Hidden Markov Models for pattern extraction

Lyonel Serradura, Thierry Brouard, Mohamed Slimane, Nicole Vincent
Laboratoire d'Informatique
Ecole d'Ingénieurs en Informatique pour l'Industrie
Université de Tours
64, avenue Jean Portalis
37200 Tours - FRANCE
Tel: +33 (0)2 47 36 14 14 - Fax: +33 (0)2 47 36 14 22
e-mail: {serradura, brouard, slimane, vincent}@univ-tours.fr

Abstract

This paper focuses on using Hidden Markov Models for 2D pattern extraction in grayscale pictures. The main question is: can we solve pattern extraction problems thanks to HMM ? Thanks to GHOSP algorithm, HMMs give models for the patterns. The two-step algorithm presented here, first looks for possible positions with a line by line seek algorithm, using comparisons between the probabilities of the best state sequences that have been computed with Viterbi algorithm. Then the second stage, based upon state durations, verifies the positions found, in order to increase the quality of the extraction. This algorithm was tested on both artificial and real pictures.

Introduction

Finding a method to extract patterns from images is an actual problem. Many efforts are under way because there are high stakes in many domains. Methods are numerous, and as far as we are concerned, we meant to study what Hidden Markov Models can bring to the solution.

Hidden Markov Models (HMM) are widely used as modeling tools in pattern recognition [10] [12] [6], especially for speech recognition [10] [8] [7] and image processing [3] [2] [5]. Here we tackle the problem of 2D pattern extraction in grayscale pictures.

We use the unsupervised learning algorithm GHOSP [11] [5] to obtain HMM modeling the pattern to match. The Forward algorithm can be used for pattern extraction. However, we have been looking for possible use of Viterbi algorithm. This algorithm computes the state sequence most probably followed by the HMM to generate the observation. So this is the algorithm we adapted for pattern extraction.

We will begin with a short explanation of fundamental characters of the problem and will present the tools we have. Then we will detail the originality of our approach and the steps of the process. We will end presenting the results.

1 Hidden Markov Models

Our main tools are Hidden Markov Models. Here are some brief explanations about HMM and basic algorithms to work with.

1.1 Definition

A Hidden Markov Model is a model for a N state stochastic process. Each of these N states $E = \{e_i, i \in [1..N]\}$ may generate any one of the M symbols $S = \{s_j, j \in [1..M]\}$. Any observation $\mathcal{O} = (o_1, o_2, \ldots, o_T)$ is a sequence of T symbols $o_t \in S$ $(\forall t \in [1..T])$ encountered during T successive instants.

The Hidden Markov Model is defined as $\lambda = (A, B, \Pi)$:

1. the stochastic matrix $A_{N,N}$ of transitions probabilities
2. the stochastic matrix $B_{N,M}$ of symbols generation probabilities
3. the stochastic vector Π (length N) of initial probabilities

The observation $\mathcal{O}$ is generated by the HMM following a certain state sequence. As this state sequence is unknown, we talk about *Hidden* Markov Model.

We are working with 256 grey levels images, our observations are sequences of grey levels from pixels, sorted line by line. Thus, our symbols are associated with grey levels. No preprocessing of the image is necessary.

1.2 Fundamental problems

Three fundamental problems need to be solved in order to use HMM efficiently.

1.2.1 Evaluation

Given an observation $\mathcal{O} = (o_1, o_2, \ldots, o_T)$ and a HMM λ, what is the probability $P(\mathcal{O}|\lambda)$ for the HMM to generate the observation.

This problem is solved in polynomial time with Forward and Backward algorithms. Their time complexity is $\mathcal{O}(N^2T)$.

1.2.2 Most possible state path

Given an observation $\mathcal{O} = (o_1, o_2, \ldots, o_T)$ and a H-MM λ, what is the most possible hidden state sequence $\mathcal{Q}^* = (q_1^*, q_2^*, \ldots, q_T^*)$ followed by the HMM to generate the observation, thus maximizing $P(\mathcal{Q}^*|\mathcal{O}, \lambda)$.

Viterbi algorithm [13] recursively compute this most possible state sequence $\mathcal{Q}^*$, also known as the best state path.

1.2.3 Optimum HMM construction

It is a learning problem. Given an observation $\mathcal{O} = (o_1, o_2, \ldots, o_T)$, what is the HMM λ which can generate this observation with the highest probability $P(\mathcal{O}|\lambda)$.

