ITERATIVE DECODING OF TWO-DIMENSIONAL HIDDEN MARKOV MODELS

Florent Perronnin[*]*, Jean-Luc Dugelay*

Institut Eurecom
Multimedia Communications Department
BP 193, F-06904 Sophia Antipolis Cedex
{perronni, dugelay}@eurecom.fr

Ken Rose[†]

Department of Electrical and Computer Engineering
University of California
Santa Barbara, CA 93106-9560
rose@ece.ucsb.edu

ABSTRACT

While the hidden Markov model (HMM) has been extensively applied to one-dimensional problems, the complexity of its extension to two-dimensions grows exponentially with the data size and is intractable in most cases of interest. In this paper, we introduce an efficient algorithm for approximate decoding of 2-D HMMs, i.e., searching for the most likely state sequence. The basic idea is to approximate a 2-D HMM with a Turbo-HMM (T-HMM), which consists of horizontal and vertical 1-D HMMs that "communicate", and allow iterated decoding (ID) of rows and columns by a modified version of the forward-backward algorithm. We derive the approach and its re-estimation equations. We then compare its performance to another algorithm designed for decoding 2-D HMMs: the Path Constrained Variable State Viterbi (PCVSV) algorithm [1]. Finally, we combine our approach with PCVSV and show that the combination outperforms each algorithm taken separately.

1. INTRODUCTION

One dimensional HMMs have a long history of success in various problem domains, perhaps most notably in speech recognition. Their success is largely due to the development of computationally efficient algorithms, namely, dynamic programming (Viterbi) and Baum-Welch (see [2] for a tutorial). However, direct extension of these techniques to 2-D HMMs suffers from exponential growth in complexity (with data size) [3] and is hence intractable in most applications of practical value.

Many approaches to solve the 2-D problem consist of approximating the 2-D HMM with one or many 1-D HMMs. Perhaps the simplest approach is to trace a 1-D scan that takes into account as much of the neighborhood relationship (or 2-D structure) of the data as possible, e.g., the Hilbert-Peano scan [4]. Another approach is the so-called pseudo 2-D HMM [3]. The assumption is that there exists a set of "super" states which are Markovian and which subsume a set of simple Markovian states. A recent example of such an algorithm, the PCVSV algorithm [1], is particularly relevant to this paper and will be briefly described in section 6.

The approach we pursue here is to first convert a 2-D HMM into a Turbo-HMM (T-HMM): a set of inter-connected horizontal and vertical 1-D HMMs that "communicate" through inducing prior probabilities on each other. A modified version of the forward-backward algorithm, is performed successively on rows and columns and the process is iterated until convergence. Similar approaches have been proposed in the image processing community, mainly in the context of image restoration [5] or page layout analysis [6]. The term "turbo" was also used in [6] in reference to the now celebrated turbo error-correcting codes. However, in [6] the layout of the document is pre-formulated with two orthogonal grammars and the problem is clearly separated into horizontal and vertical components in distinction with the more challenging case of general 2-D HMMs.

The next section specifies how to approximate a first order 2-D HMM with a T-HMM. In section 3, we derive the re-estimation equations. In section 4 we discuss potential convergence problems. Section 5 elaborates on the horizontal and vertical separability assumption. Section 6 briefly describes the PCVSV algorithm. We then proceed to combine it with our approach so as to benefit from their complementary nature. Experimental results are presented in section 7 for performance evaluation.

