Missaoui *et al. EURASIP Journal on Advances in Signal Processing* 2013, **2013**:40
http://asp.eurasipjournals.com/content/2013/1/40

EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH **Open Access**

Multi-stream continuous hidden Markov models with application to landmine detection

Oualid Missaoui[1], Hichem Frigui[2*] and Paul Gader[3]

Abstract

We propose a multi-stream continuous hidden Markov model (MSCHMM) framework that can learn from multiple modalities. We assume that the feature space is partitioned into subspaces generated by different sources of information. In order to fuse the different modalities, the proposed MSCHMM introduces stream relevance weights. First, we modify the probability density function (pdf) that characterizes the standard continuous HMM to include state and component dependent stream relevance weights. The resulting pdf approximate is a linear combination of pdfs characterizing multiple modalities. Second, we formulate the CHMM objective function to allow for the simultaneous optimization of all model parameters including the relevance weights. Third, we generalize the maximum likelihood based Baum-Welch algorithm and the minimum classification error/gradient probabilistic descent (MCE/GPD) learning algorithms to include stream relevance weights. We propose two versions of the MSCHMM. The first one introduces the relevance weights at the state level while the second one introduces the weights at the component level. We illustrate the performance of the proposed MSCHMM structures using synthetic data sets. We also apply them to the problem of landmine detection using ground penetrating radar. We show that when the multiple sources of information are equally relevant across all training data, the performance of the proposed MSCHMM is comparable to the baseline CHMM. However, when the relevance of the sources varies, the MSCHMM outperforms the baseline CHMM because it can learn the optimal relevance weights. We also show that our approach outperforms existing multi-stream HMM because the latter one cannot optimize all model parameters simultaneously.

1 Introduction

Hidden Markov models (HMMs) have emerged as a powerful paradigm for modeling stochastic processes and pattern sequences. Originally, HMMs have been applied to the domain of speech recognition, and became the dominating technology [1]. In recent years, they have attracted growing interest in automatic target detection and classification [2], computational molecular biology [3], bioinformatics [4], mine detection [5], handwritten character/word recognition [6], and other computer vision applications [7]. HMMs are categorized into discrete and continuous models. An HMM is called continuous if the observation probability density functions are continuous and discrete if the observation probability density functions are discrete.

Continuous probability density functions have the advantage of covering the entire landscape of the feature space when dealing with continuous attributes. In fact, each data point would correspond to a unique probability density value that represents its likelihood or unique occurrence rate. The discrete HMM, on the other hand, reduces the feature space to a finite set of prototypes or representatives. The quantization is typically accompanied by a loss of information that tends to reduce the generalization accuracy. Therefore, in this article, we focus on the continuous version of HMM for classification.

For complex classification problems involving data with large intra-class variations and noisy inputs, no single source of information can provide a satisfactory solution. In these cases, multiple features extracted from different modalities and sensors may be needed. HMM approaches that combine multiple features can be divided into three main categories: feature fusion or direct identification; decision fusion or separate identification (known also as

*Correspondence: h.frigui@louisville.edu
[2]University of Louisville, Louisville, KY 40292, USA
Full list of author information is available at the end of the article

late integration); and model fusion (early/intermediate integration) [8]. In feature fusion, multiple features are concatenated into a large feature vector and a single HMM model is trained [9]. This type of fusion has the drawback of treating heterogeneous features equally important. Moreover, it cannot easily represent the loose timing synchronicity between different modalities. In decision fusion, the modalities are processed separately to build independent models [10]. This approach ignores the correlation between features and allows complete asynchrony between the streams. In addition, it is computationally heavy since it involves two layers of decision. In the third category, model fusion, a more complex HMM model than the standard one is sought. The additional complexity is needed to handle the correlation between modalities and the loose synchronicity between sequences when needed. Several HMM structures have been proposed for this purpose. Examples include factorial HMM [11], coupled HMM [12] and multi-stream HMM [13]. Both factorial and coupled HMM structures assign a state sequence to each stream and allow asynchrony between sequences [14]. However, the parameter estimation of these models is not trivial and only approximate solutions can be obtained. In particular, the parameters of factorial and coupled HMMs could be estimated via EM (Baum-Welch) algorithm. However, the E-step is computationally intractable and approximation approaches are used instead [11,12]. Multi-stream HMM (MSHMM) is an HMM based structure that handles multiple modalities for temporal data. It is used when the modalities (streams) are synchronous and independent.

Multi-stream HMM techniques have been proposed for both the discrete and the continuous cases [15-17]. In our earlier study [17], we have proposed a multi-stream HMM framework for the discrete case where two distinct structures that integrate a stream relevance weight for each symbol in each state. For each structure, we generalized the Baum-Welch [1] and the minimum classification error (MCE) [18] training algorithms. In particular, we modified the objective function to include the stream relevance weights and derived the necessary conditions to optimize all of the model parameters simultaneously.

For the continuous case, multi-stream HMM was originally introduced to fuse audio and visual streams in speech recognition using continuous HMM [15,16]. In these methods, the feature space is partitioned into subspaces and different probability density functions (pdf) are learned for the different streams. The relevance of the different streams were encoded by exponent weights and a weighted geometric mean of the streams is used to approximate the pdf. This geometric approximation of the pdf makes it impossible to derive the maximum likelihood estimates of the stream relevance weights [16], unless the model is restricted to include only one Gaussian component per state [15]. Consequently, a two step learning mechanism was adapted to learn all model parameters. In the first step the MLE (standard Baum-Welch algorithm) [1] is used to learn all model parameters, except the stream relevance weights. In the second step, a discriminative training algorithm is used to learn the exponent weights. The main drawback of this approach is its inability to provide an optimization framework that learns all the HMM parameters simultaneously unless the number of components per state is limited to one which can be too restrictive for most real applications. In addition, solving this issue using two layers of training that optimize two different types of parameters is susceptible to local optima. To alleviate these limitations, the authors in [19] proposed a MSHMM structure that allows for simultaneous learning of all model parameters, including the stream relevance weights, by linearizing the approximation of the pdf. In this approach, the stream relevance weight were introduced at the mixture level, and the Baum-Welch (BW) learning algorithm was generalized to derive the necessary conditions to learn all parameters simultaneously.

In this article, we extend the MSHMM structure proposed in [19] to the state level stream weighting and generalize the MLE learning algorithm for this structure. We also generalize the minimum classification error (MCE) learning to both mixture level and state level streaming.

The organization of the rest of the article is as follows. In Section 2, we outline the baseline CHMM with maximum likelihood and discriminative training. We also provide an overview of existing HMM based structures for multi-sensor fusion. In Section 3, we present our continuous multi-stream HMM structures and we derive the necessary conditions to optimize all parameters simultaneously using both MLE and MCE/GPD learning approaches. Section 4 has the experimental results that compare the proposed multi-stream HMM with existing HMM approaches. Finally, Section 5 contains the conclusions and future directions.

2 Related study

2.1 Baseline continuous HMM

An HMM is a model of a doubly stochastic process that produces a sequence of random observation vectors at discrete times according to an underlying Markov chain. At each observation time, the Markov chain may be in one of N_s states, $s_1, \ldots, s_{N_s}$ and given that the chain is in a certain state, there are probabilities of moving to other states. These probabilities are called the transition probabilities. An HMM is characterized by three sets of probability density functions, the initial probabilities (π), the transition probabilities ($\mathbf{A}$), and the state probability density functions (**B**). Let T be the length of the observation sequence (i.e., number of time steps), $O = [o_1, \ldots, o_T]$ be the observation sequence, where each observation vector o_t is characterized by p features (i.e., $o_t \in \mathbb{R}^p$), and $Q = [q_1, \ldots, q_T]$ be the state sequence. The compact notation

$$\lambda = (\pi, \mathbf{A}, \mathbf{B}) \tag{1}$$

is generally used to indicate the complete parameter set of the HMM model. In (1), $\pi = [\pi_i]$, where $\pi_i = \Pr(q_1 = s_i)$ are the initial state probabilities; $A = [a_{ij}]$ is the state transition probability matrix, where $a_{ij} = \Pr(q_t = j | q_{t-1} = i)$ for $i, j = 1, \ldots, N_s$; and $\mathbf{B} = \{b_i(o_t), i = 1, \ldots, N_s\}$, where $b_i(o_t) = \Pr(o_t | q_t = i)$ is the set of observation probability distribution in state i. For the continuous HMM, $b_i(o_t)$ are defined by a mixture of some parametric probability density functions (pdfs). The most common parametric pdf used in continuous HMM is the mixture Gaussian densities where

$$b_i(o_t) = \sum_{j=1}^{M_i} u_{ij} b_{ij}(o_t), \quad \text{for } i = 1, \ldots, N_s. \tag{2}$$

