Designing the Minimal Structure of Hidden Markov Model by Bisimulation

Manuele Bicego, Agostino Dovier, and Vittorio Murino

Dip. di Informatica, Univ. di Verona
Strada Le Grazie 15, 37134 Verona, Italy
{bicego,dovier,murino}@sci.univr.it

Abstract. Hidden Markov Models (HMMs) are an useful and widely utilized approach to the modeling of data sequences. One of the problems related to this technique is finding the optimal structure of the model, namely, its number of states. Although a lot of work has been carried out in the context of the model selection, few work address this specific problem, and heuristics rules are often used to define the model depending on the tackled application. In this paper, instead, we use the notion of probabilistic bisimulation to automatically and efficiently determine the minimal structure of HMM. Bisimulation allows to merge HMM states in order to obtain a minimal set that do not significantly affect model performances. The approach has been tested on DNA sequence modeling and 2D shape classification. Results are presented in function of reduction rates, classification performances, and noise sensitivity.

1 Introduction

Hidden Markov Models (HMMs) represent a widespread approach to the modeling of sequences: they attempt to capture the underlying structure of a set of symbol strings. HMMs can be viewed as stochastic generalizations of finite-state automata, when both transitions between states and generation of output symbols are governed by probability distributions [1].

The basic theory of HMMs was developed by Baum *et al.* [2,3] in the late 1960s, but only in the last decade it has been extensively applied in a large number of problems. A non-exhaustive list of such problems consists of speech recognition [1], handwritten character recognition [4], DNA and protein modelling [5], gesture recognition [6] and, more in general, behavior analysis and synthesis [7].

HMMs fit very well in a large number of situations, in particular where the state sequence structure of the process examined can be assumed to be Markovian. Unfortunately, there are some drawbacks [8]. First, the iterative technique for the HMM learning (*Baum-Welch re-estimation*) converges to a local optimum, not necessarily the global one, and the choice of appropriate initial parameters' estimates is crucial for convergence. Second, a large amount of training data

M.A.T. Figueiredo, J. Zerubia, A.K. Jain (Eds.): EMMCVPR 2001, LNCS 2134, pp. 75–90, 2001.

is generally necessary to estimate HMM parameters. Finally, the HMM topology and number of states have to be determined prior to learning, and usually heuristic rules are pursued for this purpose (e.g., [9]). This paper proposes a novel approach for resolving this final problem, in particular to determine the number of states. This issue could be tackled by using traditional methods of model selection; numerous paradigms have been proposed in this context, a non-exhaustive list includes [10]: *Minimum Description Length* (MDL), *Bayesian Inference Criterion* (BIC), *Minimum Message Length* (MML), *Mixture Minimal Description Length* (MMDL), *Evidence Based Bayesian* (EBB) etc.. More computational intensive approaches are stochastic approaches (e.g., *Markov Chain Monte Carlo* (MCMC)), re-sampling based schemes, and cross-validation methods. Although principally derived for fitting mixture models, many of these techniques could be applied also in the HMM context, as proposed in [11] and [12]. It is worth noting that these approaches are devoted to find the optimal model on the basis of a criterion function by exploring all (or a large part of) the search space. Our work proposes instead a direct method to identify the model without searching the whole space, resulting less computationally intensive. In [11], starting with redundant configuration, an optimal structure can be obtained by repeated Bayesian merging of states in an incremental way, as far as new evidence arrives. In [12], a method for simultaneous learning of HMM structure and parameters is proposed. Parameters' uncertainty is minimized by introducing an entropic prior and Maximum a Posteriori Probability (MAP) estimation. In this way, redundant parameters are eliminated and the model becomes sparse; moreover posterior probability increases, and an easier interpretation of resulting architecture is allowed.

Our approach consists in eliminating syntactic redundancy of an Hidden Markov Model using a technique called bisimulation. Bisimulation is a notion of equivalence between graphs whose usefulness has been demonstrated in various fields of Computer Science. In Concurrency it is used for testing process equivalence [18], in Model-Checking as a notion of equivalence between Kripke Structures [20], in Web-like databases for providing operational semantics to query languages [17], in Set Theory, for replacing extensionality in the context of non well-founded sets [13].

With our approach, the structure of an HMM is reduced by computing bisimulation equivalence relation between states of the model, so that equivalent states can be collapsed. We employed both the notions of probabilistic and standard bisimulation. We will prove that bisimulation reduces the number of states without significant loss in term of likelihood and classification accuracy. We will test this approach reporting experiments on DNA sequence modelling and 2D shape recognition using chain code. We will show that the proposed procedure is fully automatic, efficient, and provides promising results. We also compare our approach with BIC (Bayesian Inference Criterion) method, which is equivalent to MDL [10], showing that this technique is nearly as acceptable as our, as far as classification accuracy is concerned, but is more computationally demanding.

The rest of the paper is organized as follows: Sect. 2 contains formal description of HMM. In Sect. 3, the notion of bisimulation and the algorithm to compute equivalence classes are described. In Sect. 4 we detail our strategy and in Sect. 5 experiments and results are presented. Finally, Sect. 6 contains conclusions and future perspectives.

2 Hidden Markov Models

An HMM is formally defined by the following elements (see [1] for further details):

- *A set* $S = \{S_1, S_2, \cdots, S_N\}$ *of (hidden) states.*
- *A state transition probability distribution*, also called transition matrix $A = \{a_{ij}\}$, representing the probability to go from state S_i to state S_j.