There are many solutions, and the Baum-Welch algorithm [1] is the one of them we choose. Basically this algorithm is the EM algorithm (*Expectation Maximization*), iteratively improving an initial model. So it is a gradient algorithm, and the result is a *local* optimum solution. In previous studies, we demonstrated how the association of Baum-Welch algorithm and a genetic algorithm improves learning [11] [5].

1.3 Search of the number of hidden states

The GHOSP algorithm (*Genetic Hybrid Optimization and Search of Parameters*) aims at solving the problem of the search of the best suited number of hidden states for a HMM. GHOSP is an hybridization of a genetic algorithm (GA) with the Baum-Welch algorithm [11] [5]. This algorithm solves the problem of unsupervised HMM learning: the number of hidden states is computed along with the parameters of the HMM (*i.e.* the matrix coefficients). Figure 1 is a schematic view of GHOSP algorithm.

GHOSP works on a HMM population divided in sub-populations, each defined by the number of hidden states of its HMM. The numbers of hidden states allowed are included between a minimum and a maximum limits the user gives before running the algorithm. For each iteration, the population comprises two classes, *parents* and *children*. The *children* are optimized with the Baum-Welch algorithm and

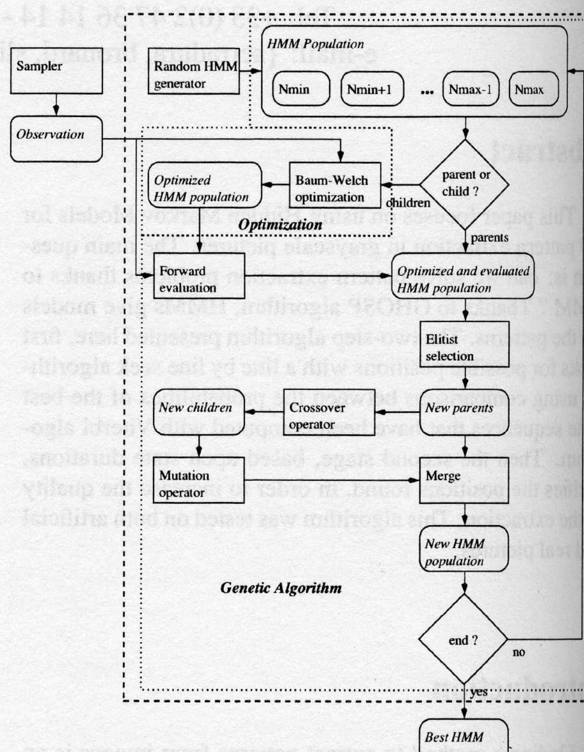

Figure 1: Scematic view of GHOSP algorithm

then evaluated with the Forward algorithm. The two classes are then merged and sorted according to their probability of generating the observation. A certain ratio (parameter chosen by user before running the algorithm) of HMMs is kept, the other part is discarded. The remaining HMMs are the new *parents* and they are crossed to produce new *children*. The last ones are eventually mutated. We developed specific crossing operators to cross and generate HMM with different numbers of hidden states.

1.4 Viterbi algorithm

The method presented here uses results from Viterbi algorithm. This algorithm computes the best state sequence, *i.e.* the state path most probably followed by the HMM to generate the observation.

1.4.1 How it works

Given a HMM λ and an observation $\mathcal{O}$, Viterbi algorithm computes the state sequence $\mathcal{Q}^*$ such as $P(\mathcal{Q}^*|\lambda, \mathcal{O})$ is maximum.

For each partial observation $\mathcal{O}_t = (o_1, o_2, \cdots, o_t)$ (length $t < T$) from observation $\mathcal{O} = (o_1, o_2, \cdots, o_T)$, the algorithm computes the probability $\delta_t(e_i) = \max_{\mathcal{O}_{t-1}} \{P(\mathcal{Q}_t, q_t = e_i|\mathcal{O}_t\lambda)\}$ of the best partial state path $\mathcal{Q}_t$ leading to state $q_t = e_i$ guided with partial observation $\mathcal{O}_t$.

The sequence giving the best state path to e_i in t steps is kept in the array $\Psi = [\psi_t(e_i)]_{\forall t \in [1..T], \forall i \in [1..N]}$: $\psi_t(e_i)$ is the state e_i such as $q_t = e_i$ and $P(\mathcal{Q}_t|\lambda, \mathcal{O}_t)$ is maximum among the partial state paths of length t leading to state $q_t = e_i$.

