Evaluation of Incremental Learning Algorithms for An HMM-Based Handwritten Isolated Digits Recognizer

Paulo R. Cavalin
École de Technologie Supérieure
cavalin@livia.etsmtl.ca

Robert Sabourin
École de Technologie Supérieure
robert.sabourin@etsmtl.ca

Ching Y. Suen
Concordia University
suen@encs.concordia.ca

Alceu S. Britto Jr.
Pontifícia Universidade Católica do Paraná
alceu@ppgia.pucpr.br

Abstract

We present an evaluation of incremental learning algorithms for the estimate of HMM parameters. The main goal was to investigate incremental learning algorithms that can replace traditional batch learning techniques, incorporating the advantages of incremental techniques for designing complex pattern recognition systems. Experiments were carried out on isolated digits, extracted from the NIST SD19, by using a state-of-the-art HMM-based isolated digits recognizer. The experiments demonstrated that batch learning performs slightly better for generating classifiers with good generalization performance. However, the results obtained by the Ensemble Training algorithm are very encouraging for pursuing further research in this subject, given that the loss in terms of performance is relatively small. Furthermore, we demonstrated that the incremental approaches provide lower-cost algorithms, which is a valuable advantage.

Keywords: Incremental Learning, Hidden Markov Models, Handwriting Recognition, Isolated Digits

1 Introduction

The main goal of a complex pattern recognition system's architect is to design classifiers that result in high generalization performances [3]. Most of the performance of a classifier comes from its parameters, which are generally adjusted by means of a training database and a learning algorithm [4, 7, 9, 10].

Traditionally, a Batch Learning (BL) setting is a standard procedure for learning parameters, and is known to be very robust [2]. Basically, a BL approach consists of learning a classifier's parameters from a training dataset, and the learning algorithm executes as many iterations on the training set as necessary for tuning such parameters.

Despite its robustness, BL presents some drawbacks. First, the training database may not be a good representation of the general problem to which the system is related, and the classifiers will provide poor generalization performances no matter how good the learning algorithm is. This problem could be solved by incorporating new information that is available through the execution of the system to which the classifiers are associated, but there is no known way to do this unless we train a new classifier using both the old and the new data. That is the second major drawback of BL approaches, because it requires lots of time and memory.

Incremental Learning (IL) is a promising solution for the problems found in BL approaches . IL consists of techniques that are originally proposed to enable classifiers to gather more information without having to access previously-learnt data. IL is also meant to be as robust as BL to estimate parameters of classifiers, but recent research has suggested that IL performs worse than BL [4, 7, 9, 10].

Since the performance of a learning algorithm, as stated by the no free-lunch theorem [3], is strictly dependent on the problem to which it is applied, the main goal of this paper is to provide an evaluation of IL algorithms in the handwritten isolated digit recognition problem, by considering a state-of-the-art HMM-based handwriting recognition system [1]. We have chosen an HMM-based framework due to the potential of this modeling technique for the handwriting recognition problem in general. Furthermore, by considering the large NIST SD19 digits database, it is possible to perform simulations of IL settings in order to observe the evolution of each learning algorithm involved in this study.

The remainder of this paper is organized as following. In Section 2 we present an overview of IL techniques focused on HMMs. Next, in Section 3 we present the methodology employed to undertake this work. Afterwards, in Section 4 we present the setup of the experimental evaluation and the corresponding results. Finally, the conclusions drawn from this work are described in Section 5

2 Incremental Learning Algorithms for HMMs

Incremental Learning is a topic with increasing interest in research involving HMMs and pattern recognition systems. In the latter, HMMs are used to compose HMM-based classifiers, in which each class is represented by one or more HMMs. IL of HMMs basically consists of updating the HMM-based classifier when unseen data is available. Unseen data may be represented by either a single observation sequence or a block of observation sequences.

An update of an HMM-based classifier may be conducted in two different ways. In the first way, the IL approach updates only the parameters of the existing HMMs when new data is available [4, 7, 9]. In the second way, the IL approach may also add new HMMs to the HMM-based classifier. The advantage of the first way lies in the complexity of the resulting classifier, which always remain stable. Such stability in terms of complexity is not assured by the second approach, since new HMMs are frequently appended to the HMM-based classifier resulting in an growth in terms of recognition complexity. For this reason, we focus only on the first approach in this work.