In (2), M_i is the number of components in state i, $b_{ij}(o_t)$ is a p-dimensional multivariate Gaussian density with mean μ_{ij} and a covariance matrix Σ_{ij}, and u_{ij} is the mixture coefficient for the jth mixture component in state i, and satisfies the constraints

$$u_{ij} \geq 0, \text{ and } \sum_{j=1}^{Mi} u_{ij} = 1, \text{ for } i = 1, \ldots, N_s. \tag{3}$$

For a C-class classification problem, each random sequence O is to be classified into one of the C classes. Each class, c, is modeled by a CHMM λ_c. Let $\mathbb{O} = [O^{(1)}, \ldots, O^{(R)}]$ be a set of R sequences drawn from these C different classes and let $g_c(O)$ be a discriminant function associated with classifier c that indicates the degree to which O belongs to class c. The classifier $\Gamma(O)$ defines a mapping from the sample space ($O \in \mathbb{O}$) to the discrete categorical set $\{1, 2, \ldots, C\}$. That is,

$$\Gamma(O) = I \quad \text{iff} \quad I = \underset{c=1,\ldots,C}{\operatorname{argmax}}\, g_c(O). \tag{4}$$

Two main approaches were considered to learn the HMM parameters. The first one is based on learning the model parameters that maximizes the likelihood of the training data. The second approach is based on discriminative training that minimizes the classification error over all classes.

2.1.1 CHMM with maximum likelihood estimation (MLE)

The Baum-Welch (BW) [1] is an MLE algorithm that is commonly used to learn the HMM parameters. It consists of adjusting the parameters of each model λ independently to maximize the likelihood $\Pr(O|\lambda)$. Maximizing $\Pr(O|\lambda)$ is equivalent to maximizing the auxiliary function:

$$\mathbb{Q}(\lambda, \bar{\lambda}) = \sum_Q \sum_F \Pr(Q, E | O, \lambda) \ln \Pr(O, Q, E | \bar{\lambda}), \tag{5}$$

where λ is the initial guess and $\bar{\lambda}$ is the subject of optimization. In fact, it was proven [20] that $\frac{\partial \Pr(O|\lambda)}{\partial \lambda} = \frac{\partial \mathbb{Q}(\lambda, \bar{\lambda})}{\partial \bar{\lambda}} |_{\bar{\lambda}=\lambda}$. In (5), $Q = [q_1, q_2, \ldots, q_T]$ is a random vector representing the underlying state at time slot t, and $E = [e_1, e_2, \ldots, e_T]$ is a random vector, where each e_t represents the index of the mixture component within the underlying state that is responsible for the generation of the observation o_t.

Using a mixture of Gaussian densities with diagonal covariance matrices, it can be shown that the HMM parameters **A** and **B** need to be updated iteratively using [1]:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \tag{6}$$

$$\bar{u}_{ij} = \frac{\sum_{t=1}^{T} \gamma_t^{(1)}(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} \tag{7}$$

$$\bar{\mu}_{ijd} = \frac{\sum_{t=1}^{T} \gamma_t^{(1)}(i,j) o_{td}}{\sum_{t=1}^{T} \gamma_t^{(1)}(i,j)} \tag{8}$$

$$\bar{\Sigma}_{ij} = \frac{\sum_{t=1}^{T} \gamma_t^{(1)}(i,j)(\mathbf{o}_t - \bar{\boldsymbol{\mu}}_{ij})^t (\mathbf{o}_t - \bar{\boldsymbol{\mu}}_{ij})}{\sum_{t=1}^{T} \kappa_t(i,j)} \tag{9}$$

In the above,

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N_s} \sum_{j=1}^{N_s} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)},$$

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{j=1}^{N_s} \alpha_t(j) \beta_t(j)},$$

$$\kappa_t(i,k) = \gamma_t(i) \left[\frac{u_{ij} b_{ij}(o_t)}{b_i(o_t)} \right]$$

The variables $\alpha_t(j)$ and $\beta_t(j)$ are computed using the Forward and Backward algorithms [1], respectively.

2.1.2 CHMM with discriminative training

The optimality of the MLE training criterion is conditioned on the availability of an infinite amount of training data and the correct choice of the model. Indeed, it was shown in [21] that, if the true distribution of the samples to be classified can be accurately described by the assumed statistical model and if the size of the training set tends to infinity, the MLE tends to be optimal. However, in practice, neither of these conditions are satisfied as the available training data are limited, and the assumptions made on the HMM structure are often inaccurate. As a consequence, the likelihood-based training may not be effective. In this case, minimization of the classification error rate is a more suitable objective than minimization of the error of the parameter estimates. A common discriminative training method is the MCE [18]. In fact, it has been reported since the mid-1990s that discriminative training techniques were more successful [18]. The optimization of the error function is generally carried out by the GPD algorithm [18], a gradient descent-based optimization, and results in a classifier with minimum error probability. Let,

$$g_c(O,\Lambda) = \log[\max_Q g_c(O,Q,\Lambda)], \tag{10}$$

be the discriminant function associated with classifier λ that indicates the degree to which O belongs to class c. In (10), Q is a state sequence correspondent to the observation sequence O, Λ includes the models parameters, and

$$\begin{aligned} g_c(O,Q,\Lambda) = \Pr(O,Q;\lambda_c) &= \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a^{(c)}_{q_t q_{t+1}} \prod_{t=1}^{T} b^{(c)}_{q_t}(o_t) \\ &= \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a^{(c)}_{q_t q_{t+1}} \prod_{t=1}^{T} \sum_{j=1}^{M} u^{(c)}_{q_t j} b^{(c)}_{q_t j}(o_t). \end{aligned} \tag{11}$$

Assuming that $\bar{Q} = (\bar{q}_0, \bar{q}_1, \ldots, \bar{q}_T)$ is the optimal state sequence that achieves $\max_Q g_c(O,Q,\Lambda)$, which could be computed using the Viterbi algorithm [22], Equation (10) can be rewritten as

$$g_c(O,\Lambda) = \log[g_c(O,\bar{Q},\Lambda)]$$

The misclassification measure of sequence O is defined by:

$$d_c(O) = -g_c(O,\Lambda) + \log\left[\frac{1}{1-C}\sum_{j,j\neq c}\exp[\eta g_j(O,\Lambda)]\right]^{\frac{1}{\eta}} \tag{12}$$

where η is a positive number. A positive $d_c(O)$ indicates misclassification, while a negative $d_c(O)$ indicates correct decision.

The misclassification measure is embedded in a smoothed zero-one function, referred to as loss function, defined as:

$$l_c(O,\Lambda) = l(d_c(O)), \tag{13}$$

where l is a sigmoid function, one example of which is:

$$l(d) = \frac{1}{1+\exp(-\zeta d + \theta)}. \tag{14}$$

In (14), θ is normally set to zero, and ζ is set to a number larger than one. Correct classification corresponds to loss values in $[0, \frac{1}{2})$, and misclassification corresponds to loss values in $(\frac{1}{2}, 1]$. The shape of the sigmoid loss function varies with the parameter $\zeta > 0$: the larger the ζ, the narrower the transition region. Finally, for any unknown sequence O, the classifier performance is measured by:

$$l(O;\Lambda) = \sum_{c=1}^{C} l_c(O;\Lambda)\mathbb{I}(O \in C_c) \tag{15}$$

where $\mathbb{I}(.)$ is the indicator function. Given a set of training observation sequences $O^{(r)}$, $r = 1, 2, \ldots, R$, an empirical loss function on the training data can be defined as

$$\mathbf{L}(\Lambda) = \sum_{r=1}^{R}\sum_{c=1}^{C} l_c(O^{(r)};\Lambda)\mathbb{I}(O^{(r)} \in C_c). \tag{16}$$

