our method of clustering with HMMs is that all of the sequences that belong in a cluster were generated by the same HMM and, as such, have high probabilities under this HMM. If a sequence has a high probability under a model, we consider it to be generated, or "accepted", by the model. If it has a low probability we consider it to be "rejected". We apply a simple statistical test to check whether an observed sequence O is generated by a given model λ. We generate a large sample of sequences from the model λ. From this sample we calculate the empirical probability distribution (see "Computer Intensive Methods" in (Cohen 1995)) of $\log(P(o|\lambda)$, the log-likelihood of the sequences generated by the model. Let L be the log-likelihood of O under the model, i.e. $L = log(P(O|\lambda)$. We then test the hypothesis that L is drawn from the probability distribution of the log-likelihood of the sequences generated by the model λ. Specifically, we test that:

$$P_\lambda(log - likelihood \leq L) > threshold$$

and reject the null hypothesis if the probability is below the threshold. If the hypothesis is rejected and L is considered not to be drawn from the probability distribution of the log-likelihood of the sequences generated by λ, then we infer that the sequence O is not accepted by the model λ.

Due to the above assumption, the task of clustering the sequences is equivalent with the task of finding a set of hidden Markov models that accept disjoint subsets of the original set of sequences. A set of sequences can be clustered by fitting an HMM to all the sequences in the set and then applying a fixed point operation that refines the HMM and "shrinks" the initial set to the subset of sequences accepted by the resulting HMM. Given a set S of sequences and a model HMM, the fixed-point operation is:

- $S_0, S_0' \longleftarrow S$
- repeat
  - $S_0 \longleftarrow S_0'$
  - re-train the HMM with S
  - $S_0' \longleftarrow$ the sequences in S_0 accepted by the HMM
- until $S_0 = S_0'$

Clustering of the set S proceeds then by repeating the fixed point operation for the set $(S \setminus S_0)$ of remaining sequences and so forth, until no sequence remains unassigned.

The fixed-point operation converges because at each iteration, S can either shrink or stay the same. In the extreme case, S is reduced to one sequence only and not to the empty set, because it is unlikely that an HMM trained exactly with that sequence will not accept it.

4.2 Clustering with DTW + HMMs

When fitting an HMM to a set of sequences, the induction algorithm will try to fit all the sequences in the set equally well. Because the number of states is set in advance and not learned from the data, it is not clear how the states are "allocated" to the different sequences. It is likely that the states' observation probability distributions will cover the regions in the observation space most often visited by the given sequences and that the state probability transitions will be changed accordingly.

This means that if the set contains sequences generated by distinct models, it is likely that the induced HMM will be a "compromise" between the original models (the most frequent states of either generating model will appear in the learned model). It is not clear what this compromise model is. Because the training algorithm converges to a local maximum, the resulting HMM is highly dependent on the initial model from which the training algorithm starts.

Therefore, if we assume that the sequences in a training set were generated by some hidden Markov models and our task is to identify these models, then it is advantageous to start the HMM clustering algorithm with even an approximate initial clustering. If the majority of sequences in the initial cluster come from the same model, then it is likely that the learned compromise HMM will be closer to this one model. Since the DTW clustering technique can provide a good initial partition, the HMM clustering algorithm is initialized with it. For each cluster in the DTW partitioning, an HMM is created by applying the fixed-point operation described in the previous section to the sequences of the cluster. The remaining sequences from each DTW cluster are then checked against the HMMs of the other DTW clusters. Finally, if any sequences are still unassigned to an HMM, they are placed in a set that is clustered solely by HMM clustering.

5 Experiments

We tested our algorithm on an artificial dataset generated as in (Smyth 1997), from two hidden Markov models. The two hidden Markov models that generated the artificial dataset each have two states, one that emits one symbol from the normal density with mean 0 and variance 1, $\mathcal{N}(0,1)$, and one that emits one symbol from $\mathcal{N}(3,1)$. The two models differ in their transition probability matrices. These matrices are:

$$A_{HMM_1} = \begin{pmatrix} .6 & .4 \\ .4 & .6 \end{pmatrix}$$

$$A_{HMM_2} = \begin{pmatrix} .4 & .6 \\ .6 & .4 \end{pmatrix}$$

As explained in (Smyth 1997) this is only apparently an easy problem.