The algorithms that update an existent HMM employ the following idea. Suppose the learning method is receiving a block of data D_t, at a given time $t \geq 1$ given the current HMM λ_{t-1}, an update consist of computing the parameters of the new HMM λ_t, where:

$$\lambda_t = \lambda'_{t-1} \tag{1}$$

In this case, λ'_{t-1} corresponds to a mathematical transformation involving both λ_{t-1} and ϕ_t, the sufficient statistics computed from D_t and λ_{t-1}.

All the algorithms that fall in this category differ mainly in three aspects: 1) the amount of data accumulated in D_t; 2) the importance of the data presented in D_t; and 3) the combination of λ_{t-1} and ϕ_t.

Regarding the first aspect, D_t can be composed of a single observation sequence, as in [4, 9], or it can be composed of a block of N_t samples, as in [7]. With a smaller number of samples in D_t, the learning algorithm performs more updates in λ_{t-1}, and consequently may converge very quickly. However, by saving more observation sequences in D_t, the algorithm is less sensitive to noise in the data stream, but more memory and time is necessary to store and process the block of data.

The second aspect refers to the learning rate of D_t [9]. This rate is important to define the behavior of the algorithm in terms of conservatism and adaptation. The higher is the learning rate of D_t, the more adaptive the algorithm to new data. Old data is forgotten very quickly. On the other hand, lower learning rates define an algorithm that gives as much importance to newer data as it does to older one, conserving all the observed information as long as possible.

The third aspect involves how λ_{t-1} and ϕ_t are combined to generate λ'_{t-1}. One solution presented in the literature consists in updating λ_{t-1} by performing a partial expectation step (E-step) of the Baum-Welch algorithm [11, 4, 9, 7], then the maximization step (M-step) is executed after each time t. One problem of this approach is that once one parameter in λ_{t-1} is equal to zero, there is no way to re-estimate this parameter. To work around that problem, a small constant ϵ is added to each parameter in λ_{t-1}, but this solution results in noise for the recognition process and additional evaluations are required for finding the best value of ϵ.

3 Methodology

The methodology employed in this work includes the baseline isolated digits recognizer, presented in Section 3.1, the IL algorithms related to this work, presented in Section 3.2, and the method proposed for complexity analysis, presented in Section 3.3.

3.1 The Baseline System

The baseline system is the isolated digits recognizer presented in [1]. This recognizer is divided into three modules: Pre-processing, Feature Extraction, and Recognition (see Figure 1 for an overview of this system).

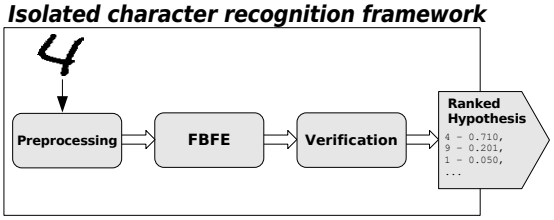

Figure 1. An overview of the isolated character recognition framework

The Pre-processing module performs corrections of slant inclination in isolated digits. The Foreground-Background Feature Extraction (FBFE) module extracts two observation sequences, one considering column observations and another considering row observations, based on a sliding-window approach. Each observation represents a 47-dimensional feature vector combining both foreground and background information. The Recognition module combines both column and row likelihoods to classify the corresponding image into one of the 10 classes of digits.

3.2 Incremental Learning algorithms

In this section we provide a brief description of three IL algorithms for HMM, pointing out advantages and dis-

advantages of each one.

3.2.1 The Incremental Baum-Welch algorithm

The Incremental Baum-Welch (IBW) algorithm is a straight-forward adaptation of the original BL Baum-Welch algorithm to IL. First proposed in [12] for continuous HMMs, it was later adapted to discrete models in [5].

The IBW algorithm consists of performing a partial E-step using just a single observation sequence, and a M-step for each time step t. In other words, the values of $\overline{a}_{ij}$ and $\overline{b}_j(k)$, respectively corresponding to the matrices A and B of an HMM, are updated at each time step t, given λ_{t-1} and D_t, where D_t is composed of a single observation sequence.