Minimizing the empirical loss is equivalent to minimizing the total misclassification error. The CHMM parameters are therefore estimated by carrying out a gradient descent on $\mathbf{L}(\Lambda)$. In order to ensure that the estimated CHMM parameters satisfy the stochastic constraints of $a_{ij} \geq 0$, $\sum_{j=1}^{N_s} a_{ij} = 1$ and $u_{ij} \geq 0$, $\sum_{j=1}^{M} u_{ij} = 1$, and $\mu_{ijd} \geq 0$, and $\Sigma_{ij} \geq 0$, these parameters are mapped using

$$\begin{aligned} a_{ij} \rightarrow \tilde{a}_{ij} = \log a_{ij},\ u_{ij} \rightarrow \tilde{u}_{ij} = \log u_{ij},\ \mu_{ijd} \rightarrow \tilde{\mu}_{ijd} \\ = \frac{\mu_{ijd}}{\Sigma_{ij}} \text{ and } \Sigma_{ij} \rightarrow \tilde{\Sigma}_{ij} = \log \Sigma_{ij} \end{aligned} \tag{17}$$

Then, the parameters are updated with respect to $\tilde{\Lambda}$. After updating, the parameters are mapped back using

$$\begin{aligned} a_{ij} = \frac{\exp \tilde{a}_{ij}}{\sum_{j'=1}^{N_s} \exp \tilde{a}_{ij'}},\ u_{ij} = \frac{\exp \tilde{u}_{ij}}{\sum_{j'=1}^{M} \exp \tilde{u}_{ij'}},\ \mu_{ijd} = \tilde{\mu}_{ijd}\sigma_{ijd}, \\ \text{and } \Sigma_{ij} = \exp \tilde{\Sigma}_{ij} \end{aligned} \tag{18}$$

Using a batch estimation mode, it can be shown that the CHMM parameters $\tilde{a}^{(c)}_{ij}$, $\tilde{u}^{(c)}_{jk}$, $\tilde{\mu}^{(c)}_{ijd}$, and $\tilde{\Sigma}^{(c)}_{ij}$ need to be updated using [18]:

$$\tilde{\Lambda}(\tau+1) = \tilde{\Lambda}(\tau) - \epsilon \nabla_{\tilde{\Lambda}} \mathbf{L}(\Lambda)\Big|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)}, \tag{19}$$

where

$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{a}_{ij}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta\, l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))$$
$$\delta(q_t^r = i, q_{t+1}^r = j)(1-a_{ij}^{(c)})\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)},$$
$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{u}_{ij}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta\, l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))u_{ij}^{(c)}$$
$$\times\,(1-u_{ij}^{(c)})\delta(q_t,i)\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)}\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)},$$
$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\mu}_{ijd}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta\, l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))$$
$$\times\,\sigma_{ijd}^{(c)}\delta(q_t,i)u_{ij}^{(c)}(o_{td}^{(r)}-\mu_{ijd}^{(c)})\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)}\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)},$$
$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\Sigma}_{ij}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta\, l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))$$
$$\times\left[\sigma_{ijd}^{(c)}\right]^{-1}\delta(q_t,i)u_{ij}^{(c)}$$
$$\times\left[(\mathbf{o}_{td}^{(r)}-\boldsymbol{\mu}_{ijd}^{(c)})\Sigma_{ij}^{-1}(\mathbf{o}_{td}^{(r)}-\boldsymbol{\mu}_{ijd}^{(c)})^t-1\right]$$
$$\times\,\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)}\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)},$$

In the above,

$$\frac{\partial d_c(O)}{\partial g_m(O,\Lambda)} = \begin{cases} -1 & \text{if } c = m \\ \frac{\exp[\eta g_c(O,\Lambda)]}{\sum_{j,j\neq c}\exp[\eta g_j(O,\Lambda)]} & \text{if } c \neq m \end{cases}. \tag{20}$$

2.2 HMM structures for multiple streams

For complex classification systems, data is usually gathered from multiple sources of information that have varying degrees of reliability. Within the context of hidden Markov models, different modalities could contribute to the generation of the sequence. These sources of information usually represent heterogeneous types of data. Assuming that the different sources are equally important in describing all the data might lead to suboptimal solutions.

Multi-modalities appear in several applications and could be broadly grouped into natural modalities and synthetic modalities. The first category consists of naturally available modalities such as audio and video used in automatic audio-visual speech recognition (AAVSR) systems [14]. Both speech and lips movement (possibly captured by video) are available when someone speaks. Natural modalities also appear in sign language recognition where multi-stream HMM, based on hand position and movement, has been used [23]. In the second category, the modalities are synthesized by several feature extraction techniques with different characteristics and expressiveness. For instance, for automatic speech recognition (ASR), Mel-frequency cepstral coefficients (MFCC) and formant-like features have been used as different sources within HMM classifiers [24]. Synthesized modalities have also been used to combine upper contour features and lower contour features as two streams for off-line handwritten word recognition [25].

Under the assumption of synchronicity and independence, the streams are handled using multi-stream HMM (MSHMM). MSHMM assumes that for each time slot, there is a single hidden state, from which different streams interpret the observations. The independence of the streams means that their interpretation of the hidden states and their generation of the observations is performed independently. Multi-stream HMM techniques have been proposed for both the discrete and the continuous case [15-17]. In our earlier study [17], we have proposed a multi-stream HMM framework for the discrete case that integrate a stream relevance weight for each symbol in each state, and we have generalized the BW and the MCE/GPD training algorithms for this structure.

For the continuous case, few types of MSHMM have been proposed in the literature to learn audio and visual stream relevance weights in speech recognition using continuous HMM [15,16]. In these methods, the feature space is partitioned into subspaces generated by the different streams, and different probability density functions (pdf) are learned for each subspace. The relevance weights for each stream could be fixed a priori by an expert [13], or learned via Minimum Classification Error/Generalized Probabilistic Descent (MCE/GPD) [16]. In [15], the authors have adapted the Baum-Welch algorithm [26] to learn the stream relevance weights. However, to derive the maximum likelihood equations, the model was restricted to include only one Gaussian component per state.

In the above approaches, the stream relevance weighting was introduced within the pdf characterizing the continuous HMM at the mixture level and at the state level. The mixture level weighting is based on factorizing each mixture into a product of weighted streams [16]. In particular, in [16] each component of the MFCC feature vector is considered as a separate stream. This is reflected on the observation probability as,

$$b_i(o_t) = \sum_{j=1}^{M} u_{ij}\prod_{k=1}^{L}\left[\phi(o_t^{(k)},\mu_{ijk},\Sigma_{ijk})\right]^{w_{ijk}}, \tag{21}$$

subject to

$$\sum_{j=1}^{M} u_{ij} = 1 \text{ and } \sum_{k=1}^{L} w_{ijk} = 1, \tag{22}$$

where w_{ijk} is the relevance weight of each stream k within component j of state i. It is learned via the minimum classification error (MCE) approach with generalized probabilistic descent (GPD) [16]. There is no method to learn the weights using the maximum likelihood (ML) approach. In the rest of the article, we refer to this method by MSCHMM^{G_M}.

On the other hand, the state level weighting treats the pdf as a product of exponent weighted mixture of Gaussians [27]. In [27], the streams are the audio and visual modalities of the speech signal, and the observation probability is given by

$$b_i(o_t) = \prod_{k=1}^{L} \left[\sum_{j=1}^{M} u_{ijk}\phi(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk}) \right]^{w_{ik}}, \tag{23}$$

subject to

$$\sum_{j=1}^{M} u_{ijk} = 1, \text{ and } \sum_{k=1}^{L} w_{ik} = 1, \tag{24}$$

where w_{ik} is the relevance weight of each stream k within state i. For this approach, it was shown [16] that it is not possible to derive an update equation for the exponent weights using maximum likelihood learning. As an alternative, in [28] the authors proposed an algorithm where these weights are learnt via the MCE/GPD approach while the remaining HMM parameters are estimated by means of traditional maximum likelihood techniques.

We should note here that, in general, (21) and (23) do not represent a probability distribution, and was therefore referred to as "score". In the rest of the article, we refer to this method by MSCHMM^{G_S}.

Even though existing MSCHMM structures provide a solution to combine multiple sources of information and were shown to outperform the baseline HMM, they are not general enough and they have several limitations. In particular, they do not provide an optimization framework that learns all the HMM parameters simultaneously. In general, a two step training approach is needed. First, the BW learning algorithm is used to learn the parameters of the HMM relative to each subspace. Then, the MCE/GPD algorithm is used to learn the relevance weights. This two-step approach is due to the difficulty that arises when using the proposed pdf within the BW learning algorithm. Consequently, the feature relevance weights learned with MCE/GPD may not correspond to local minima of the ML optimization. The only approach that extends the BW learning was derived for the special case that limits the number of components per state to one. This can be too restrictive for many applications.