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i] \qquad 1 \leq i, j \leq N \tag{1}$$

  with $a_{ij} \geq 0$ and $\sum_{j=1}^{N} a_{ij} = 1$.
- *A set* $V = \{v_1, v_2, \cdots, v_M\}$ *of observation symbols.*
- *An observation symbol probability distribution*, also called emission matrix $B = \{b_j(k)\}$, indicating the probability of emission of symbol v_k when system state is S_j.

$$b_j(k) = P[v_k \text{ at time } t \,| q_t = S_j] \qquad 1 \leq j \leq N, 1 \leq k \leq M \tag{2}$$

  with $b_i(k) \geq 0$ and $\sum_{j=1}^{M} b_j(k) = 1$.
- *An initial state probability distribution* $\pi = \{\pi_i\}$, representing probabilities of initial states.

$$\pi_i = P[q_1 = S_i] \qquad 1 \leq i \leq N \tag{3}$$

  with $\pi_i \geq 0$ and $\sum_{i=1}^{N} \pi_i = 1$. For convenience, we denote an HMM as a triplet $\lambda = (A, B, \pi)$, which determines uniquely the model.

3 Bisimulation

Bisimulation is a notion of equivalence between graphs useful in several fields of Computer Science. The notion was introduced by Park for testing process equivalence, extending a previous notion of automata simulation by Milner. Milner then employed bisimulation as the core for establishing observational equivalence of the Calculus of Communicating Systems [18].

Kanellakis and Smolka in [16] relate the bisimulation problem with the general (relational) coarsest partition problem and pointed out that the partition refinement algorithm in [19] solves this task. More precisely, in [19] Paige and Tarjan solve the problem in which the stability requirement is relative to a relation E (edges) on a set N (nodes) with an algorithm whose complexity is $O(|E| \log |N|)$.

Standard Bisimulation. Bisimulation can be equivalently formulated as a relation between two graphs and as a relation between nodes of a single graph. We adopt the latter definition since we are interested in reducing states of a unique graph.

Definition 1. *Given a graph* $G = \langle N, E\rangle$ *a* bisimulation *on* G *is a relation* $b \subseteq N \times N$ *s.t. for all* $u_0, u_1 \in N$ *s.t.* $u_0\, b\, u_1$ *and for* $i = 0, 1$*: if* $\langle u_i, v_i\rangle \in E$*, then there exists* $\langle u_{1-i}, v_{1-i}\rangle \in E$ *s.t.* $v_0\, b\, v_1$*.*

In order to minimize the number of nodes of a graph, we look for the maximal bisimulation $\equiv$ on G. Such a maximal bisimulation always exists, it is unique, and it is an equivalence relation over the set of nodes of G [13]. The minimal representation of $G = \langle N, E\rangle$ is therefore the graph:

$$\langle N/\equiv, \{\langle [m]_\equiv, [n]_\equiv\rangle : \langle m, n\rangle \in E\}\rangle$$

which is usually called the *bisimulation contraction of* G. Using the algorithm in [19] the problem can be solved in time $O(|E| \log |N|)$; for acyclic graphs and for some classes of cyclic graphs it can be solved in linear time w.r.t. $|N| + |E|$ [15].

Bisimulation on labeled graphs. If the graphs are such that nodes and/or edges are labeled, the notion can be reformulated as follows:

Definition 2. *Let* $G = \langle N, E, \ell\rangle$ *be a graph with a labeling function* ℓ *for nodes, and labeled edges of the form* $m \xrightarrow{a} n$ *(a belongs to a set of labels). A* bisimulation *on* G *is a relation* $b \subseteq N \times N$ *s.t. for all* $u_0, u_1 \in N$ *s.t.* $u_0\, b\, u_1$ *it holds that:* $\ell(u_1) = \ell(u_2)$ *and for* $i = 0, 1$*, if* $u_i \xrightarrow{a} v_i \in E$*, then there exists* $u_{1-i} \xrightarrow{a} v_{1-i} \in E$ *s.t.* $v_0\, b\, v_1$*.*

If only the nodes are labeled, the procedure in [19] can be employed to find the bisimulation contraction, provided that in the initialization phase nodes with the same labels are put in the same class. The case in which edges are labeled can be reduced to the last one by replacing a labeled edge $m \xrightarrow{a} n$ by a new node ν labeled by a and by the edges $\langle m, \nu\rangle$ and $\langle \nu, n\rangle$. Therefore, finding the bisimulation contraction also in this case can be done using the algorithm of [19]; moreover, the procedure of [19] can be modified in order to deal directly (i.e., without preprocessing) with the general case described.