Mathematically, the estimator $\overline{a}_{ij}$, at the current time step t, is given by:

$$\overline{a}_{ij}^{t} = \frac{\overline{a}_{ij}^{t-1}(\sum_{t'=1}^{t-2}\gamma_{t'}(i)) + \xi_{t-1}(i,j)}{\sum_{t'=1}^{t-1}\gamma_{t'}(i)} \tag{2}$$

and the output probability, $\overline{b}_j^t(k)$, is given by:

$$\overline{b}_j^{t}(k) = \frac{\overline{b}_j^{t-1}(k)(\sum_{t'=1}^{t-1}\gamma_{t'}(j)) + \psi(t,j,k)}{\sum_{t'=1}^{t}\gamma_{t'}(j)} \tag{3}$$

where $\psi(t,j,k)$ is an auxiliary function defined as:

$$\psi(t,j,k) = \begin{cases} 0 & \text{if} O_t \neq v_k \\ \delta_t(j) & \text{otherwise} \end{cases} \tag{4}$$

As we can see in the above equations, both $\overline{a}_{ij}$ and $\overline{b}_j(k)$ are updated by taking into account the sufficient statistics stored in $\gamma_{t'}(i)$ from $t' = 1, \ldots, t$. Although just a single observation sequence is taken into account to update λ_{t-1}, the sufficient statistics represent information computed from all observed data.

In [4], the addition of a constant ϵ in the matrices A and B was proposed to avoid that some parameters receive the value zero. This may be done each time λ_t is computed, or after a time interval Δt.

3.2.2 The Incremental Maximum-Likelihood algorithm

In [7], the Incremental Maximum-Likelihood (IML) algorithm, which updates an existing HMM by considering a block of data, has been evaluated in an IL setting.

Since the main objective in the work was to speed up the learning process, the proposed IML algorithm works by dividing an off-line training database into smaller blocks. Each iteration of the algorithm processes a different block of data. Thus, given an initial HMM λ_0, and the blocks of data $D_t, 1 \leq t \leq T$ drawn from the training set, this algorithm works according to the following steps:

1) Initialize sufficient statistics $\phi_t \forall t$ to zero.

2) For $t = 1, 2, \ldots, T$ or until convergence do

- **a.** Compute sufficient statistics ϕ_t
- **b.** $\phi_t = \phi_t + \phi_{t-1}$
- **c.** Compute λ'_{t-1}
- **d.** $\lambda_t = \lambda'_{t-1}$

For an IL setting, IML can be easily adapted, although T is not known a priori. Furthermore, the blocks of data D_t are acquired over time.

Although this algorithm processes a small block of data in each iteration, the sufficient statistics ϕ_t are always updated taking into account the information from the previously-processed blocks. By using this approach, those authors claim that the algorithm presents the same performances of the the BL Expectation Maximization, but with a faster convergence.

Despite not being mentioned by the author, we believe that this algorithm also requires the addition of a small constant ϵ to the matrices A and B. Otherwise, the information can not be completely learnt by this algorithm if unseen observation sequences contain previously-unseen observations.

3.2.3 Ensemble Training

Another interesting approach for IL, namely Ensemble Training (ET), was proposed in [8]. Although this algorithm has never been employed within an IL setting (to our knowledge), this algorithm can be easily adapted to this kind of setting since the parameters of the final HMM (i.e. the model used for the recognition) are independently computed for each observation sequence. And despite being originally proposed to deal with single observation sequences, this algorithm can also be easily extended to work with blocks of observation sequences.

In a sense analogous to the Multi-sequence Baum-Welch (MSBW) algorithm [11], ET consists of independently doing the learning of each of the observation sequences from the training set so that each sequence generates a corresponding HMM. After all the sequences are learnt, the corresponding models are combined to generate a single model representing the whole data.

In greater detail, ET works as follows. Suppose that K observation sequences are available for training, and for each of these K observation sequences, one model $\lambda_k = A_k, B_k, \pi_k$ is estimated by ET, resulting in the formation of K independent model estimates from the training set. From these K models, the matrices A, B, and π, for the final HMM, are computed in the following way:

$$\overline{a}_{ij} = \frac{\sum_k W_k a_{ij}^k}{\sum_k W_k} \tag{5}$$

$$\overline{b}_{ij} = \frac{\sum_k W_k b_{ij}^k}{\sum_k W_k} \tag{6}$$

$$\overline{\pi}_i = \frac{\sum_k W_k \pi_i^k}{\sum_k W_k} \quad (7)$$

where W_k is the weighting factor for each sequence.