To overcome the above limitations, we propose a generic approach that integrates stream discrimination within the CHMM classifier. In particular, we propose linear "scores" instead of the geometric ones in (21) and (23). We show that all parameters of the proposed model could be optimized simultaneously and we derive the necessary conditions to optimize them for both the MLE and MCE training approaches.

3 Multi-stream continuous HMM

We assume that, we have L streams of information. These streams could have been generated by different sensors and/or different feature extraction algorithms. Each stream is represented by a different subset of features. We propose two multi-stream continuous HMM (MSCHMM) structures that integrate stream relevance weights and alleviate the limitations of existing MSCHMM structures. In particular, we generalize the objective function to include stream relevance weights and derive the necessary conditions to update all parameters simultaneously. This is achieved by linearizing the "score" or the pdf approximate of the observation. We use the compact notation

$$\lambda = (\pi, \mathbf{A}, \mathbf{B}, \mathbf{W}), \tag{25}$$

to indicate the complete set of parameters of the proposed model. This includes the initial probabilities π, the transition probability $\mathbf{A}$, the observation probability distribution $\mathbf{B}$, and the stream relevance weights $\mathbf{W}$. The distributions π and $\mathbf{A}$ are defined in the same way as for the baseline CHMM. However, $\mathbf{B}$ and $\mathbf{W}$ are defined differently and depend on whether the streaming is at the mixture or at the state level.

In this article, we propose two forms of pdfs approximations. The first one is a mixture level streaming pdf that integrates local stream relevance weights that depend on the states and their mixture components. We will refer to this model as MSCHMM^{Lm}. The second version uses state level streaming pdf where the relevance weights depend only on the states. We will refer to this model as MSCHMM^{Ls}.

3.1 Multi-stream HMM with mixture level streaming

Let $\mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk})$ be a normal pdf with mean μ_{ijk} and covariance matrix Σ_{ijk} that represents the jth component in state i taking into account only the feature subset generated by stream k. Let w_{ijk} be the relevance weight of stream k in the jth component of state i. To cover the aggregate feature space generated by the L streams, we use a mixture of L normal pdfs, i.e.,

$$b_{ij}(o_t) = \sum_{k=1}^{L} w_{ijk} b_{ijk}(o_t^{(k)}) = \sum_{k=1}^{L} w_{ijk}\mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk}). \tag{26}$$

To model each state by multiple components, we let

$$b_i(o_t) = \sum_{j=1}^{M} u_{ij} b_{ij}(o_t), \tag{27}$$

subject to

$$\sum_{j=1}^{M} u_{ij} = 1, \text{ and } \sum_{k=1}^{L} w_{ijk} = 1. \tag{28}$$

In (27), u_{ij} is the mixing coefficient as defined in the standard CHMM (3). This linear form of the probability density function is motivated by the following probabilistic reasoning:

$$\begin{aligned} b_i(o_t) &= \Pr(o_t|q_t = i; \lambda) \\ &= \sum_{j=1}^{M} \Pr(o_t|q_t = i, e_t = j; \lambda) \Pr(e_t = j|q_t = i; \lambda) \end{aligned}$$

where e_t is a random variable representing the index of the component occurring at time t. By introducing a random variable, f_t, that represents the index of the most relevant stream at time t, we can rewrite $b_i(o_t)$ as:

$$\begin{aligned} b_i(o_t) = \sum_{j=1}^{M} \Pr(e_t = j|q_t = i; \lambda) \sum_{k=1}^{L} \Pr(o_t|q_t = i, e_t = j, f_t \\ = k; \lambda) \Pr(f_t = k|q_t = i, e_t = j; \lambda) \end{aligned}$$

If we assume that at time t one of the L streams is significantly more relevant than the others. In other words, the fusion of the L sources of information is performed in a mutual exclusive manner, and not in "collective" way where all the sources contribute (each with a small portion) to the characterization of the raw data. Then,

$$\begin{aligned} b_i(o_t) \approx \sum_{j=1}^{M} \Pr(e_t = j|q_t = i; \lambda) \sum_{k=1}^{L} \Pr(f_t = k|q_t = i, e_t \\ = j; \lambda) \Pr(o_t^{(k)}|q_t = i, e_t = j, f_t = k; \lambda) \end{aligned}$$

It follows then that:

$$\begin{aligned} b_{ijk}(o_t^{(k)}) &= \Pr(o_t^{(k)}|q_t = i, e_t = j, f_t = k; \lambda), \\ w_{ijk} &= \Pr(f_t = k|q_t = i, e_t = j; \lambda), \\ u_{ij} &= \Pr(e_t = j|q_t = i; \lambda). \end{aligned}$$

The MLE learning algorithm is an iterative approach that is prone to local minima. Therefore, it is important to provide good initial estimates of the parameters. For our approach, we propose the following initialization scheme. First, we use the SCAD algorithm [29] to cluster the training data into N_s clusters. The prototype of each of the N_s clusters is taken as the state representative vector. Next, we partition the observations assigned to each state cluster into M clusters to learn the M Gaussian components within each state. One advantage of using SCAD to perform this partitioning is that this algorithm learns feature relevance weights for each cluster. These relevance weights and the cardinality, mean, and covariance of each of the clusters are then used to initialize the MSCHMM parameters. After initialization, the model parameters are then tuned using the maximum Likelihood or the discriminative learning approaches. In the following, we generalize these learning methods for the proposed MSCHMM architectures.

3.1.1 Learning model parameters with generalized MLE

Given a sequence of training observation $O = [o_1, \ldots, o_T]$, the parameters of λ could be learned by maximizing the likelihood of the observation sequence O, i.e., $\Pr(O|\lambda)$. We achieve this by generalizing the continuous Baum-Welch algorithm to include a stream relevance weight component. We define the genera lized Baum-Welch algorithm through the following auxiliary function:

$$\mathbb{Q}(\lambda, \bar{\lambda}) = \sum_{Q} \sum_{E} \sum_{F} \Pr(Q, E, F|O, \lambda) \ln \Pr(O, Q, E, F|\bar{\lambda}), \tag{29}$$

where $E = [e_1, \ldots, e_T]$ and $F = [f_1, \ldots, f_T]$ are two sequences of random variables representing the component and stream indices at each time step. It can be shown that a critical point of $\Pr(O|\lambda)$, with respect to λ, is a critical point of the new auxiliary function $\mathbb{Q}(\lambda, \bar{\lambda})$ with respect to $\bar{\lambda}$ when $\bar{\lambda} = \lambda$, that is: $\frac{\partial \Pr(O|\lambda)}{\partial \lambda} = \frac{\partial \mathbb{Q}(\lambda, \bar{\lambda})}{\partial \bar{\lambda}}|_{\bar{\lambda}=\lambda}$. Maximizing the likelihood of the training data results in the following update equations (see Appendix 2):

$$w_{ij}^{(k)} = \frac{\sum_{t=1}^{T} \gamma_t^{(2)}(i,j,k)}{\sum_{t=1}^{T} \kappa_t(i,j)} \tag{30}$$

$$u_{ij} = \frac{\sum_{t=1}^{T} \kappa_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} \tag{31}$$

$$\mu_{ijkd} = \frac{\sum_{t=1}^{T} \nu_t(i,j,k) o_{td}^{(k)}}{\sum_{t=1}^{T} \nu_t(i,j,k)} \tag{32}$$

$$\Sigma_{ijk} = \frac{\sum_{t=1}^{T} \nu_t(i,j,k)(\mathbf{o}_{td}^{(k)} - \boldsymbol{\mu}_{ijkd})(\mathbf{o}_{td}^{(k)} - \boldsymbol{\mu}_{ijkd})^t}{\sum_{t=1}^{T} \gamma_t^{(2)}(i,j,k)}. \tag{33}$$

In the above,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N_s} \alpha_t(j)\beta_t(j)}, \tag{34}$$

$$\kappa_t(i,j) = \gamma_t(i) \frac{u_{ij} b_{ij}(o_t)}{b_i(o_t)}, \tag{35}$$

and

$$\nu_t(i,j,k) = \gamma_t(i) \frac{u_{ij} w_{ijk} \mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk})}{b_j(o_t)}. \tag{36}$$

In the case of multiple observations $[O^{(1)}, \dots, O^{(R)}]$, it can be shown that the update equations become:

$$w_{ijk} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(2)}(i,j,k)}{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(1)}(i,j)} \tag{37}$$

$$\mu_{ijkd} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(2)}(i,j,k)o_{td}^{(k)}}{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(2)}(i,j,k)} \tag{38}$$

$$\Sigma_{ijk} = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(2)}(i,j,k)(\mathbf{o}_{td}^{(k)} - \boldsymbol{\mu}_{ijkd})(\mathbf{o}_{td}^{(k)} - \boldsymbol{\mu}_{ijkd})^t}{\sum_{r=1}^{R}\sum_{t=1}^{T}\gamma_t^{(2)}(i,j,k)}. \tag{39}$$

Algorithm 1 outlines the steps of the proposed generalized BW algorithm to learn all of the MSCHMM^{Lm} parameters simultaneously.