Probabilistic Bisimulation The notion of bisimulation over labeled graphs (Def. 2) has been introduced in a context where labels denote actions executed (e.g. a symbol is emitted) by processes during their run. Labels can also store pairs of values $\langle x, y\rangle$: an action x and a probability value y (that could be read as: this edge can be crossed with probability y and in this case an action x is done). In this case another notion of bisimulation is perhaps more suitable. Consider, for instance, the graph of Fig. 1 (we use n_1–n_8 to refer to the nodes: they are not labels). n_7 and n_8 are trivially equivalent since they have no outgoing edges. Nothing can be done in both the cases. The four nodes n_2, n_3, n_5, n_6 are in the same equivalence class, since they have equivalent successors (reachable performing the same action b, with probability 1). The nodes n_1 and n_4 are instead not

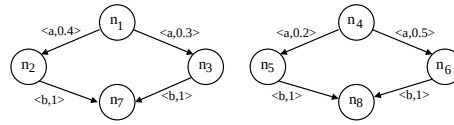


Fig. 1. n_1 and n_4 are not bisimilar, but probabilistically bisimilar.

equivalent, since, for instance, there is the edge $n_1 \overset{\langle a,0.4\rangle}{\rightarrow} n_3$ but no edges labeled $\langle a, 0.4\rangle$ starts from n_4. However, both from n_1 and n_4 it can be reached one of the equivalent states, performing action a with probability 0.7: the two nodes should be considered equivalent. These graphs are called *Fully Probabilistic Labelled Transition System* (FPLTS).

The notion of *probabilistic bisimulation* [14] is aimed at formally justifying this intuitive concept. We start by providing two auxiliary notions: Given a graph $G = \langle N, E\rangle$ with edge labeled by pairs as above, and $b \subseteq N \times N$ a relation, then for two nodes $m, n \in N$ and a symbol a, we define the functions B and S as follows

$$B(m,n,a) = \{\mu : \exists q(m \overset{\langle a,q\rangle}{\rightarrow} \mu \in E \wedge \mu\, b\, n)\} \text{ and } S(m,n,a) = \sum_{m \overset{\langle a,q\rangle}{\rightarrow} \mu \in E, \mu b n} q$$

Definition 3. *Let $G = \langle N, E\rangle$ be a graph with edge labeled by pairs consisting of symbols and probability values, a* probabilistic bisimulation *on G is a relation $b \subseteq N \times N$ s.t.: for all $u_0, u_1 \in N$, if $u_0\, b\, u_1$ then for $i = 0, 1$ if $u_i \overset{\langle a,p\rangle}{\rightarrow} v_i \in E$, then then there exists $v_{1-i} \in N$ s.t.:*

- $u_{1-i} \overset{\langle a,p'\rangle}{\rightarrow} v_{1-i} \in E$,
- $S(u_i, v_i, a) = S(u_{1-i}, v_{1-i}, a)$, *and*
- *and for all $m \in B(u_i, v_i, a)$ and $n \in B(u_{1-i}, v_{1-i}, a)$ it holds that $m\, b\, n$.*

In [14] a modification of the Paige-Tarjan procedure is presented in this case and proved to correctly return the probabilistic contraction of a graph $G = \langle N, E\rangle$ in time $O(|N||E| \log |N|)$. In the example of Fig. 1 the two nodes n_1 and n_4 are put in the same class.

In this paper we will further extend the possible labels for edges. We admit triplets $\langle p_1, a, p_2\rangle$ where a is a symbol while p_1 and p_2 are probabilistic values. We extend the notion of the above Definition 3 point to point. In other words, we reason as if the edge $\langle p_1, a, p_2\rangle$ is replaced by the two edges $\langle a, p_1\rangle, \langle \hat{a}, p_2\rangle$ and $\hat{a}$ can not be confused with a (see Fig. 2).

4 The Strategy

HMM as labeled graphs. Probabilistic bisimulation is defined on FPLTS, which are slightly different from HMMs. Neglecting notation, the real problem is represented by emission probability of each state, which has not counterpart in

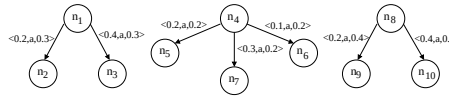


Fig. 2. n_1 and n_4 are probabilistically bisimilar. n_1 and n_8 are not.

FPLTS. As described in Sect. 3, we can solve the problem by choosing an appropriate initial partition, whose sets contains states with same emission probability and then run the algorithm of [19]. This approach is correct, but it is too restrictive with respect to the concept of probabilistic bisimulation. In other words, using this initialization we create classes of bisimulation equivalence using concept of syntactic labelling, loosing instead the semantic labeling concept, which is the kernel of the probabilistic bisimulation.

Thus, we propose another method, a bit more expensive in terms of memory allocation and computational cost, but offering a better semantic characterization.

Definition 4. *Given a HMM $\lambda = (A, B, \pi)$, trained with a set of strings from an alphabet $V = \{v_1, v_2, \cdots, v_M\}$, the equivalent FPLTS is obtained as follows. For each state S_i:*

- *Let A_i be the set of edges outgoing from the state S_i, defined as*

$$A_i = \{\langle S_i, S_j \rangle : a_{ij} \neq 0,\ 1 \leq j \leq N\}$$

- *each edge e in A_i is replaced by M edges, whose labels are $\langle a_{ij}, v_k, B_i(k) \rangle$, where, for $1 \leq i, j \leq N$, $1 \leq k \leq M$:*
  - *a_{ij} is probability of e;*
  - *v_k is k-th symbol of V;*
  - *$B_i(k)$ is probability of emission of v_k from state S_i.*

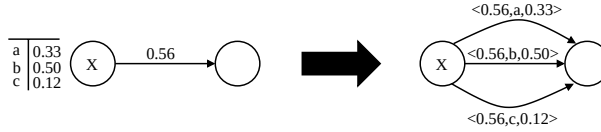


Fig. 3. Basic idea of procedure to represent HMM as a FPLTS.