A straight-forward way to adapt ET to work in an IL setting is by conserving a current HMM λ_{t-1}, which corresponds to all data up to the time step $t-1$. The re-estimation of A, B, and π (the new current model λ_t), when new data is available, considers only λ_{t-1} and the model generated at time t ($\lambda_{t'}$). One important aspect to assure that the older information is kept in λ_t is to consider the weights of the previously-seen data, by accumulating both W_{t-1} and $W_{t'}$ into W_t. In detail, suppose we are updating the model $\lambda_{t-1} = A_{t-1}, B_{t-1}, \pi_{t-1}$ after observing the data t, thus we compute $\lambda_t = A_t, B_t, \pi_t$, given $\lambda_{t'}$, by using the following equations:

$$\overline{a}_{ij}^t = \frac{W_{t-1} a_{ij}^{t-1} + W_{t'} a_{ij}^{t'}}{W_{t-1} + W_{t'}} \quad (8)$$

$$\overline{b}_{ij}^t = \frac{W_{t-1} b_{ij}^{t-1} + W_{t'} b_{ij}^{t'}}{W_{t-1} + W_{t'}} \quad (9)$$

$$\overline{\pi}_i^t = \frac{W_{t-1} \pi_i^{t-1} + W_{t'} \pi_i^{t'}}{W_{t-1} + W_{t'}} \quad (10)$$

$$W_t = W_{t-1} + W_{t'} \quad (11)$$

The ET algorithm is very flexible because any learning algorithm can be used to generate the HMM corresponding to the new data, including the original Baum-Welch algorithm. Furthermore, we see that ET is a technique similar to Ensembles of Classifiers, despite generating only one HMM for the classification scheme, which is a result of the combination function used by this approach.

3.3 Complexity Analysis

The methodology proposed for complexity analysis is useful to compare different learning methodologies employing the same training data. Such methodology is based on the number of samples in each block of data, and the total number of iterations until convergence on each block.

Suppose NB is the number of blocks of data, NS_i corresponds to the number of samples in block i, where $1 \leq i \leq NB$, and NI_i corresponds to the number of iterations to converge on block i, so the complexity factor CF for learning all data is defined by:

$$CF = \sum_{i}^{NB} (NS_i \times NI_i) \quad (12)$$

The complexity for estimating the parameters of a single sample is not taken into account by this method. In this work this information is not relevant due to the nature of the algorithms involved in this paper, which share the same re-estimation procedure (i.e., they are all based on the forward-backward procedure).

4 Experimental Evaluation

The experimental evaluation of the different learning algorithms considers isolated digits from the NIST SD19 database. The isolated digits are organized in 195,000 samples for training (equally distributed into 19,500 samples per class), 28,000 for validation (both from hsf_{0,1,2,3}), and 60,089 samples for test taken from (hsf_7).

An IL setting is simulated by dividing the training dataset into 19 blocks of 10,000 samples (1,000 samples per class), and one block of 5,000 samples (500 samples per class). The learning is carried out by presenting one block at a time, and the algorithm progressively incorporates the information of each new block. The performance evaluation is after processing each block, on both validation and test sets.

For all the experiments, we considered a codebook of 256 symbols, built from the whole training set. The number of states for the HMMs were optimized using Wang's method, as described in [1].

The experiments are fourfold. First, we validated the implementation of the system in a BL setting, using the traditional Baum-Welch algorithm. Then, we repeated the same experiment with the three algorithms presented in Section 3.2. All the experiments were repeated five times (except for BL due to complexity reasons), using different samples in each block, in order to provide a better statistical estimate of the results, and the recognition rates are represented by the average of the five runs. In order to compare the performances of the algorithms, we consider the performances of the classifiers generated after learning the whole training set.

4.1 Evaluation of The Batch Learning Setting

We evaluated the system in a BL setting, using the traditional Baum-Welch algorithm. For this task, several iterations were performed on the training set until convergence was reached. The validation set was used to select the models with the lowest errors.

In order to evaluate the impact of the training set's size, and to simulate the application of this method for IL, we used training sets with sizes from 1,000 samples up to 19,500 per class. See in Figure 2 the results on the validation set, and in Figure 3 the results on the test set. After learning the whole training set, this method results in classifiers with accuracy of 98.94% on the validation set, and 97.88% on the test set.

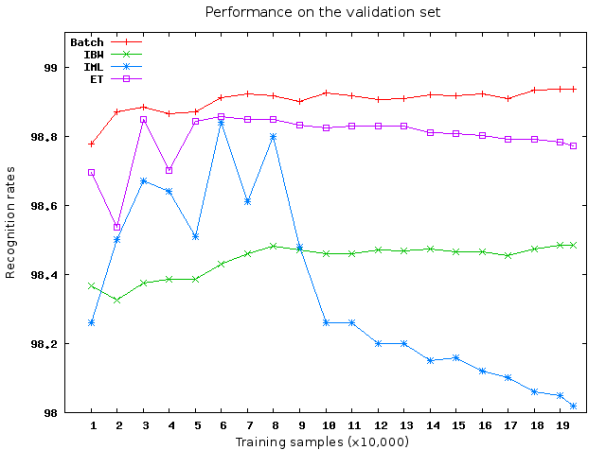

Figure 2. The recognition results of all algorithms on the validation set.