2. APPROXIMATION OF THE LIKELIHOOD FUNCTION

We assume in the following that the reader is familiar with 1-D HMMs (see e.g., [2]). Let $O = \{o_{i,j}, i = 1, \ldots, I, j = 1, \ldots, J\}$ be the set of all observations. For convenience we also introduce the notations $o_i^{\mathcal{H}}$ and $o_j^{\mathcal{V}}$ for the i-th row and j-th column of observations, respectively. Similarly, $Q = \{q_{i,j}, i = 1, \ldots, I, j = 1, \ldots, J\}$ denotes the set of all states, while $q_i^{\mathcal{H}}$ and $q_j^{\mathcal{V}}$ denote the i-th row and j-th column of states. Finally, let λ be the set of all model parameters, and let $\lambda_i^{\mathcal{H}}$ and $\lambda_j^{\mathcal{V}}$ be the respective rows and columns of parameters. We first approximate the joint likelihood of O and Q given λ.

$$\begin{aligned} P(O,Q|\lambda) &= P(O|Q,\lambda)P(Q|\lambda) \\ &= \prod_{i,j} P(o_{i,j}|q_{i,j},\lambda)P(q_{i,j}|q_{i,j-1},q_{i-1,j},\lambda). \end{aligned}$$

Note that the conditional probability $P(q_{i,j}|q_{i,j-1}, q_{i-1,j}, \lambda)$ reduces to $P(q_{1,j}|q_{1,j-1}, \lambda)$ if $i = 1$, to $P(q_{i,1}|q_{i-1,1}, \lambda)$ if $j = 1$ and to $P(q_{1,1}|\lambda)$ if $i = j = 1$. We will assume from now on that $P(q_{i,j}|q_{i,j-1}, q_{i-1,j}, \lambda)$ is *separable*, i.e. that it can be decomposed into the product of horizontal and vertical components. This approximation will allow us to run the forward-backward algorithm on rows and columns. Hence:

$$\begin{aligned} P(q_{i,j}|q_{i,j-1},q_{i-1,j},\lambda) &= \nu(q_{i,j-1},q_{i-1,j}) f^{\mathcal{H}}(q_{i,j},q_{i,j-1}) \\ &\quad f^{\mathcal{V}}(q_{i,j},q_{i-1,j}) \end{aligned}$$

[*] The research activities of F. Perronnin were funded by France Telecom

[†] This work is supported in part by the NSF under grants EIA-9986057 and EIA-0080134, the University of California MICRO program, Conexant Systems, Inc., Dolby Laboratories, Inc., Lucent Technologies, Inc., Mindspeed Technologies, and Qualcomm, Inc.

where $\nu(q_{i,j-1}, q_{i-1,j})$ is a normalization factor: $\nu(q_{i,j-1}, q_{i-1,j}) = 1/(\sum_{q_{i,j}} f^{\mathcal{H}}(q_{i,j}, q_{i,j-1}) f^{\mathcal{V}}(q_{i,j}, q_{i-1,j}))$.

In effect, the factors $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$ and $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$ approximate $P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}})$ and $P(q_{i,j}|q_{i-1,j}, \lambda_i^{\mathcal{V}})$, respectively. This approximation will be justified in section 5. Hereafter, we will replace the notations $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$ and $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$ with the more intuitive notations $P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}})$ and $P(q_{i,j}|q_{i-1,j}, \lambda_j^{\mathcal{V}})$. Moreover, we propose an additional simplifying assumption. To avoid the complexity due to terms that depend on states that are both on different rows and columns we assume that $\nu(q_{i,j-1}, q_{i-1,j})$ is approximately constant (i.e. does not depend on $q_{i,j-1}$ and $q_{i-1,j}$).

$$\begin{aligned} & P(O, Q|\lambda) \\ \approx & \prod_{i,j} P(o_{i,j}|q_{i,j}, \lambda) P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}}) P(q_{i,j}|q_{i-1,j}, \lambda_j^{\mathcal{V}}) \\ \approx & \prod_{j} P(o_j^{\mathcal{V}}|q_j^{\mathcal{V}}, \lambda_j^{\mathcal{V}}) P(q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_{i} P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}}) \\ \approx & \prod_{j} P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_{i} P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}}). \end{aligned}$$