1.4.2 Results from Viterbi algorithm

Thanks to this computation process, we obtain two important results:

1. the best state path $\mathcal{Q}^*$ recursively defined by:

$$\begin{cases} q_T^* = \arg\left(\max_{i \in [1..N]}\{\delta_T(i)\}\right) \\ q_t^* = \psi_{t+1}(q_{t+1}^*) \qquad \forall t \in [1..T-1] \end{cases}$$

2. the probability P^* of the best state path $\mathcal{Q}^*$:

$$P^* = P(\mathcal{Q}^*|\lambda, \mathcal{O}) = \max_{i \in [1..N]} \{\delta_T(i)\}$$

1.4.3 Probabilities of sub-paths

The different calculus needed to obtain the best state path $\mathcal{Q}^*$ and its probability P^*, allow to know the probability of each partial path $\mathcal{Q}_t^*$:

$$P(\mathcal{Q}_t^*|\lambda, \mathcal{O}) = \delta_t(q_t^*) \qquad \forall t \in [1..T]$$

We can then deduce the probability of each sub-path of the best state path. The partial paths begin at the first state of the whole state path, while sub-path may begin anywhere. From the definition of δs, there is a relation between $P(\mathcal{Q}^*(t_1)|\lambda, \mathcal{O})$, $P(\mathcal{Q}^*(t_2)|\lambda, \mathcal{O})$ and $P(\mathcal{Q}^*(t_1, t_2)|\lambda, \mathcal{O})$. The probability of sub-path $\mathcal{Q}(t_1, t_2)$ is then $\frac{\delta_{t_2}(q_{t_2}^*)}{\delta_{t_1}(q_{t_1}^*)} \times \pi_{q_{t_1}^*}.b_{q_{t_1}^*}(o_{t_1})$.

However, this theorical result is almost always useless. Actually, if any $\delta_t(q_t^*)$ is 0, the relation above is either 0 or undefined. So we will use $P(\mathcal{O}|\lambda, \mathcal{Q}^*(t_1, t_2))$ expression.

2 Pattern extraction

The goal of extraction is to point out every possible position of the object in the image.

To reduce computing time, we propose a two-step method. We first study the image line by line to select points, with a certain confidence rate, that might be part of the object. Then we verify that the object is actually present.

Recognition with state path relies on the following assumption: if the best state sequence from the unknown observation doesn't have a probability at least as high as the one of the learned pattern, this unknown observation is very unlikely to be the pattern to match.

2.1 Observations

The learned object is a 256 grey level image, with $L_\mathcal{V} \times H_\mathcal{V}$ pixels. We call $\mathcal{V}$ the observation that comes from this image. It is a unique observation vector we give to GHOSP. Finally we get a HMM, λ, to model this image.

The image where we seek the pattern is also a 256 grey level image, but with $L_\mathcal{O} \times H_\mathcal{O}$ pixels. We divide it into lines, the $H_\mathcal{O}$ of them are transformed into an observation called $\mathcal{O}$. It is composed of the $H_\mathcal{O}$ observation vectors $\mathcal{O}_k$ ($k \in [1..H_\mathcal{O}]$, length $L_\mathcal{O}$). This sampling of the image enables to use M symbols corresponding to grey levels.

2.2 State paths

We call $\mathcal{Q}_\mathcal{V}$ the best state sequence followed by the HMM λ to generate the observation $\mathcal{V}$. We also call $\mathcal{Q}_k$ the best state sequences for each of the $H_\mathcal{O}$ observation vectors $\mathcal{O}_k$.

2.3 Sub-paths study

The first step relies on partial paths and sub-paths study. We do it line by line: we compare the probabilities of each partial path of each $L_\mathcal{V}$ long sub-path of $\mathcal{Q}_k$ with the probabilities of the partial paths of the $L_\mathcal{V}$ long partial path $\mathcal{Q}_{L_\mathcal{V}}$ of $\mathcal{Q}_\mathcal{V}$.