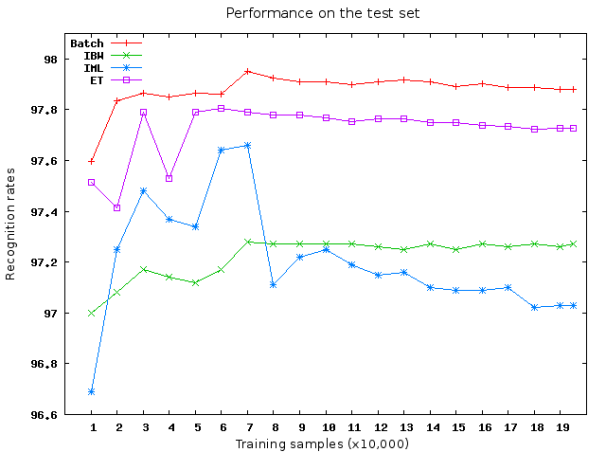

Figure 3. The recognition results of all algorithms on the test set.

4.2 Evaluation of The Incremental Baum-Welch Algorithm

For the evaluation of the Incremental Baum-Welch (IBW) algorithm, the only additional parameter that had to be evaluated was the constant ϵ. That constant was set equal to 0.00001 after a few preliminary evaluations on the validation set.

See in Figures 2 and 3 the results of this algorithm on the validation and on the test set, respectively. The recognition rates reported by the last classifier, after learning the 20 blocks of data (i.e. 195,000 samples), are 98.42% and 97.27% on the validation and on the test set respectively.

4.3 Evaluation of The Incremental ML Algorithm

For evaluating the Incremental Maximum-Likelihood (IML) algorithm, the only additional parameter required by this algorithm is the constant ϵ, similar to IBW. After preliminary evaluations on the validation set, ϵ was set equal to 0.0001

Figures 2 and 3 respectively show the results of this algorithm on the validation and on the test set. The recognition rates reported by the last classifier, after learning the 20 blocks of data (i.e. 195,000 samples), are 98.02% and 97.03% on the validation and on the test set respectively.

4.4 Evaluation of Ensemble Training

The evaluation of the Ensemble Training (ET) algorithm comprised a few preliminary steps in order to define the best method for learning each HMM corresponding to new data. We evaluated the employment of the validation dataset for controlling the number of iterations on the training set, and also different pre-defined numbers of iterations. We observed that by using 10 training iterations we obtain the best results. Furthermore, we also observed that by iterating on the whole block of data the algorithm generates better results than by iterating on each observation sequence independently.

See also in Figures 2 and 3 the results of this algorithm, respectively on the validation set and on the test set. After learning the whole training set, this algorithm presents recognition rates of 98.77% and 97.73% on the validation set and on the test set.

4.5 Discussion

We see from this experimental evaluation that BL trained the classifiers that resulted in the best performances. Nonetheless, ET presented very promising results, with a difference in the recognition rates of only 0.17% and 0.15%, on the validation and the test set respectively. On the other hand, the other IL algorithms produced very poor performances in these experiments.

In Figure 4 we show the evaluation of BL and ET considering a rejection method employing multiple thresholds (see [6]). We observed that by considering no rejection error, e.g. a rejection error equal to 0.0, the difference between the recognition errors of BL and ET is reduced to 0.06% (over 0.17% for the zero-level rejection experiments). Nonetheless, the rejection rate of ET is very similar to the one produced by BL. In some cases, ET rejects less samples than BL.

Despite the performance aspects, ET is also an interesting approach in terms of complexity. Figure 5 demonstrates the complexity analysis of the algorithms using the methodology presented in Section 3.3. Even though in these experiments it was ten times slower than the other

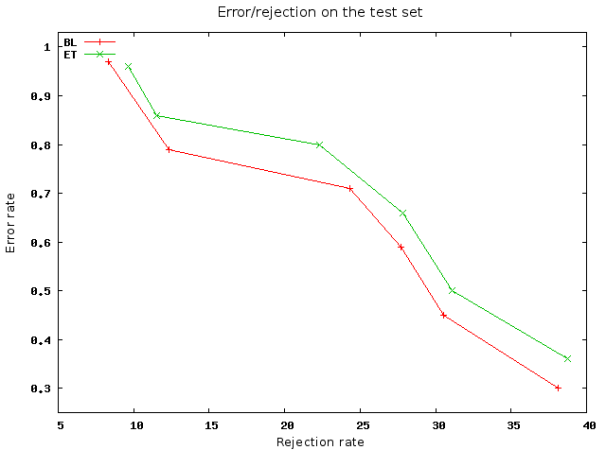

Figure 4. Error rate and rejection rate analysis of BL and ET.

incremental solutions, ET was about four times faster than BL.