We approximate the conditional probability $P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}})$ by modifying the condition:

$$P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}}) \approx P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}),$$

which can be justified as follows:

$$P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) = \sum_{q_{i,j-1}} P(q_{i,j-1}, q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) =$$

$$\sum_{q_{i,j-1}} \frac{\alpha_{i,j-1}^{\mathcal{H}}(q_{i,j-1}) P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}}) P(o_{i,j}|q_{i,j}, \lambda) \beta_{i,j}^{\mathcal{H}}(q_{i,j})}{P(o_i^{\mathcal{H}}|\lambda_i^{\mathcal{H}})},$$

where α and β are the forward and backward variables used in the forward-backward algorithm (see Table 1 for the list of notations). So $P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}})$ can be seen as a weighted average of the $P(q_{i,j}|q_{i,j-1}, \lambda_i^{\mathcal{H}})$'s. The weights depend on the probability of the paths passing through $q_{i,j-1}$ and $q_{i,j}$. Now we have:

$$P(O, Q|\lambda) \approx \prod_j \left[P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_i P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) \right],$$

where each term $P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}})$ corresponds to a 1-D vertical HMM. Note that $\prod_i P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}})$ is in effect a horizontal prior for column j and, hence, horizontal and vertical decoding "communicate". We assume that the quantity $P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}})$ is known, i.e., that it was obtained during the previous horizontal step.

3. THE MODIFIED FORWARD-BACKWARD ITERATIONS

If we sum over all possible paths, we obtain the marginal:

$$\begin{aligned} P(O|\lambda) &= \sum_{Q} P(O, Q|\lambda) \\ &\approx \sum_{q_1^{\mathcal{V}} \ldots q_J^{\mathcal{V}}} \prod_j \left[P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_i P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) \right] \\ &\approx \prod_j \sum_{q_j^{\mathcal{V}}} \left[P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_i P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) \right] \end{aligned}$$

$b_{q_{i,j}}(o_{i,j})$	$P(o_{i,j}\|q_{i,j}, \lambda)$
$\pi_{q_{1,1}}$	$P(q_{1,1}\|\lambda)$
$a^{\mathcal{V}}_{q_{i,j}, q_{i-1,j}}$	$P(q_{i,j}\|q_{i-1,j}, \lambda_i^{\mathcal{V}})$
$\alpha_{i,j}^{\mathcal{V}}(q_{i,j})$	$P(o_{1,j}, \ldots o_{i,j}, q_{i,j}\|\lambda_j^{\mathcal{V}})$
$\beta_{i,j}^{\mathcal{V}}(q_{i,j})$	$P(o_{i+1,j}, \ldots o_{I,j}\|q_{i,j}, \lambda_j^{\mathcal{V}})$
$\gamma_{i,j}^{\mathcal{H}}(q_{i,j})$	$P(q_{i,j}\|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}})$
$\gamma_{i,j}(q_{i,j})$	$(\gamma_{i,j}^{\mathcal{H}}(q_{i,j}) + \gamma_{i,j}^{\mathcal{V}}(q_{i,j}))/2$

Table 1. HMM notation summary

Let us note $P_j^{\mathcal{V}} = \left[P(o_j^{\mathcal{V}}, q_j^{\mathcal{V}}|\lambda_j^{\mathcal{V}}) \prod_i P(q_{i,j}|o_i^{\mathcal{H}}, \lambda_i^{\mathcal{H}}) \right]$. $P_j^{\mathcal{V}}$'s can be computed with a modified version of the forward-backward algorithm which we describe next after introducing one last notation:

$$b_{q_{i,j}}^{\mathcal{H}}(o_{i,j}) = \begin{cases} b_{q_{i,j}}(o_{i,j}) & \text{if } j = 1 \\ b_{q_{i,j}}(o_{i,j}) \gamma_{i,j}^{\mathcal{H}}(q_{i,j}) & \text{if } j > 1 \end{cases}$$