Given an HMM with N states, K edges and M symbols, with this approach the complexity of bisimulation contraction grows from $O(KNlogN)$ to $O(MKNlogN)$ for time, and from $O(KN)$ to $O(MKN)$ for space.

By applying bisimulation to a HMM we have to face another important issue: the partial control of compression rate of our strategy. To this end, we introduce the concept of *quantization* of probability: given a set of quantization level values (prototypes) in the interval $[0, 1]$, we approximate each probability with the

closest prototype. A uniform quantization is adopted on interval $[0, 1]$. To control this approximation we define a *reduction factor*, representing the number of levels that subdivide the interval: it is calculated as *(number of prototypes - 2)*. For example, reduction factor 3 means that probability are approximated with the values $\{0, 0.25, 0.5, 0.75, 1\}$. Thus, the notion of equivalent labels is governed by the test of equality of their quantization, where $quant(p)$ is define as the prototype j closest to p.

As a final consideration, the reduction factor represents a tuning parameter for deciding the degree of compression adopted. Obviously, for a low value of the factor, information lost in approximation is high, and the resulting model can be a very poor representation of the original one.

Algorithm. Given a problem, determining optimal number of HMM states is performed following the following steps:

1. Training of HMM with a number of states that is reasonably large with respect to the problem considered. This number strongly depends from available data, and it can be determined using some heuristics.
2. Transform HMM in labelled graph (FPLTS), using procedure described in Def. 4 of Sect. 4. In this step we have to choose a reduction factor, that provides a measure of accuracy adopted in the conversion. It also gives a rough meaning of reduction rate: lower precision likely means higher compression.
3. Run bisimulation algorithm on such graph, obtaining equivalence classes. Optimal number of states N' is represented by cardinality of the quotient set (i.e. the number of different classes determined by bisimulation).
4. Retraining of the HMM using N' states.

This method is designed for discrete HMM, but can be generalized for other typologies by working on Step 2 of the procedure.

5 Experimental Results

The aim of the following experiments is to show that this method reduces HMM states without significant loss in terms of likelihood and classification accuracy. We tested these two properties on two distinct problems: DNA modeling, i.e. using HMM to model and recognize different DNA sequences (typically, fragments of genes), and 2D shape classification using chain code (modeled by HMM). In all tests, each HMM was trained in three learning sessions, using Baum-Welch re-estimation and choosing the one presenting the maximum likelihood. Each learning started using random initial estimates of A, B and π and ended when likelihood is converged or after 100 training cycles. Performances are measured in terms of some indices:

– *Compression Rate*, representing a percentage measure of the number of states eliminated by bisimulation: $CR = 100\left(\frac{N_{orig}-N_{reduct}}{N_{orig}}\right)$, where N_{reduct} are the number of states after bisimulation on a HMM with N_{orig} states;

– *Log Likelihood Loss*, estimating the difference in LL between original and reduced HMM: $LLL = 100\left(\frac{LL_{orig}-LL_{reduct}}{LL_{orig}}\right)$, where LL_{reduct} and LL_{orig} are log likelihood of HMM with N_{reduct} and N_{orig} number of states, respectively.

5.1 DNA Modeling

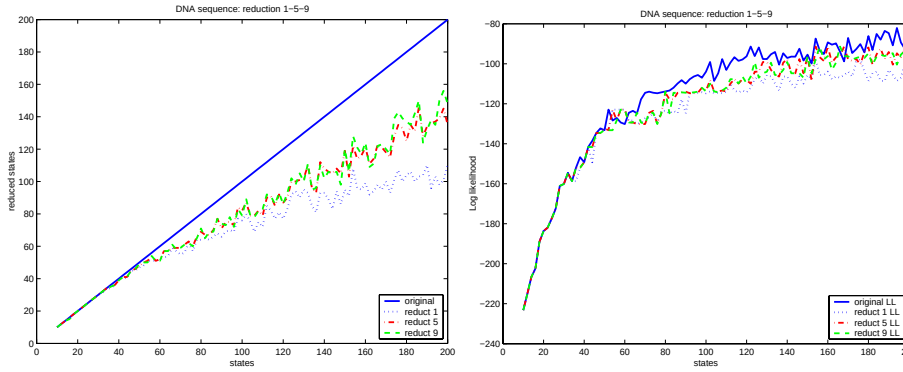
Genomics offers tremendous challenges and opportunities for computational scientists. DNA are sequences of various lengths formed by using 4 symbols: A, T, C, and G. Each symbol represent a base, *Adenine, Thymine, Cytosine*, and *Guanine* respectively. Recent advances in biotechnology have produced enormous volumes of DNA related information, needing suitable computational techniques to manage them [21].

From a machine learning point of view [22], there are three main problems to deal with : *genome annotation*, including identification of genes and classification into functional categories, *computational comparative genomics*, for comparing complete genomic sequences at different levels of detail, and *genomic patterns*, including identification of regular pattern in sequence data. Hidden Markov Models are widely used in resolving these problems, in particular for classification of genes, protein family modeling, and sequence alignment. This is because they are very suitable in modeling strings (as DNA or protein sequences), and can provide useful measures of similarity (LL) in comparing genes.