Because the output of the states is continuous and we implemented our clustering algorithm with discrete HMMs, we discretized the output values with a Kohonen network with 20 units (so the output alphabet has 20 symbols in our experiments). Again as in (Smyth 1997), the training set consists of 40 sequences. The first 20 are generated by HMM_1 and the last 20 by HMM_2. Each sequence has length 200.

The clusters resulting from DTW alone are shown below. For each cluster the indices of the time series belonging to that cluster are shown. Ideally, cluster 1 would contain indices 0 through 19 and cluster 2 would contain indices 20 through 39.

- cluster 1: 1 2 3 4 5 6 9 10 11 12 13 15 17 18 19 23 24 33 35 37
- cluster 2: 0 7 8 14 16 20 21 22 25 26 27 28 29 30 31 32 34 36 38 39

The resulting DTW+HMM clustering is:

- cluster 1: 1 2 3 4 5 6 9 10 11 13 14 15 16 18
- cluster 2: 20 21 22 24 26 27 29 30 31 34 36 37 38 39
- cluster 3: 0 8 12 17 19 23 25 28 33
- cluster 4: 7 32 35

The transition matrices of the four HMMs are:

$$A_{HMM_1} = \begin{pmatrix} .69 & .30 \\ .38 & .61 \end{pmatrix}$$

$$A_{HMM_2} = \begin{pmatrix} .29 & .70 \\ .64 & .35 \end{pmatrix}$$

$$A_{HMM_3} = \begin{pmatrix} .55 & .44 \\ .53 & .46 \end{pmatrix}$$

$$A_{HMM_4} = \begin{pmatrix} .49 & .50 \\ .46 & .53 \end{pmatrix}$$

It can be noticed that the HMM clustering "cleans" the clusters obtained by DTW. For example, the sequences 33, 35, 37, that appear in the first cluster in the DTW partitioning, are removed by the HMM clustering, and the resulting DTW+HMM cluster has only sequences generated by the first model. The second cluster has only sequences generated by the second model, too. It can also be noticed that when HMM clustering alone is applied for the sequences removed from the DTW clusters, the resulting clusters, 3 and 4, have mixed sequences. Thus, HMM clustering alone does not work well: when trained with mixed sequences a compromise HMM is learned, rather than an HMM close to one of the true models. The transition matrices for the first two models are very different: each HMM fits the idiosyncrasies of the sequences emitted by the true models. As for the last two models, each of them is a compromise between the two original HMMs. The above results indicate that further improvement is achievable by two ways:

- by alternating DTW and HMM clustering in repeated iterations
- by trying to apply the fixed point operation to "grow" each resulting HMM in the DTW+HMM partitioning with sequences from other clusters; in preliminary experiments, the first HMM would "steal" good sequences from the last two clusters, but the process does not always converge; we intend to explore this further

We also applied the technique to clustering time series of robot sensor values. The time series were collected during several simple experiences: the robot was approaching, pushing or passing an object. While we do not know the true clusters in the robot data, we considered a good clustering one which reflects the kinds of

experiences enumerated above. We observed the same effect of "cleaning" the DTW clusters by the HMM, but the set of sequences removed by the HMM fixed-point operation was large and poorly clustered by the HMM clustering method. We think that the problem of the unknown number of HMM states must be solved before trying to cluster and represent real data with HMMs. We plan to apply the minimum description length principle for tackling this difficult problem.

6 Conclusion

We presented a hybrid time series clustering algorithm that uses dynamic time warping and hidden Markov model induction. The algorithm worked well in experiments with artificial data. The two methods complement each other: DTW produces a rough initial clustering and the HMM removes from these clusters the sequences that do not belong to them. The downside is that the HMM removes some good sequences along with the bad ones. We suggested possible ways of improving the method and are currently working on validating them.

References

Cohen, P. R. 1995. *Empirical Methods for Artificial Intelligence*. Cambridge: The MIT Press.

Corman, T. H.; Leiserson, C. E.; and Rivest, R. L. 1990. *Introduction to Algorithms*. MIT Press.

Everitt, B. 1993. *Cluster Analysis*. John Wiley & Sons, Inc.

Jelinek, F. 1997. *Statistical Methods for Speech Recognition*. MIT Press.

Kruskall, J. B., and Liberman, M. 1983. The symmetric time warping problem: From continuous to discrete. In *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley.

Rabiner, L. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–285.

Sankoff, D., and Kruskall, J. B. 1983. *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley.

Smyth, P. 1997. Clustering sequences with hidden markov models. In *Advances in Neural Information Processing 9*.