Algorithm 1 Generalized BW training for the mixture level MSCHMM

```
Require:
 1: For each model $\lambda_c$, provide training data
    $[O_1, \dots, O_R]$, where $O_i = [o_1, \dots, o_T]$.
 2: Fix the variables $N_s$, $M$, and $L$.
Ensure:
 3: Cluster training data into $N_s$ subsets.
 4: Cluster each subset into $M$ clusters using
    SCAD and learn initial coefficients $u_{ij}$, stream
    relevance weights $w_{ijk}$, $\mu_{ijk}$, and $\Sigma_{ijk}$ for each
    cluster.
 5: while stopping criteria not satisfied do
 6:     Compute the probability density $b_i(o_t)$
    of the each observation vector $o_t$ using (27);
 7:     update A using (6);
 8:     update $u_{ij}$ using (7);
 9:     update $w_{ijk}$ using (37);
10:     update $\mu_{ijk}$ using (38);
11:     update $\Sigma_{ijk}$ using (39);
12: end while
```

3.1.2 Learning model parameters with generalized MCE/GPD

As an alternative training approach, we generalize the MCE/GPD to develop a discriminative training for the proposed MSCHMM^{Lm}. In particular, we extend the discriminant function in (10) to accommodate for the stream relevance weights using:

$$\begin{aligned} g_c(O,Q,\Lambda) = P(O,Q;\lambda_c) &= \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^{T} b_{q_t}^{(c)}(o_t) \\ &= \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^{T}\sum_{j=1}^{M} v_{q_t j}^{(c)} \sum_{k=1}^{L} w_{q_t jk}^{(c)} b_{q_t jk}^{(c)}(o_t) \end{aligned} \tag{40}$$

In the above, $b_{ijk}(o_t) = \mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk})$, where $\mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk})$ represents the normal density function with mean μ_{ijk} and covariance Σ_{ijk}. We assume that the covariance matrix Σ_{ijk} is diagonal. Hence, $\Sigma_{ijk} = \left[(\sigma_{ijkd})^2\right]_{d=1}^{p}$. Thus, $g_c(O,\Lambda) = \log[g_c(O,\bar{Q},\Lambda)]$, where $\bar{Q} = (\bar{q}_0, \bar{q}_1, \dots, \bar{q}_T)$ is the optimal state sequence that achieves $\max_q g_c(O,q,\Lambda)$, which could be computed using the Viterbi algorithm [22].

The misclassification measure of sequence O is defined by:

$$d_c(O) = -g_c(O,\Lambda) + \log\left[\frac{1}{1-C}\sum_{j,j\neq c}\exp[\eta g_j(O,\Lambda)]\right]^{\frac{1}{\eta}}, \tag{41}$$

where η is a positive number. A positive $d_c(O)$ implies misclassification and a negative $d_c(O)$ implies correct decision.

The misclassification measure is embedded in a smoothed zero-one function, referred to as loss function, defined as:

$$l_c(O,\Lambda) = l(d_c(O)), \tag{42}$$

where l is the sigmoid function in (14).

For an unknown sequence O, the classifier performance is measured by:

$$l(O;\Lambda) = \sum_{c=1}^{C} l_c(O;\Lambda)\mathbb{I}(O \in C_c) \tag{43}$$

where $\mathbb{I}(.)$ is the indicator function. Given a set of training observation sequences $O^{(r)}$, $r = 1, 2, \dots, R$, an empirical loss function on the training data, that can approximate the true Bayes risk is defined as:

$$\mathbf{L}(\Lambda) = \sum_{r=1}^{R}\sum_{c=1}^{C} l_c(O;\Lambda)\mathbb{I}(O \in C_c). \tag{44}$$

The MSCHMM^{Lm} parameters are estimated by applying a steepest descent optimization to $\mathbf{L}(\Lambda)$. In order to ensure that the estimated MSCHMM^{Lm} parameters satisfy the stochastic constraints, we map them using (17) and

$$w_{ijk} \to \tilde{w}_{ijk} = \log w_{ijk}. \tag{45}$$

Then, the parameters are updated with respect to $\tilde{\Lambda}$. After updating, we map them back using (18) and

$$w_{ijk} = \frac{\exp \tilde{w}_{ijk}}{\sum_{k'=1}^{L} \exp \tilde{w}_{ijk'}}. \tag{46}$$

Using a batch estimation mode, it can be shown that the MSCHMM^{Lm} parameters, $\tilde{u}_{ij}^{(c)}$, $\tilde{w}_{ijk}^{(c)}$, $\tilde{\mu}_{ijkd}^{(c)}$, and $\tilde{\sigma}_{ijkd}^{(c)}$ need to be updated iteratively using:

$$\tilde{\Lambda}(\tau+1) = \tilde{\Lambda}(\tau) - \epsilon \nabla_{\tilde{\Lambda}} \mathbf{L}(\Lambda)\Big|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)}, \tag{47}$$

where

$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{u}_{ij}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))u_{ij}^{(c)} (1-u_{ij}^{(c)})\delta(q_t,i)\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)}\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)} \tag{48}$$

$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda))w_{ijk}^{(c)} (1-w_{ijk}^{(c)})\delta(q_t,i)\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \times \frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)}, \tag{49}$$

$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\mu}_{ijkd}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda)) \times \sigma_{ijkd}^{(c)}\delta(q_t,i)u_{ij}^{(c)} (o_{td}^{(r)}-\mu_{ijkd}^{(c)})\frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \times \frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)}, \tag{50}$$

and

$$\frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\Sigma}_{ijk}^{(c)}} = \sum_{r=1}^{R}\sum_{m=1}^{C}\sum_{t=1}^{T} \zeta l_m(O_r,\Lambda)(1-l_m(O_r,\Lambda)) \times \left[\Sigma_{ijk}^{(c)}\right]^{-1}\delta(q_t,i)u_{ij}^{(c)} \times \left[(\mathbf{o}_t^{(r)}-\boldsymbol{\mu}_{ijk}^{(c)})\Sigma_{ijk}^{-1}(\mathbf{o}_t^{(r)}-\boldsymbol{\mu}_{ijk}^{(c)})^t - 1\right] \times \frac{b_{q_tj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)}\frac{\partial d_c(O_r)}{\partial g_m(O_r,\Lambda)}. \tag{51}$$

In the above, $\frac{\partial d_c(O)}{\partial g_m(O,\Lambda)}$ is as defined in (20). The update equation for $\tilde{a}_{ij}^{(c)}$ remains the same as that in given by (19).

Algorithm 2 outlines the steps needed to learn the parameters of all the models λ_c using the MCE/GPD framework.