In this paper, we employ HMM to model gene sequences for classification purposes. This simple example is nevertheless significant to demonstrate HMM ability in recognizing genes, also in conditions of noise (as biological mutations). Data were obtained extracting a 200 bp (base pair) fragment of *recA* gene sequence of a lactobacillus. We trained 95 HMMs on this sequence, where N (number of states) grows from 10 to 200 (step 2). We applied the bisimulation contraction algorithm on each HMM, with reduction factor varying from 1 to 9 (step 2), computing the number of resulting states. We then compared Log Likelihood (LL) of original sequence produced by original and reduced HMMs, obtaining results plotted on Fig. 4(b). One can notice that the two curves are very similar, in particular when reduction factor is high. In Table 1, average and maximum loss of likelihood (LLL) are presented for each value of resolution factor, with maximum compression rate: loss of Log Likelihood is fairly low, decreasing when augmenting precision of bisimulation (reduction factor). This kind of analysis is performed to show the graceful evolution of the HMM likelihood when number of states is decreased using bisimulation.

In Fig. 4(a) original number of states vs. reduced number of states are plotted, at varying number of states. More precisely, for a generic value N on abscissa, ordinate represents the number of states obtained after running bisimulation on N-states HMM. It is worth noting that compression rate increases when the number of states grows: this is reasonable, because small structures cannot have a large redundancy.

The second part of this experiment tries to exploit performance of our algorithm regarding classification accuracy. To perform this step we trained two



(a) (b)

Fig. 4. Compression rate (a) and comparison of Likelihood curve for original and reduced HMM (b) on DNA modeling experiment. Reduction factor are 1, 5 and 9.

Table 1. Maximum compression rate, average and maximum Log Likelihood loss for DNA modeling experiment at varying reduction factors.

Reduction factor	Maximum CR (%)	Average LLL (%)	Maximum LLL (%)
1	50.00	9.57	32.20
3	38.16	6.69	20.07
5	33.14	5.45	20.31
7	34.87	5.91	18.90
9	34.04	5.77	22.30

HMMs with 150 states on 200 bases fragments of two different *recA* genes: one was from *glutamicum bacillus* and second was from *tubercolosis bacillus*. Each HMM was then reduced using bisimulation, varying reduction factor from 1 to 9 (step 2). Then, HMMs were retrained with reduced number of states, resulting in 10 reduced HMMs (5 for each sequence). Compression rate varies from 32% for reduction factor 1 to 22% for reduction factor 9 (see Table 2). We tested classification accuracy of HMMs using 300 sequences, obtained by adding synthetic noise to the original two. The noising procedure is the following: each base is changed with fixed probability p (ranging from 0.3 to 0.4), and following determined biological rules (for examples, A becomes T with probability higher than G). Each sequence of this set was evaluated using both models, and classified as belonging to the class whose model showed highest LL. Error rate was then calculated counting misclassified trials and dividing by the total number of trials. Figure 5 shows error rate for original and reduced HMMs, varying the probability of noise. One can notice that error rate trend is quite similar, and that error is very low, always below 5%, proving that HMMs work very well on this type of problems. In Table 2 (a–b), average errors on original and reduced HMMs are presented, respectively, varying noise level and reduction factor value. For the latter, maximum compression rate and maximum LL loss are also pre-

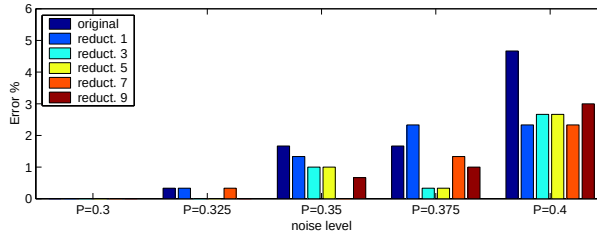


Fig. 5. Error rate for different noise level for DNA modeling experiment.

Table 2. Error on original and reduced HMMs for DNA modeling experiments in function of (a) varying noise level, and (b) varying reduction factor value.

(a)

Noised Level	Error on Original (%)	Error on Reduced (%)
0.3	0.00	0.00
0.325	0.33	0.13
0.35	1.67	0.80
0.375	1.67	1.07
0.4	4.67	2.60

(b)

Reduction Factor	Average CR (%)	Average LLL (%)	Error on Original (%)	Error on Reduced (%)
1	32.00	3.89	1.67	1.27
3	25.33	1.72	1.67	0.80
5	22.00	4.15	1.67	0.80
7	21.33	5.61	1.67	0.80
9	21.66	2.14	1.67	0.93

sented. One can notice that the difference between two errors grows with noise level, i.e., error value becomes higher when noise level increases, and differences can be more significant. Nevertheless, LL losses are very low if compared with compression rate and amount of noise. Actually, classification errors remain below 5%, even on experiments with 40% noise level. Moreover, error level seems to be lower in the reduced case than in the original one. Reasonably, HMMs with less states are able to generalize better, so as recognize also sequences with higher noise, even if we expect a breakdown point, causing a reversing behavior between original and reduced HMMs.

5.2 2D Shape Recognition

Object recognition, shape modeling, and classification are related issues in computer vision. A lot of three-dimensional (3-D) object recognition techniques are based on the analysis of two-dimensional (2-D) aspects (images) and several work can be found in literature on the analysis of 2-D shape or presenting methods devoted to planar object recognition.