3.1. The Forward α Variables

- Initialization: $\alpha_{1,j}^{\mathcal{V}}(q_{1,j}) = \begin{cases} \pi_{q_{1,1}} b_{q_{1,1}}(o_{1,1}) & \text{if } j = 1 \\ b_{q_{1,j}}^{\mathcal{H}}(o_{1,j}) & \text{if } j > 1 \end{cases}$
- Recursion:

$$\alpha_{i+1,j}^{\mathcal{V}}(q_{i+1,j}) = \left[\sum_{q_{i,j}} \alpha_{i,j}^{\mathcal{V}}(q_{i,j}) a^{\mathcal{V}}_{q_{i,j}, q_{i+1,j}} \right] b^{\mathcal{H}}_{q_{i+1,j}}(o_{i+1,j})$$

- Termination: $P_j^{\mathcal{V}} = \sum_{q_{I,j}} \alpha_{I,j}^{\mathcal{V}}(q_{I,j})$

3.2. The Backward β Variables

- Initialization: $\beta_{I,j}^{\mathcal{V}} = 1$
- Recursion:

$$\beta_{i,j}^{\mathcal{V}}(q_{i,j}) = \sum_{q_{i+1,j}} a^{\mathcal{V}}_{q_{i,j}, q_{i+1,j}} b^{\mathcal{H}}_{q_{i+1,j}}(o_{i+1,j}) \beta_{i+1,j}^{\mathcal{V}}(q_{i+1,j})$$

3.3. Occupancy Probability γ

$$\gamma_{i,j}^{\mathcal{V}}(q_{i,j}) = \frac{\alpha_{i,j}^{\mathcal{V}}(q_{i,j}) \beta_{i,j}^{\mathcal{V}}(q_{i,j})}{\sum_{q_{i,j}} \alpha_{i,j}^{\mathcal{V}}(q_{i,j}) \beta_{i,j}^{\mathcal{V}}(q_{i,j})}$$

Similar formulae can be derived for the horizontal pass. It is worthwhile to note that our re-estimation equations are similar to the ones derived for the page layout problem in [6] based on the graphical model formalism. Also, we can see that the interaction between horizontal and vertical processing, which is based on the occupancy probability γ, is not as simple as the one used in [5].

Let us next consider the steps of the algorithm. We first initialize γ's uniformly (i.e. assuming no prior information). Then, the modified forward-backward algorithm is applied successively and iteratively on the rows and columns. Whether the iterative process is initialized with row or column operation may theoretically impact the performance. However, this choice had a very limited impact in our experiments and we always started with a horizontal pass. This algorithm is clearly linear in the size of the data and can be further accelerated with a parallel implementation, simply by running the modified forward-backward for each row or column on a different processor.

Our goal in this paper is to perform decoding. Both [5] and [6] used the following heuristic: they chose the most likely state for every observation. Although this is known to be suboptimal when one is concerned with the probability of occurrence of sequences of states [2], we tested this heuristic since it gave acceptable results. We also tried another approach which combines our iterative decoding (ID) and PCVSV (see section 6) and which proved to perform better for our task.

4. A MEASURE OF CONVERGENCE

Although we cannot guarantee that the ID algorithm converges, i.e. that the horizontal and vertical passes will "agree" in a well defined sense, we can define and employ a measure of convergence. The *divergence* or *Kullback Leibler distance* between two probability mass functions p and q is defined as [7]:

$$\mathcal{D}(p,q) = \sum_k p(k) \log \frac{p(k)}{q(k)}.$$

Let $\gamma^{\mathcal{H}}$ (resp. $\gamma^{\mathcal{V}}$) be the joint distributions of the $\gamma^{\mathcal{H}}_{i,j}(k)$'s (resp. $\gamma^{\mathcal{V}}_{i,j}(k)$'s). $\mathcal{D}(\gamma^{\mathcal{H}}, \gamma^{\mathcal{V}})$ is a measure of how well the horizontal and vertical decodings agree over the entire image. If we further assume independence of $\gamma^{\mathcal{H}}_{i,j}(k)$'s and similarly of $\gamma^{\mathcal{V}}_{i,j}(k)$'s, then

$$\mathcal{D}(\gamma^{\mathcal{H}}, \gamma^{\mathcal{V}}) = \sum_{i,j} \mathcal{D}(\gamma^{\mathcal{H}}_{i,j}, \gamma^{\mathcal{V}}_{i,j})$$