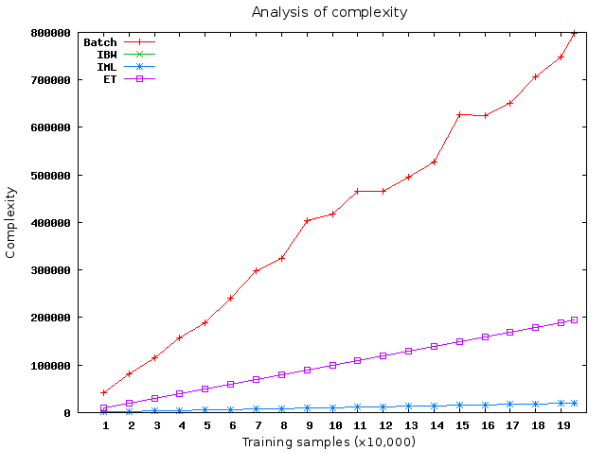

Figure 5. Complexity analysis of all algorithms, employing the methodology described in Section 3.3.

5 Conclusions and Future Work

In this work we presented the evaluation of three different IL algorithms for HMMs. We compared their performances with the traditional Baum-Welch algorithm in a BL setting. From the experimental results presented in this paper, we clearly see that BL performs slightly better that IL algorithms. Nevertheless, ET has demonstrated only a small loss compared to BL, which is a very promising result. Furthermore, ET requires fewer iterations than BL to generate good models in terms of generalization performance and can be executed in a parallel architecture, which are both important factors when dealing with large datasets.

We can pursue this work in several directions. We believe that a promising direction is the investigation of different methods for merging the models in the ET algorithm, and also the investigation of different schemes for weighting each new model. Furthermore, we should also investigate the employment of ordinary ensembles of classifiers instead of merging the models, despite the increase of complexity for the classification scheme.

References

[1] A. S. Britto, R. Sabourin, F. Bortolozzi and C. Y. Suen, "Recognition of Numeral Strings Using a Two-Stage HMM-Based Method", *Internation Journal on Document Analysis and Recognition*, 5(2):102–117, 2003.

[2] J.-X. Dong, A. Krzyzak and C. Y. Suen, "Fast SVM Training Algorithm with Decomposition on Very Large Data Sets", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):603–618, April 2005.

[3] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, Wiley-Interscience Publication, 2000.

[4] G. Florez-Larrahondo, *Incremental Learning of Discrete Hidden Markov Models* .PhD thesis, Mississipi State University, 2005.

[5] G. Florez-Larrahondo, S. Bridges and E. A. Hansen, "Incremental Estimation of Discrete Hidden Markov Models Based on a New Backward Procedure", *Proceedings the National Conference on Artificial Intelligence*, 2005, pp 758–763.

[6] G. Fumera, F. Roli and G. Giacinto, "Reject option with multiple thresholds", *Pattern Recognition*, 33(12):2099–2101, 2000.

[7] Y. Gotoh, M. M. Hochberg and H. F. Silverman, "Efficient Training Algorithms for HMM's Using Incremental Estimation", 6(6):539–548, 1998.

[8] D. J. C. Mackay, "Ensemble Learning for Hidden Markov Models", Technical report, Cavendish Laboratory, University of Cambridge, IK, 1997.

[9] J. Mizuno, T. Watanabe, K. Ueki, K. Amano, E. Takimoto and A. Maruoka, "On-line Estimation of Hidden Markov Model Parameters", *Proceedings of Third International Conference Discovery Science (DS 2000)*, 2000, volume 1967, pp 155–169.

[10] R. Polikar, L. Udpa, S. S. Udpa and V. Honavar, "Learn++: An Incremental Learning Algorithm for Supervised Neural Networks", *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 31(4):497–508, 2001.

[11] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE*, 77(2):257–286, 1989.

[12] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee and J. Buhmann, "Topology Free Hidden Markov Models: Application to Background Modeling", *Proceedings of IEEE International Conference on Computer Vision*, 2001, volume 1, pp 294–301.