A key issue is the kind of image feature used to describe an object, and its representation. Object contours are widely chosen as features, and their representation is basic to the design of shape analysis techniques. Different types of approaches have been proposed in the previous years, like, e.g., Fourier descriptors, chain code, curvature-based techniques, invariants, auto-regressive coefficients, Hough-based transforms, associative memories, and others, each one featured

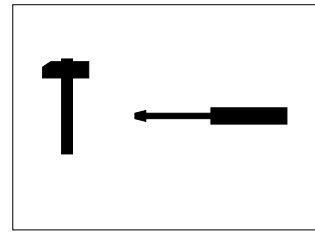


Fig. 6. Toy images for 2D shape recognition using Chain Code.

by different characteristics like robustness to noise and occlusions, invariance to translation, rotation and scale, computational requirements, and accuracy.

In this paper, HMMs are proposed as a tool for shape classification. This preliminary experiment aims at presenting a simple example on the capability of HMM on discriminating object classes, showing its robustness in terms of partial views and, in minor way, of noise. Shape is modeled using chain code, a well-known method to represent contours, which presents some inherent characteristic like the invariance to rotation (if code local differences are considered), and translation.

Although a large literature addresses these issues, the use of HMM for shape analysis has not been widely addressed. To our knowledge, only the work of He and Kundu [9] has been found to have some similarities with our approach. They utilize HMMs to model shape contours represented as auto-regressive (AR) coefficients. Results are quite interesting and presented in function of the number of HMM states ranging from 2 to 6. Moreover, shapes are constrained to be a closed contour.

In our experiment, although limited to a pair of similar objects, the degree of occlusion is quite large, and noise has been included to affect object coding, (without heavily degrading classification performances). Due to lack of space, we will not present results on rotational and scale invariance. Let us only state that scale is not a problem, as the HMM structure can manage it due to the possibility of permanence in the same state. Actually, simple tests on some differently scaled and noised objects have confirmed that HMMS behave correctly in this case. A detailed description of our approach with extensive experiments is not in the scope of this paper, and will be the subject of our future work. In this paper, we would only like to show the capabilities of the HMM to discriminate (also similar) shapes and its stable performances when the minimal structure is obtained by bisimulation with respect to the redundant topology.

In our experiment, given an image of 2D objects, data are gathered assigning at each object its chain code, calculated on object contours. Edges are extracted using *Canny edge detector* [23], while chain code is calculated as described in [24]. Fig. 6 shows the two simple objects, a stylized hammer and a screwdriver, used in the experiment. We train one HMM for each object, varying the number of states from 4 to 20. After applying bisimulation contraction, with reduction factor from 1 to 9, we re-trained HMMs with reduced number of states and compared

Table 3. Maximum compression rate, average and maximum Log Likelihood loss for 2D shape recognition test, at varying reduction factor.

Reduction factor	Maximum CR	Average LLL	Maximum LLL
1	16.74	4.91	72.20
3	9.43	0.55	29.39
5	6.33	2.30	72.26
7	6.40	1.72	72.85
9	4.71	1.28	68.73

them in term of Log Likelihood. Average and maximum Log Likelihood loss are calculated, and results are shown in Table 3, with maximum compression rate for different reduction factor values. Average LLL values are confortantly low: bisimulation does not seem to affect HMM characteristics. Nevertheless, we can also observe that average loss is very low compared with related maximum LLL. This is because compression is not so strong, as evident in Table 3, and therefore some learning session on reduced HMM can produce better results in terms of Log Likelihood. LL of an HMM on a sequence typically grows with N. On the other hand, LL depends on how well the training algorithm worked on the data. Baum-Welch re-estimation ensures to reach the nearest local optimum, without any information about global optimum. So, it is possible that for closed N_1, N_2, with $N_1 < N_2$, a HMM with N_1 states shows larger LL than those with N_2 states, because the training algorithm worked better. To partially solve the problem of convergence, each HMM was trained three times, starting with different random initial conditions. The case of so high LL loss may be explained by a low compression rate (the HMMs have the similar number of states) and very bad training (in this case three trials seems to be insufficient to ensure correct learning).

For testing classification accuracy, we synthetically create two test sets. The first set is obtained considering, for each object, fragments of their chain code of variable length, expressed as percentage rate of the whole length. It varies from 20 to 90 percent, and the point where fragment starts was randomly chosen. The second set is obtained by adding synthetic noise to the two chain codes, using a procedure similar to that used for DNA noising procedure. Each code is changed with fixed probability P, i.e. if cc_i is the original code, with probability P, $(((cc_i - 1) \pm 1) \bmod 8) + 1$ is carried out. Probability ranges from 0.05 to 0.35, and, for each value, 60 sequences are generated. As usual, a sequence is assigned to the class whose model shows the highest Log Likelihood, and error rate is estimated counting misclassified patterns. For each of the two test sets, we calculate performance using original and reduced HMMs and varying reduction factor from 1 to 9. In Table 4, average error for original and reduced HMMs on set of pieces are presented varying reduction factor from 1 to 9. We can see that the difference between two errors is very low.

The same results are presented in Table 5 for a set of noisy sequences, varying reduction factor (Table 5(a)) and noise level (Table 5(b)).

Table 4. Error on original and reduced HMMs for 2D shape recognition experiment (fragments set): (a) varying resolution factor; (b) varying fragment length.