The measure of convergence is useful as a stopping criterion. After the k-th pass of ID, we compute $\mathcal{D}^{(k)}(\gamma^{\mathcal{H}}, \gamma^{\mathcal{V}})$ and stop iterating if $\mathcal{D}^{(k)} - \mathcal{D}^{(k-1)}$ falls below a pre-defined threshold θ. Generally, only two to three iterations were necessary to converge in our experiments.

5. OPTIMAL SEPARABILITY

In section 2 we approximated $P(q_{i,j}|q_{i,j-1}, q_{i-1,j}, \lambda)$ by the product of horizontal $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$ and vertical $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$ factors. Here we derive equations for the optimal horizontal and vertical components. We then show that, if $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$'s and $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$'s are optimal, then they effectively approximate $P(q_{i,j}|q_{i,j-1}, \lambda)$ and $P(q_{i,j}|q_{i-1,j}, \lambda)$, respectively.

Let us first consider the problem generally. Consider a conditional distribution $p_{i|jk}$ where $\sum_i p_{i|jk} = 1, \forall(j,k)$. We want to approximate $p_{i|jk}$ into the product $a_{ij}b_{ik}$, where a_{ij} and b_{ik} are non negative and satisfy the requirement: $\sum_i a_{ij} = 1, \forall j$ and $\sum_i b_{ik} = 1, \forall k$. A positive normalization factor n_{jk} is needed to ensure: $\sum_i n_{jk}a_{ij}b_{ik} = 1, \forall(j,k)$. Since for all (j,k), both $p_{i|jk}$ and $n_{jk}a_{ij}b_{ik}$ are probability distributions, we can define the divergence between them:

$$\mathcal{D}_{jk} = \sum_i p_{i|jk} \log \left(\frac{p_{i|jk}}{n_{jk}a_{ij}b_{ik}} \right)$$

Our goal is to minimize $\sum_{j,k} \mathcal{D}_{jk}$ subject to the above constraints. We hence minimize the Lagrangian:

$$\mathcal{L} = \sum_{j,k} \mathcal{D}_{jk} + \sum_j \lambda_j (\sum_i a_{ij} - 1) + \sum_k \mu_k (\sum_i b_{ik} - 1)$$

We obtain the following formulae:

$$\frac{\partial \mathcal{L}}{\partial a_{ij}} = 0 \quad \Rightarrow \quad a_{ij} = \frac{\sum_k p_{i|jk}}{\sum_{i,k} p_{i|jk}} \; \forall(i,j)$$

$$\frac{\partial \mathcal{L}}{\partial b_{ik}} = 0 \quad \Rightarrow \quad b_{ik} = \frac{\sum_j p_{i|jk}}{\sum_{i,j} p_{i|jk}} \; \forall(i,k)$$

$$\sum_i n_{jk}a_{ij}b_{ik} = 1 \quad \Rightarrow \quad n_{jk} = \frac{1}{\sum_i a_{ij}b_{ik}} \; \forall(j,k)$$

Since index j and k run from 1 to J and K, respectively, we can simplify the formulae for a_{ij} and b_{ik}:

$$a_{ij} = \frac{\sum_k p_{i|jk}}{K} \qquad b_{ik} = \frac{\sum_j p_{i|jk}}{J}$$

Now to interpret the result we observe that in general $p_{i|k} = \sum_j p_{i|jk}p_{j|k}$. If we further assume that $p_{j|k}$ is maximally non-informative, i.e., uniformly distributed then we obtain

$$p_{i|k} = \sum_j \frac{p_{i|jk}}{J},$$

which is exactly the formula we derived for b_{ik} above. A similar observation can be made regarding a_{ij}.