The unknown image is scrutated line by line thanks to the sampling used. Then the pixel by pixel scrutation is a consequence of the evolution of the starting point i ($i \in [1..L_{\mathcal{O}}]$) of the sub-path $L_{\mathcal{V}}$. Moreover, when i is greater than $L_{\mathcal{O}} - L_{\mathcal{V}}$ the sub-path studied begin at i and has a length of $L_{\mathcal{O}} - i$.

As a result, each and every pixel is evaluated as a start position of the pattern to match.

Probabilities of partial paths of the sub-paths of $\mathcal{Q}_{\mathcal{V}}$ directly come from the computing of $\delta_t(q_i^*)$, the δ_ts and the q_i^* sequence used in Viterbi algorithm.

Probabilities of sub-paths are computed as $P(\mathcal{O}_k(t_1,t_2)|\mathcal{Q}_k,\lambda)$. This also gives the probabilities associated with each partial path of sub-path $\mathcal{Q}_k(t_1,t_2)$.

Probabilities of partial paths are compared rank by rank. Scoring is achieved incrementing each time the probabilty in the unknown image is greater than or equal to the one of the learned image. Then the score is normalized between 0 and 1 according to the width $L_{\mathcal{V}}$.

This score gives an (incremental) response better suited than the whole path probability itself.

Thus, this line by line seeking algorithm computes for each position the probability that it is the beginning of the first line of the seeked object.

2.4 Selection and verification

The second step selects the interesting points amid those obtained from the line by line algorithm. This verification step uses state durations [9]. This step creates an observation the same size of the learned image at each position with a non zero score from the previous step. Then the state durations of this new observation are computed and compared to those of the pattern to match.

State durations are pair sequences, the first indicates each state transition, the second states how long the HMM staies in the corresponding state.

State durations comparison takes into account possible noise: while the learned image stay in a state, the unknown image is mostly (*i.e.* more than a given time) in the same state.

Moreover, relative state durations are rather independent of the size of the object in the unknown image, at least provided that proportions are the same. The durations are kept relative to the size of the state path, not absolute. This way, we can stay in the unknown image for the sufficient time and then deduce the height from its width.

At the end, our method gives for each pixel of the unknown image the probability that it is the first pixel (usually the upper left one) of an occurrence of the pattern to match.

3 Results

Here we present results coming from two kinds of images. The first image is constructed from patterns, one of them is the one we are looking for. The second image is a picture from which the pattern to match is a part.

To point out the two steps of the method proposed here, the results are given as two 256 grey level images. The first one presents the intermediate results (*i.e.* the probabilities from the line by line study), the second one the final result.

In these figures, each grey level is a representation of a probability between 0 (level 0, black) and 1 (level 255, white). This way, the brighter the pixel, the greater the likelihood.

3.1 Constructed image

Our first test is to find out a specific face in a picture composed of various faces. Figure 2 is a collection of thumbnailed pictures of faces. The one we are looking for is actually present.

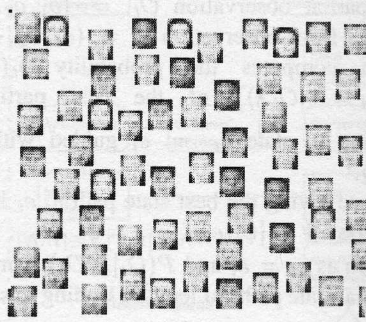

Figure 2: Image constructed from patterns. The pattern to match is one of them

Figure 3 shows the face we are looking for, and its positions in the collection.

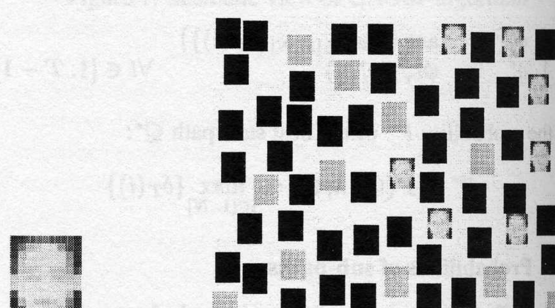

Figure 3: Thumbnail of the face to seek, and its positions in the constructed image

Figure 4 shows a second view of the same face and where it is in the collection. The point of view is slightly different in this thumbnail.