Algorithm 2 MCE/GPD training of the mixture level MSCHMM

```
Require:
 1: Training data [$O_1,\dots,O_R$],
    $O_i = [o_1,\dots,o_T]$.
 2: Fix the variables $N_s$, $M$, and $L$ for each model
    $\lambda_c$.
Ensure:
 3: For each $\lambda_c$, cluster training data into $N_s$
    subsets.
 4: For each $\lambda_c$, cluster each subset into $M$
    clusters using SCAD and learn initial
    coefficients $u_{ij}$, stream relevance weights $w_{ijk}$.
    The center and covariance of each cluster
    initialize $\mu_{ijk}$ and $\Sigma_{ijk}$.
 5: while stopping criteria not satisfied do
 6:     for each model $\lambda_c$ do
 7:         Compute the probability density
    $b_i(o_t)$ of each observation vector $o_t$ using (27).
 8:         Compute the loss function of each
    sequence $O$ using (43);
 9:         update $\mathbf{A}$ using (47);
10:         update $u_{ij}$ using (47);
11:         update $w_{ijk}$ using (47);
12:         update $\mu_{ijk}$ using (47);
13:         update $\Sigma_{ijk}$ using (47);
14:     end for
15: end while
```

3.2 Multi-stream HMM with state level streaming

For the MSCHMM^{Ls} structure, we assume that the streaming is performed at the state level, i.e., each state is generated by L different streams, and each stream embodies M Gaussian components. Let b_{ik} be the probability density function of state i within stream k. Since stream k is modeled by a mixture of M components, b_{ik} can be written as:

$$b_{ik}(o_t^{(k)}) = \sum_{j=1}^{M} u_{ikj} b_{ikj}(o_t^{(k)}) = \sum_{j=1}^{M} u_{ikj}\mathcal{N}(o_t^{(k)},\mu_{ikj},\Sigma_{ikj}). \tag{52}$$

Let w_{ik} be the relevance weight of stream k in state i. The probability density function covering the entire feature space is then approximated by:

$$b_i(o_t) = \sum_{k=1}^{L} w_{ik} b_{ik}(o_t^{(k)}), \tag{53}$$

subject to

$$\sum_{k=1}^{L} w_{ik} = 1, \text{ and } \sum_{j=1}^{M} u_{ikj} = 1. \tag{54}$$

The linear form of the probability density function in (53) is motivated by the following probabilistic reasoning:

$$\begin{aligned} b_i(o_t) &= \Pr(o_t|q_t = i; \lambda) \\ &= \sum_{k=1}^{L} \Pr(o_t|q_t = i, f_t = k; \lambda)\Pr(f_t = k|q_t = i; \lambda) \end{aligned}$$

where f_t is a random variable representing the most relevant stream at time t. Similar to the component level case, we assumed that the fusion of the L sources of information is performed in a mutual exclusive manner. Hence, we have the following approximation:

$$\Pr(o_t|q_t = i, f_t = k; \lambda) = \Pr(o_t^{(k)}|q_t = i, f_t = k; \lambda)$$

It follows that:

$$\begin{aligned} b_i(o_t) &\approx \sum_{k=1}^{L} \Pr(o_t^{(k)}|q_t = i, f_t = k; \lambda)\Pr(f_t = k|q_t = i; \lambda) \\ &= \sum_{k=1}^{L} \Pr(f_t = k|q_t = i; \lambda) \sum_{j=1}^{M} \Pr(o_t^{(k)}|q_t = i, f_t = k, e_t \\ &\qquad = j; \lambda)\Pr(e_t = j|q_t = i, f_t = k; \lambda) \end{aligned}$$

where e_t and f_t a random variable that represents the index of the component that occurs at time t. It follows then that

$$\begin{aligned} b_{ikj}(o_t^{(k)}) &= \Pr(o_t^{(k)}|q_t = i, f_t = k, e_t = j; \lambda), \\ u_{ikj} &= \Pr(e_t = j|q_t = i, f_t = k; \lambda), \\ w_{ik} &= \Pr(f_t = k|q_t = i; \lambda). \end{aligned}$$

3.2.1 Learning model parameters with generalized MLE

Using similar steps to those used in the MSCHMM^{Lm}, it can be shown (see Appendix 2) that the model parameters need to be updated iteratively using:

$$w_{ik} = \frac{\sum_{t=1}^{T} \gamma_t^{(1)}(i,k)}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{55}$$

$$u_{ikj} = \frac{\sum_{t=1}^{T} \nu_t(i,k,j)}{\sum_{t=1}^{T} \kappa_t(i,k)}, \tag{56}$$

$$\mu_{ikjd} = \frac{\sum_{t=1}^{T} \nu_t(i,k,j) o_{td}^{(l)}}{\sum_{t=1}^{T} \nu_t(i,k,j)}, \tag{57}$$

$$\Sigma_{ikj} = \frac{\sum_{t=1}^{T} \nu_t(i,k,j)(\mathbf{o}_t^{(k)} - \boldsymbol{\mu}_{ikjd})(\mathbf{o}_{td}^{(k)} - \boldsymbol{\mu}_{ikjd})^t}{\sum_{t=1}^{T} \nu_t(i,k,j)}. \tag{58}$$

In the above,

$$\begin{aligned} \gamma_t(i) &= \Pr(q_t = i|O, \lambda) \\ \kappa_t(i,k) &= \gamma_t(i) \frac{w_{ik} b_{ik}(o_t)}{b_i(o_t)} \\ \nu_t(i,k,j) &= \gamma_t(i) \frac{w_{ik} u_{ikj} \mathcal{N}(o_t^{(k)}, \mu_{ikj}, \Sigma_{ijk})}{b_i(o_t)} \end{aligned} \tag{59}$$

The updating equation for a_{ij} remains the same as in standard Baum-Welch algorithm (i.e., as in (6)). In the case of multiple observations $[O^{(1)}, \ldots, O^{(R)}]$, it can be shown that the learning equations need to be updated using:

$$w_{ik} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \kappa_t^r(i,k)}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_t^r(i)}, \tag{60}$$

$$u_{ikj} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \nu_t^r(i,k,j)}{\sum_{r=1}^{R} \sum_{t=1}^{T} \kappa_t^r(i,k)}, \tag{61}$$

$$\mu_{ikjd} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \nu_t(i,k,j) o_{td}^{(k)(r)}}{\sum_{r=1}^{R} \sum_{t=1}^{T} \nu_t(i,k,j)}, \tag{62}$$

$$\Sigma_{ikj} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} \nu_t^r(i,k,j)(\mathbf{o}_t^{(k)(r)} - \boldsymbol{\mu}_{ikj})(\mathbf{o}_t^{(k)(r)} - \boldsymbol{\mu}_{ikjd})^t}{\sum_{r=1}^{R} \sum_{t=1}^{T} \nu_t^r(i,k,j)}. \tag{63}$$

Algorithm 3 outlines the steps of the MLE training procedure of the different parameters of the MSCHMM$^{\text{Ls}}$.

Algorithm 3 Generalized BW training for the state level MSCHMM

```
Require:
 1: Training data $[O_1, \ldots, O_R]$, where
    $O_i = [o_1, \ldots, o_T]$.
 2: Fix the variables $N_s$, $M$, and $L$.
Ensure:
 3: Clustering training data into $N_s$ clusters
    using SCAD, $w_i^{(k)}$ are then initialized
 4: Partitioning each state cluster into $M$
    clusters, using SCAD, and initialize the
    parameters, $\mu_{ij}^{(k)}$, $\Sigma_{ij}^{(k)}$ and $v_{ij}^{k}$.
 5: while stopping criteria not satisfied do
 6:     Compute the probability density $b_i(o_t)$
    of each observation vector $o_t$ (52).
 7:     update $\mathbf{A}$ using (6)
 8:     update $w_{ik}$ using (60)
 9:     update $u_{ikj}$ using (61)
10:     update $\mu_{ikjd}$ using (62)
11:     update $\Sigma_{ikj}$ using (63)
12: end while
```

3.2.2 Learning model parameters with generalized MCE/GPD

We generalize the MCE/GPD training approach for the MSCHMM^{Ls} by extending the discriminant function in (10) to accommodate for the stream relevance weights using:

$$\begin{aligned} g_c(O, Q, \Lambda) &= \Pr(O, Q; \lambda_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^{T} b_{q_t}^{(c)}(o_t) \\ &= \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^{T} \sum_{k=1}^{L} w_{q_t k}^{(c)} \sum_{j=1}^{M} v_{q_t kj}^{(c)} b_{q_t kj}^{(c)}(o_t) \end{aligned} \tag{64}$$