Next we specialize to the problem of interest, hence, $p_{i|jk}$ is replaced with $P(q_{i,j}|q_{i,j-1}, q_{i-1,j}, \lambda)$, a_{ij} with $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$ and b_{ik} with $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$. So, when $f^{\mathcal{H}}(q_{i,j}, q_{i,j-1})$'s (resp. $f^{\mathcal{V}}(q_{i,j}, q_{i-1,j})$'s) are chosen optimally, they approximate $P(q_{i,j}|q_{i,j-1}, \lambda^{\mathcal{H}}_i)$'s (resp. $P(q_{i,j}|q_{i-1,j}, \lambda^{\mathcal{V}}_j)$'s) assuming no prior information on $P(q_{i-1,j}|q_{i,j-1}, \lambda)$ (resp. $P(q_{i,j-1}|q_{i-1,j}, \lambda)$).

6. PATH-CONSTRAINED VARIABLE STATE VITERBI

In this section, we will first introduce the PCVSV algorithm and then show how to combine it in a simple manner with our proposed algorithm to exploit their complementary contributions.

6.1. The PCVSV algorithm

We recall that $q^{\mathcal{H}}_i$ is the sequence of states on the i-th row. $q^{\mathcal{H}}_i$'s can be seen as states of a 1-D HMM. However, this 1-D HMM has such a huge number of states that the direct application of the Viterbi algorithm is often unpractical. The central idea in PCVSV is to consider only the N sequences with the largest posterior probabilities (hence the name "Path Constrained Variable State Viterbi" algorithm). A fast algorithm is designed to avoid the calculation of posterior probabilities for all state sequences. It separates the blocks on a row from other blocks by neglecting their statistical dependencies. So the selection of N near optimal nodes for row i consists simply of identifying N state sequences with the largest $\sum_j \log b_{q_{i,j}}(o_{i,j})$.

Columns or diagonals could also be chosen instead of rows. In [1], diagonals are chosen since blocks on diagonals are more geometrically distant than blocks on rows or columns and are therefore expected to exhibit less correlation. In the following, PCVSV-H (resp. PCVSV-D) will be the notation for the horizontal (resp. diagonal) variant of the PCVSV algorithm.

6.2. A Combined Technique

PCVSV is guaranteed to converge to an at least local maximum-likelihood solution for sufficiently large N, the number of preselected paths. However, one may have to choose a very large value N to find an optimal path and hence defeat the purpose of the method. Indeed, the preselection of paths is inefficient since we only take into account local information. On the other hand, the computation of γ's during the iterative decoding takes into account both local and context information but the selection of the best path is admittedly suboptimal.

Hence the idea is to combine both approaches. The preselection of N nodes for PCVSV is based on the γ's computed during the iterated forward-backward. During the preselection, we will look for the N state sequences with the largest $\sum_j \log \gamma_{i,j}(q_{i,j})$. As we will see in the next section, this combined approach provides significant performance improvement over the individual application of the proposed approach or PCVSV.

7. EXPERIMENTAL RESULTS

7.1. The Database

We use 40 images from the ORL face database [8] and we generate synthetic images using a 2-D HMM (c.f. Fig. 1). The original ORL images are tiled into blocks which can undergo two types of transformations: Gaussian noise addition and small shifts. The parameters of the Gaussian noise (mean and variance) are determined by the emission probabilities of the 2-D HMM and the shifts by the transition probabilities. For each original image, we generate 25 images, which results in a total of 1,000 synthetic images.