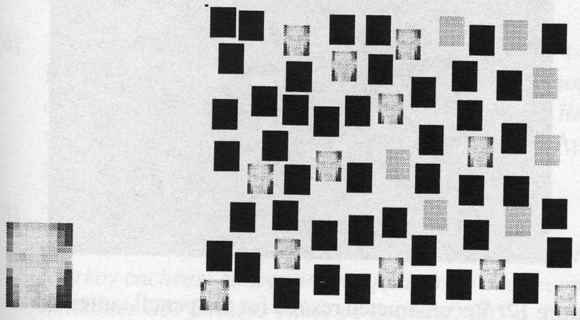

Figure 4: Thumbnail of an other view of the same face, and its positions in the constructed image

GHOSP algorithm gives a HMM to model the first thumbnail. Then, we use the method proposed here to extract this thumbnail from the collection. Figure 5, figure 6, and figure 7 shows the results.

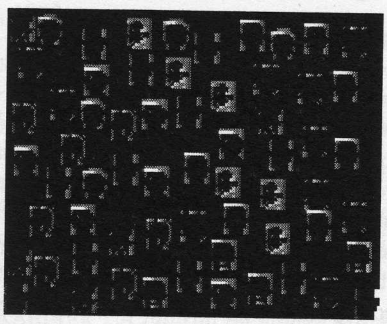

Figure 5: Results of the first stage of the extraction for the first thumbnail

At first glance, we can see that the first step (figure 5) already found out every position of the seeked thumbnail with the highest probability of 1. The second thumbnail is also greatly considered, with a probability of 0.77. The final result (figure 6) is coherent, the seeked faces are found out everywhere they were with the maximum probability. On the other hand, the second view of the same face is still the second best result, but only with a probability of 0.30: it is the same face, but actually it is not the same thumbnail. The image of figure 7 was reconstructed to focus on these results.

3.2 Real picture

The unknown image is now the photography in figure 8 (320×240 pixels).

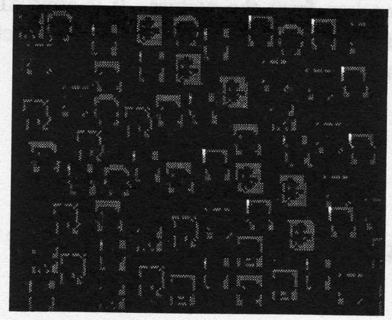

Figure 6: Results of the second stage of the extraction for the first thumbnail

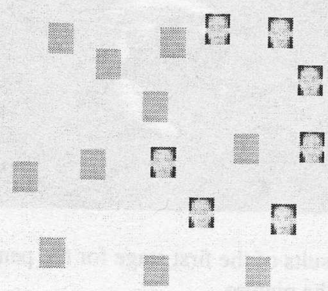

Figure 7: Reconstructed results of the extraction of the first thumbnail

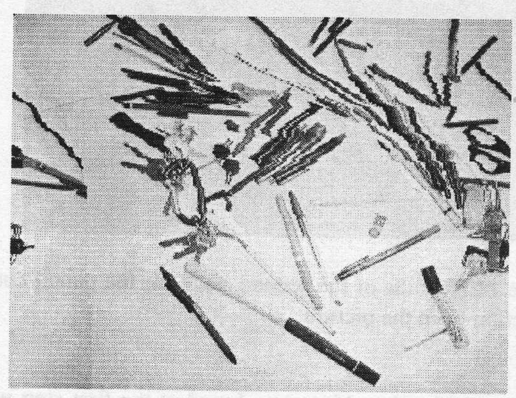

Figure 8: Picture where the extraction will take place

The seeked objet is a pencil cutter and its image (13 × 17 pixels) is taken from the picture. It is approximatively centered in the picture. Figure 9 shows the object learned and then extracted from the picture.

Figure 9: Object learned, to be extracted

Figure 10, figure 11, and figure 12 present the results.

Figure 10: Results of the first stage for the pencil cutter extraction from the picture

Figure 11: Results of the second stage for the pencil cutter extraction from the picture

Once again, the object was found at the first step with a probability of 1 (figure 10). However we can see that the line by line seeking algorithm may be greatly confused if the background color is the same as the first pixel of the image of the object in the learning process (this does not occur in the first example). This highlights the importance of the second step (figure 11): the pattern to match is found out, and it is the only one with a probability of 1. The image of figure 12 was reconstructed to point out where the extraction algorithm found the object.