Defining the misclassification measure as in the component level streaming (Equation (41)) and following similar steps to minimize it, it can be shown that the MSCHMM^{Ls} parameters need to be updated iteratively using

$$\tilde{\Lambda}(\tau + 1) = \tilde{\Lambda}(\tau) - \epsilon \nabla_{\tilde{\Lambda}} \mathbf{L}(\Lambda)\Big|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)}, \tag{65}$$

where

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{a}_{ij}^{(c)}} &= \sum_{r=1}^{R} \sum_{m=1}^{C} \sum_{t=1}^{T} \zeta\, l_m(O_r, \Lambda)(1 - l_m(O_r, \Lambda)) \\ &\quad \times \delta(q_t^r = i, q_{t+1}^r = j)(1 - a_{ij}^{(c)}) \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}, \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ik}^{(c)}} &= \sum_{r=1}^{R} \sum_{m=1}^{C} \sum_{t=1}^{T} \zeta\, l_m(O_r, \Lambda)(1 - l_m(O_r, \Lambda)) w_{ik}^{(c)} \\ &\quad \times (1 - w_{ik}^{(c)}) \delta(q_t, i) \frac{b_{q_t k}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}, \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{u}_{ikj}^{(c)}} &= \sum_{r=1}^{R} \sum_{m=1}^{C} \sum_{t=1}^{T} \zeta\, l_m(O_r, \Lambda)(1 - l_m(O_r, \Lambda)) u_{ikj}^{(c)} \\ &\quad \times (1 - u_{ikj}^{(c)}) w_{ik}^{(c)} \delta(q_t, i) \frac{b_{q_t kj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}, \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\mu}_{ijd}^{(c)}} &= \sum_{r=1}^{R} \sum_{m=1}^{C} \sum_{t=1}^{T} \zeta\, l_m(O_r, \Lambda)(1 - l_m(O_r, \Lambda)) \\ &\quad \times \sigma_{ijd}^{(c)} \delta(q_t, i) w_{ik}^{(c)} u_{ikj}^{(c)} (o_{td}^{(r)} \\ &\quad - \mu_{ijd}^{(c)}) \frac{b_{q_t kj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}, \end{aligned}$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{\Sigma}_{ikj}^{(c)}} &= \sum_{r=1}^{R} \sum_{m=1}^{C} \sum_{t=1}^{T} \zeta\, l_m(O_r, \Lambda)(1 - l_m(O_r, \Lambda)) \left[\Sigma_{ikj}^{(c)}\right]^{-1} \\ &\quad \times \delta(q_t, i) w_{ik}^{(c)} u_{ikj}^{(c)} \left[(\mathbf{o}_t^{(r)} - \boldsymbol{\mu}_{ikj}^{(c)}) \Sigma_{ikj}^{-1} (\mathbf{o}_t^{(r)} - \boldsymbol{\mu}_{ikj}^{(c)})^t - 1\right] \\ &\quad \times \frac{b_{q_t kj}^{(c)}(o_t)}{b_{q_t}^{(c)}(o_t)} \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}, \end{aligned}$$

In the above, $\frac{\partial d_c(O)}{\partial g_m(O,\Lambda)}$ is as defined in (20). Algorithm 4 outlines the steps of the MCE/GPD training procedure for the different parameters of the MSCHMM^{Ls}.

Algorithm 4 MCE/GPD training of the state level MSCHMM

```
Require:
 1: Training data [$O_1, \ldots, O_R$],
    $O_i = [o_1, \ldots, o_T]$;
 2: Fix $N_s$, $M$ and $L$.
Ensure:
 3: Clustering training data into $N_s$ clusters
    using SCAD, $w_{ik}$ are then initialized
 4: Partitioning each state cluster into $M$
    clusters, using SCAD, and initialize the
    parameters,$u_{ikj}$, $\mu_{ikj}$, and $\Sigma_{ikj}$.
 5: while stopping criteria not satisfied do
 6:     Compute the probability density $b_i(o_t)$
    of each observation vector $o_t$ using (52).
 7:     Update $\mathbf{A}$ using (65)
 8:     Update $w_{ik}$ using (65)
 9:     Update $u_{ikj}$ using (65)
10:     Update $\mu_{ikjd}$ using (65)
11:     Update $\Sigma_{ikj}$ using (65)
12: end while
```

4 Experimental results

To illustrate the performance of the proposed MSCHMM architectures, we first use synthetically generated data sets to outline the advantages of the proposed structures and their learning algorithms. Then, we apply them to the problem of landmine detection using ground penetrating radar (GPR) sensor data.

4.1 Synthetic data

4.1.1 Data generation

We generate two synthetic data sets. The first one is a single stream sequential data, and the second is a multi-stream one. Both sets are generated using two continuous HMMs to simulate a two class problem. We follow an approach similar to the one used in [30] to generate sequential data using a continuous HMM with $N_s = 4$ states and $M = 3$ components per state with 4D. We start by fixing $\mu_k \in \mathbb{R}^4$, $k = 1, \ldots, N_s$ to represent the different states. Then, we randomly generate M vectors from each normal distribution, with mean μ_k and identity covariance matrix, to form the mixture components of each state. The mixture weights of the components within each state are randomly generated and then normalized. The covariance of each mixture component is set to the identity matrix. The initial state probability distribution and the state transition probability distribution are generated randomly from a uniform distribution in the interval $[0, 1]$. The randomly generated values are then scaled to satisfy the stochastic constraints. For more information about the data generation procedure, we refer the reader to [30].

For the single stream sequential data, we generate R sequences of length $T = 15$ vectors for each of the two classes. We start by generating a continuous HMM with N_s states and M components as described above. Then, we generate the single stream sequences using Algorithm 5.

Algorithm 5 Single stream sequential data generation for each class

```
for $r = 1$ to $R$ do
    Select an initial state according to the initial
    states probability distribution $\pi$
    Randomly pick a component $v$ from the $M$
    components of the selected state according
    to its mixture weights
    Sample an observation from a normal
    distribution with mean $v$ and covariance $\sigma I$
    for $t = 2$ to $T$ do
        Select next state according to the
        probabilities transition matrix $A$,
        Randomly pick a component $v$
        among those representing the
        selected state,
        Sample an observation $o_t$ from the
        normal distribution which mean $v$
        and covariance $\sigma I$.
    end for
end for
```

For the multi-stream case, we assume that the sequential data is synthesized by L=2 streams, and that each stream k is described by N_s states, where each state is represented by vector μ_n^k of dimension p_k=2. For each state i, three components are generated from each stream k, and concatenated to form a double-stream components. To simulate components with various relevance weights, we use a variation of three combinations of components in each state. The first combination concatenates a component from each stream by just appending the features (i.e., both streams are relevant). The second combination concatenates noise (instead of stream 2 features) to stream 1 (i.e., stream 1 is relevant and stream 2 is irrelevant). The last combination concatenates noise (instead of stream 1 features) to stream 2 (i.e., stream 1 is irrelevant and stream 2 is relevant). Thus, for each state i we have a set of double-stream components where the streams have different degrees of relevance. Once the set of double-stream components is generated, a state transition probability distribution is generated, and the double-stream sequential data is generated using Algorithm 5.

4.1.2 Results

First, we apply the baseline CHMM and the proposed multi-stream CHMM structures to the single stream sequential data where the features are generated from one homogeneous source of information. The MSCHMM architectures treat the single stream sequential data as a double-stream one (each stream is assumed to have 2D observation vectors). In this experiment all models are trained using standard Baum-Welch (for the baseline CHMM), generalized Baum-Welch (for the MSCHMM), standard and generalized MCE/GPD algorithms, or a combination of the two (Baum-Welch followed by MCE/GPD). The results of this experiment are reported in Table 1. As it can be seen, the performance of the proposed MSCHMM structures and the baseline CHMM are comparable for most training methods. This is because when both streams are equally relevant for the entire data, the different streams receive nearly equal weights in all states' components and the MSCHMM reduces to the baseline CHMM. Figure 1 displays the weights of stream 1 components. As it can be seen, most weights are clustered around 0.5 (maximum weight is less than 0.6 and minimum weight is more than 0.4). Since

Table 1 Classification rates of the different CHMM structures of the single stream data

Classifier	Baum-Welch	MCE	BW and MCE
Baseline CHMM	89.00 %	91.25 %	93.15 %
MSCHMM^{Lm}	93.25 %	94.00 %	95.00 %
MSCHMM^{Ls}	92.75 %	95.25 %	97.45 %

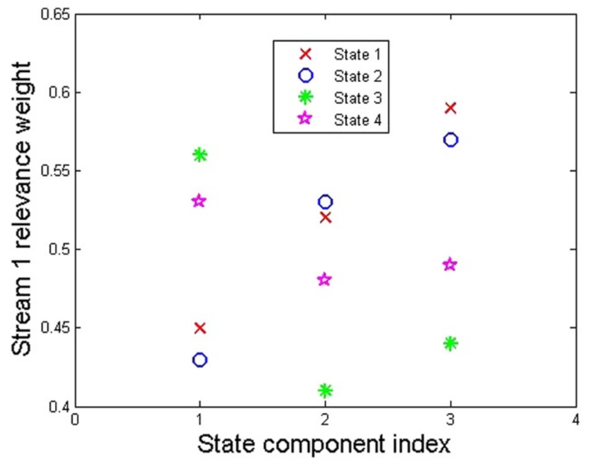

Figure 1 Stream 1 relevance weights of the mixture components in all four states, learned by the MSCHMM^{Lm} model for the single-stream sequential data.