(a)

Reduction factor	Error on Original (%)	Error on Reduced (%)
1	2.52	2.91
3	2.52	2.19
5	2.52	1.51
7	2.52	0.44
9	2.52	2.70

(b)

Fragment Length (%)	Error on Original (%)	Error on Reduced (%)
20 %	4.50	4.33
30 %	3.60	3.28
40 %	2.77	2.32
50 %	3.23	2.31
60 %	3.23	1.75
70 %	2.83	1.36
80 %	0.00	0.23
90 %	0.00	0.01

Table 5. Error on original and reduced HMMs for 2D shape recognition experiment (noised set): (a) varying resolution factor; (b) varying noise level (b).

(a)

Reduction factor	Error on Original (%)	Error on Reduced (%)
1	29.08	24.83
3	29.08	29.05
5	29.08	21.14
7	29.08	28.23
9	29.08	25.97

(b)

Noise level (%)	Error on Original (%)	Error on Reduced (%)
5	11.33	9.64
10	20.5	17.24
15	27.11	23.61
20	31.67	28.21
25	35.24	31.70
30	37.78	34.16
35	39.95	36.33

A consideration can be made on performance of HMMs applied to this problem: average error in recognizing the fragment sequence is 1.21%, a very low value. This means that a simple HMM can be invariant of some type of object occlusions. Nevertheless, noise seems to be a more serious problem, but working on topology and training algorithms classification accuracy may be less affected by this problem.

Another point regards the similarity of the two objects which may seriously affect performances. Using very different objects this problems may be attenuated. More extensive tests on invariance on scale and rotation should be carried out to better evaluate HMM performance for shape classification.

5.3 Comparison with Other Methods

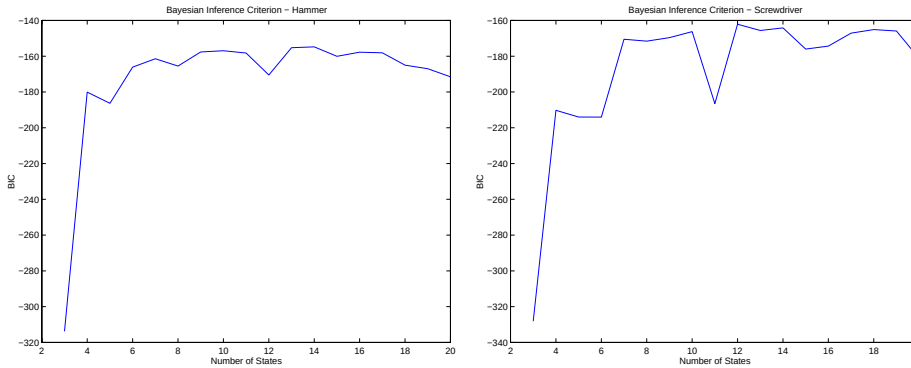
Regarding the model selection approaches present in literature and listed in Section 1, an interesting comparative evaluation is presented in [25]. In that paper, a comparison between MDL/BIC, EBB and MDL for gaussian mixture model is reported, showing comparable performances and proving their superiority with respect to other methods. For convenience, we choose the BIC method [26] for our

comparative analysis. BIC is a likelihood criterion penalized by the model complexity, i.e., in our case, the number of HMM states. Let $X = \{x_i, i = 1, \cdots, N\}$ be the data set we are modeling and $M = \{M_i, i = 1, \cdots, K\}$ be the candidate models. Let us denote as $|M_i|$ the number of parameters of the model M_i, and assuming to maximize the likelihood function $\mathcal{L}(X, M_i)$ for each possible model structure M_i, the BIC criterion is defined as:

$$BIC(M_i) = \log \mathcal{L}(X, M_i) - \frac{1}{2}|M_i| \log(N)$$

This strategy selects the model for which the BIC criterion is maximized.

We compare our strategy with this approach related to the 2D shape experiment. We train 18 HMMs, with states number varying from 3 to 20, and for each model we compute the BIC value. BIC vs number of states curves are plotted in Fig. 7, for the two objects. We then choose the HMM showing the highest



(a) (b)

Fig. 7. BIC value vs number of states curves for the 2D shape recognition experiment with (a) the hammer and (b) the screwdriver.

BIC value (corresponding to 12 and 14 states, respectively for screwdriver and hammer).

With our bisimulation approach we train one HMM with 20 states, apply bisimulation and train another HMM with calculated number of states, varying reduction factor fro 1 to 9 (step 2). To compare the two methods we create a test set by adding synthetic noise (of various entity) to the two chain codes, in a way similar to that presented in the previous section, obtaining, for each noise level, 120 sequences to be classified. We then calculate the classification error applying the two approaches, presenting results in Table 6, in function of variable noise level. We can notice that, on the average, classification accuracy is quite similar: in fact BIC method needs 18 training session, while our method only two, plus the time for determining bisimulation contraction (that is $O(MKN \log N)$, given an HMM with N states, K edges and M symbols). In problems with a short

Table 6. Comparison between BIC method and our approach.