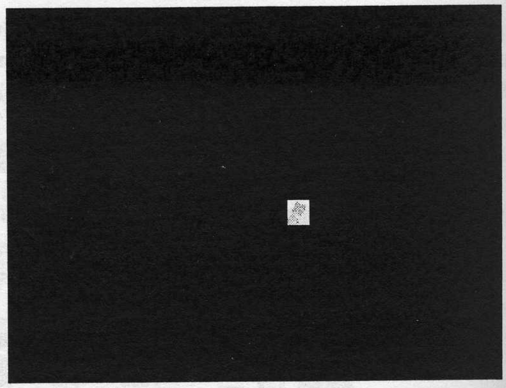

Figure 12: Reconstructed results for the pencil cutter extraction from the picture

Conclusion

The extraction algorithm we have presented here uses the probabilities of the best state paths computed by Viterbi algorithm to perform its task. This method appeared to be simple and efficient. It is especially very efficient on constructed images, but it also works rather fine on real pictures. However, the first step of the method may be easily confused if background color and the color of the first pixels of the seeked object are the same. The use of state durations is a good answer to this problem. And it provides some kind of protection against noise and scale change.

We can now answer our main question: we can actually use HMM and Viterbi algorithm for pattern extraction.

References

[1] L.E. Baum, J.A. Eagon : *An inequality with applications to statistical estimation for probabilistic functions, of Markov processes and to a model for ecology,* Bulletin American Society, vol. 73, pp. 360-363, 1967.

[2] T. Brouard, M. Slimane, G. Venturini, J.-P. Asselin de Beauville : *Segmentation non-supervisée d'images par chaînes de Markov cachées*, Cinquièmes Rencontres de la Société Francophone de Classification, pp. 177-180, Lyon, 1997.

[3] T. Brouard, M. Slimane, G. Venturini, J.-P. Asselin de Beauville : *Apprentissage du nombre*

d'états cachés d'une chaîne de Markov cachée pour la reconnaissance d'images, 16ème Colloque GRETSI sur le Traitement du Signal et des Images, pp. 845-848, Grenoble 15-19 septembre 1997.

[4] T. BROUARD, M. SLIMANE, J.-P. ASSELIN DE BEAUVILLE, G. VENTURINI : *Apprentissage d'une chaîne de Markov cachée - problèmes numériques liés à l'application à l'image*, Revue de Statistique Appliquée, vol. XLVI no 2, pp. 83-108, 1998.

[5] T. BROUARD, *Algorithme hybride d'apprentissage de chaînes de Markov cachées: conception et application à la reconnaissance des formes*, Thèse de doctorat, 205 p, Laboratoire d'Informatique, Ecole d'Ingénieurs en Informatique pour l'Industrie, Tours, 1999.

[6] M. GILLOUX : *Real-time handwritten word recognition within large lexicons*, Proceedings of International Workshop on Frontiers in Handwritting Recognition, pp 301-304, Colchester (United Kingdom),

[7] X. HUANG, Y. AKIRI, M. JACK : *Hidden Markov models for speech recognition*, Edimburgh University Press, 1990.

[8] A. KRIOUILLE : *La reconnaissance automatique de la parole et les modèles Markoviens cachés*, Thèse de Doctorat, 149 p, Université de Nancy I, 1990.

[9] M. A. MAHJOUB : *Amélioration du pouvoir discriminant des modèles de Markov cachés par modélisation de la durée d'état*, 17ème Colloque GRETSI, pp. 443-446, Vannes 13-17 septembre 1999.

[10] L.R. RABINER : *A tutorial on hidden Markov models and selected application in speech recognition*, Proceedings of IEEE, vol. 77, pp. 257-286, 1989.

[11] M. SLIMANE, G. VENTURINI, J.-P. ASSELIN DE BEAUVILLE, T. BROUARD, A. BRANDEAU : *Optimizing hidden Markov models with a genetic algorithm*, Lectures notes in computer science no 1063, pp. 384-396, Ed. Springer Verlag, septembre 1995.

[12] M. SLIMANE, G. VENTURINI, J.-P. ASSELIN DE BEAUVILLE , T. BROUARD: *Hybrid Genetic Learning of Hidden Markov Models for Time Series Prediction*, "Biomimetic approaches in management science", K-LUWER Academics Eds, à paraitre, 14 p.

[13] A.J. VITERBI : *Error bounds for convolutional codes and asymptotically optimum decoding algorithm*, IEEE transactions on Information Theory, vol. 13, pp.260-269, 1967.