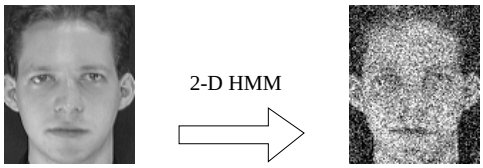

Fig. 1. Transformation of a face through a 2-D HMM

7.2. Results

The goal is to find the best possible match between the blocks of a synthetic image and its associated ORL image knowing the 2-D HMM parameters. The performance measure is the difference between the log-likelihood of the path Q that generated the synthetic data and the log-likelihood of the best path that is found. The larger this quantity, the closer is the match to the maximum-likelihood solution. This quantity may even exceed zero if the algorithm finds a path that "explains" the data better than Q. This score was plotted on Fig. 2 as a function of the computation time to match two images. For ID and PCVSV, the maximum number of iterations or the number N of preselected paths are indicated.

We can see that PCVSV-H is outperformed by both PCVSV-D (as expected) and ID, and that PCVSV-D performs slightly better than ID. This is due to the suboptimal choice for the best path in ID. However, when PCVSV-D is initialized with ID, the performance is significantly better than each algorithm taken separately, at the expense of a very modest amount of additional computation.

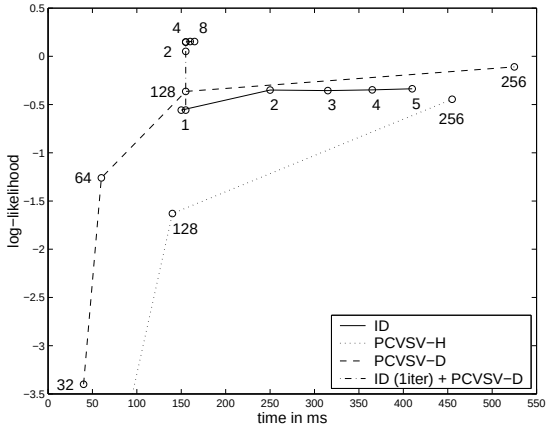

Fig. 2. Performance of ID, PCVSV-H, PCVSV-D and ID + PCVSV-D

8. CONCLUSION

We introduced in this paper a novel algorithm for decoding 2-D HMMs. The idea is to approximate a 2-D HMM by a T-HMM and to perform the forward-backward algorithm alternatively and iteratively on rows and columns. We derived re-estimation equations and discussed convergence and separability issues. We compared the performance of the proposed approach with the PCVSV algorithm and combined them into a new algorithm that outperforms each of the individual techniques.

While this paper focused on decoding, future work will concentrate on the problem of training 2-D HMMs based on the same principles.

9. REFERENCES

[1] J. Li, A. Najmi and R. M. Gray, "Image classification by a two-dimensional hidden markov model," *IEEE Trans. on Signal Processing*, vol. 48, no. 2, Feb 2000.

[2] L. R. Rabiner, "A tutorial on hidden markov models and selected applications," *Proc. of the IEEE*, vol. 77, no. 2, Feb 1989.

[3] S.-S. Kuo, O. Agazzi, "Keyword spotting in poorly printed documents using pseudo 2-d hidden markov models," *IEEE Trans. on PAMI*, vol. 16, no. 8, Aug 1994.

[4] K. Abend, T. J. Harley and L. N. Kanal, "Classification of binary random patterns," *IEEE Trans. on Information Theory*, vol. 11, no. 4, 1965.

[5] C. Miller, B. R. Hunt, M. A. Neifeld and M. W. Marcellin, "Binary image reconstruction via 2-d viterbi search," in *ICIP*, 1997, vol. 1, pp. 181–184.

[6] T. A. Tokuyasu, *Turbo Recognition: an Approach to Decoding page Layout*, Ph.D. thesis, University of California, Berkeley, 2001.

[7] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, Inc., 1993.

[8] F. S. Samaria, *Face Recognition Using Hidden Markov Models*, Ph.D. thesis, Trinity College, Cambridge University, 1994.