Method	States		Classification Error						
	Screw.	Hammer	Noise 0.05	Noise 0.10	Noise 0.15	Noise 0.20	Noise 0.25	Noise 0.30	Noise 0.35
BIC	12	14	13.33	25.00	32.50	36.88	39.83	41.81	42.98
Bisim RF 1	14	15	20.00	30.00	33.89	36.25	42.67	44.44	48.57
Bisim RF 3	15	16	21.67	34.17	39.44	42.08	43.67	44.72	45.48
Bisim RF 5	18	18	10.00	10.83	22.78	29.17	34.67	40.28	35.71
Bisim RF 7	17	19	28.33	38.33	42.22	44.17	45.33	46.11	46.67
Bisim RF 9	20	20	31.67	37.50	37.22	37.08	37.00	34.17	30.24

alphabet (as DNA modeling and chain code problems), our method is definitively faster than BIC, giving approximately the same classification accuracy.

6 Conclusions

In this paper, probabilistic bisimulation is used to estimate the minimal structure of a HMM. It has been shown that starting from a redundant configuration, bisimulation allows to merge equivalent states while preserving classification performances. Redundant and minimal HMM architectures have been tested on two different cases, DNA modeling and 2D shape classification, showing the usefulness of the approach. Moreover, our method has been compared with a classic model selection scheme, showing comparative performances but with a less computational complexity.

References

1. Rabiner, L.R.: A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. of IEEE **77(2)** (1989) 257–286.
2. Baum, L.E., Petrie, T.E, Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Annals of Math. Statistics. **41(1)** (1970) 164–171.
3. Baum, L.E.: An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequality **3** (1970) 1–8.
4. Veltman, S.R., Prasad, R.: Hidden Markov Models applied to on-line handwritten isolated character recognition. IEEE Trans. Image Proc., **3(3)** (1994) 314–318.
5. Churchill, G.A.: Hidden Markov Chains and the analysis of genome structure. Computers & Chemistry **16** (1992) 107–115.
6. Eickeler, S., Kosmala, A., Rigoll, G.: Hidden Markov Model based online gesture recognition. Proc. Int. Conf. on Pattern Recognition (ICPR) (1998) 1755–1757.
7. Jebara, T., Pentland, A.: Action Reaction Learning: Automatic Visual Analysis and Synthesis of interactive behavior. In 1st Intl. Conf. on Computer Vision Systems (ICVS'99) (1999).
8. Theodoridis, S., Koutroumbas, K.: Pattern recognition. Academic Press (1999).
9. He, Y., Kundu, A.: 2-D shape classification using Hidden Markov Model. IEEE Trans. Pattern Analysis Machine Intelligence. **13(11)** (1991) 1172–1184.

10. Figueiredo, M.A.T., Leitao, J.M.N., Jain, A.K.: On Fitting Mixture Models, in E. Hancock and M. Pellilo(Editors), Energy Minimization Methods in Computer Vision and Pattern Recognition, 54–69, Springer Verlag, 1999.
11. Stolcke, A., Omohundro, S.:Hidden Markov Model Induction by Bayesian Model Merging. Hanson, S.J., Cowan, J.D., Giles, C.L. eds. Advances in Neural Information Processing Systems **5** (1993) 11–18.
12. Brand, M.: An entropic estimator for structure discovery. Kearns, M.S., Solla, S.A., Cohn, D.A. eds. Advances in Neural Information Processing Systems **11** (1999).
13. Aczel, P.: Non-well-founded sets. Lecture Notes, Center for the Study of Language and Information **14** (1988).
14. Baier, C., Engelen, B., Majster-Cederbaum, M.: Deciding Bisimilarity and Similarity for Probabilistic Processes. J. Comp. and System Sciences **60** (1999) 187–231.
15. Dovier, A., Piazza, C., Policriti, A.: A Fast Bisimulation Algorithm. In proc. of *13th Conference on Computer Aided Verification, CAV'01*, 2001. Paris, France.
16. Kanellakis, P.C., Smolka, S.A.: CCS Expressions, Finite State Processes, and Three Problems of Equivalence. Information and Computation, **86(1)** (1990) 43–68.
17. Lisitsa, A., Sazanov, V.: Bounded Hyperset Theory and Web-like Data Bases. In 5th Kurt Gödel Colloquium. LNCS**1289** (1997) 172–185.
18. Milner, R.: Operational and Algebraic Semantics of Concurrent Processes. In J. van Leeuwen, editor, Handbook of Theoretical Computer Science (1990).
19. Paige, R., Tarjan, R. E.: Three Partition refinement algorithms. SIAM Journal on Computing **16(6)** (1987) 973–989.
20. Van Benthem, J.: Modal Correspondence Theory. PhD dissertation, Universiteit van Amsterdam, Instituut voor Logica en Grondslagenonderzoek van Exacte Wetenschappen, (1978) 1-148.
21. Salzberg, S.L.: Gene discovery in DNA sequences. IEEE Intelligent Systems **14(6)** (1999) 44–48.
22. Salzberg, S.L., Searls, D., Kasif, S.: Computational methods in Molecular Biology. Elsevier Science (1998).
23. Canny, J.F.: A computational approach to edge detection. IEEE Trans. Pattern Analysis Machine Intelligence **8(6)** (1986) 679–698.
24. Jain, R., Kasturi, R., Schunck, B.G.: Machine Vision. McGraw-Hill (1995).
25. Roberts, S., Husmeier, D., Rezek, I., Penny, W.: Bayesian Approaches to gaussian mixture modelling, IEEE Trans. on P.A.M.I., **20(11)** (1998) 1133–1142.
26. Schwarz, G.: Estimating the dimension of a model, The Annals of Statistics, **6(2)** (1978) 461–464.