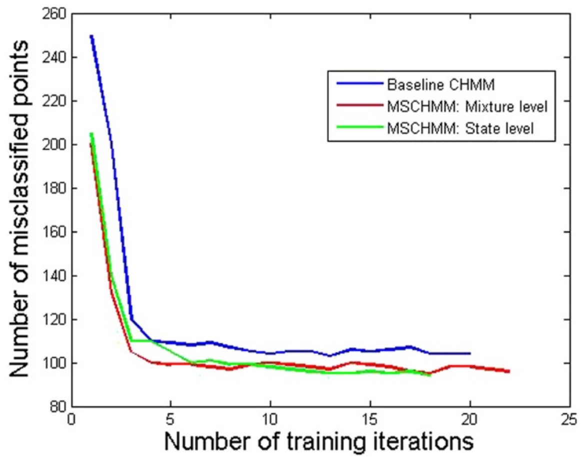

Figure 2 Number of misclassified samples versus training iteration number for the standard and multistream CHMMs.

weights of both streams must sum to 1, both weights are equally important for all symbols.

The second experiment involves applying both the baseline CHMM and the proposed MSCHMM to the double stream sequential data where the features are generated from two different streams. In this experiment, the various models are trained using Baum-Welch, MCE, and Baum-Welch followed by MCE training algorithms. First, we note that using stream relevance weights, the generalized Baum-Welch and MCE training algorithms converge faster and the MCE results in smaller training error. Figure 2 displays the number of misclassified samples versus the number of iterations for the baseline CHMM and the proposed MSCHMM using MCE/GPD training. As it can be seen, learning stream relevance weights causes the error to drop faster. In fact, at each iteration, the classification error for the MSCHMM is lower than that of the baseline CHMM. However, as shown in Table 2, for each iteration, the computational complexity involved in the proposed MSCHMM is about 2.5 times of the baseline CHMM.

The testing results are reported in Table 3. First, we note that all proposed multi-stream CHMMs outperform the baseline CHMM for all training methods. This is because the data set used for this experiment was generated from two streams with different degrees of relevance and the baseline CHMM treats both streams equally important. The proposed MSCHMM structures on the other hand, learn the optimal relevance weights for each symbol within each state. The learned weights for stream 1 by the MSCHMM^{Lm} are displayed in Figure 3. As it can be seen, some components are highly relevant (weight close to 1) in some states, while others are completely irrelevant (weights close to 0). The latter ones correspond to the components where stream 1 features were replaced by noise in the data generation. We should note here that in theory, we assumed that at time t one of the L streams is significantly more relevant than the others in order to derive update equations for all parameters using the Baum-Welch algorithm (refer to Section 3.1). However, in practice, the performance of the algorithm does not break down if this assumption does not hold. For instance, in Figure 1, the weights are equal when all streams are relevant while in Figure 3 the weights are different but not binary.

In Table 3, we also compare our approach to the two state of the art MSCHMM that were discussed in Section 2.2. The proposed multi-stream CHMMs outperform both of these methods. This is mainly due to the fact that the parameters of the proposed MSCHMM structures allow for a simultaneous update for both Baum-Welch and MCE/GPD training. However, for the MSCHMM^G, the parameters learned separately by two different algorithms and two different objective functions.

From Table 3, we also notice that using the generalized Baum-Welch followed by the MCE to learn the model parameters is a better strategy. This is consistent with what has been reported for the baseline HMM [18].

4.2 Application to landmine detection

4.2.1 Data collection

We apply the proposed multi-stream CHMM structures to the problem of detecting buried landmines. We use data collected using a robotic mine detection system. This system includes a ground penetrating radar (GPR) and a Wideband Electro-Magnetic Induction (WEMI) sensor and is shown in Figure 4. Each sensor collects data as the system moves. Only data collected by the GPR sensor

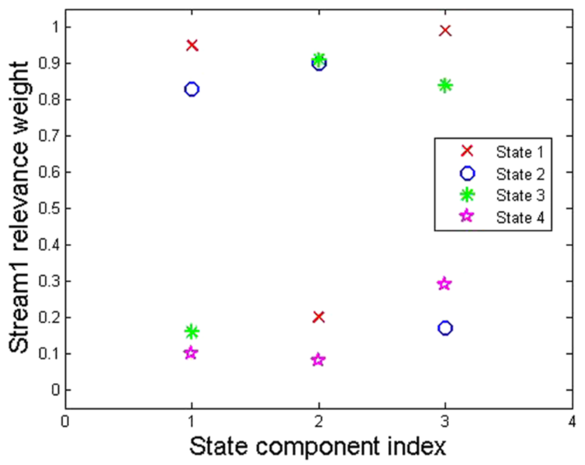

Figure 3 Stream 1 relevance weights of the mixture components in all four states learned by the MSCHMM^{Lm} model for the double-stream sequential data.

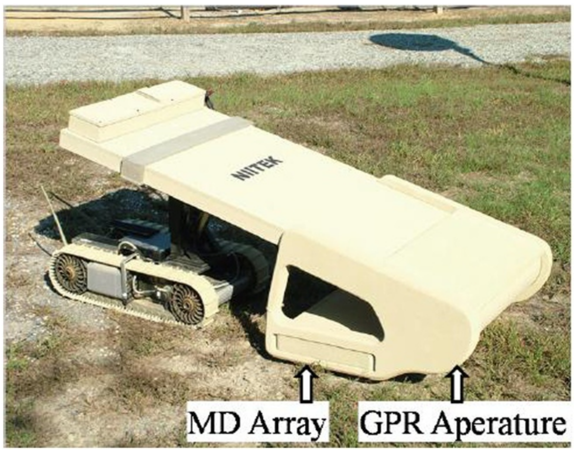

Figure 4 NIITEK autonomous mine detection system.

is used in our experiments. The GPR sensor [31] collects 24 channels of data. Adjacent channels are spaced approximately 5 cm apart in the cross-track direction, and sequences (or scans) are taken at approximately 1 centimeter down-track intervals. The system uses a V-dipole antenna that generates a wide-band pulse ranging from 200 MHz to 7 GHz. Each A-scan, that is, the measured waveform collected in one channel at one down-track position, contains 516 time samples at which the GPR signal return is recorded. We model an entire collection of input data as a 3D matrix of sample values, $S(z, x, y)$; $z = 1, \ldots, 516; x = 1, \ldots, 24; y = 1, \ldots, T$, where T is the total number of collected scans, and the indices z, x, and y represent depth, cross-track position, and down-track positions, respectively.

The autonomous mine detection system (shown in Figure 4) was used to acquire large collections of GPR data from two geographically distinct test sites in the eastern U.S. with natural soil. The two sites are partitioned into grids with known mine locations. Twenty eight distinct mine types that can be classified into four categories: anti-tank metal (ATM), anti-tank with low metal content (ATLM), anti-personal metal (APM), and anti-personal with low metal content (APLM) were used. All targets were buried up to 5 inches deep. Multiple data collections were performed at each site resulting in a large and diverse collection of signatures. In addition to mines, clutter signatures were used to test the robustness of the detectors. Clutter arises from two different processes. One type of clutter is emplaced and surveyed. Objects used for this clutter can be classified into two categories: high metal clutter (HMC) and non-metal clutter (NMC). High metal clutter such as steel scraps, bolts, soft-drink cans, was emplaced and surveyed in an effort to test the robustness of the detection algorithms. Non-metal clutter such as concrete blocks and wood blocks was emplaced and surveyed in an effort to test the robustness of the GPR based detection algorithms. The other type of clutter, referred to as blank, is caused by disturbing the soil.

For our experiment, we use a subset of the data collection that includes 600 mine and 600 clutter signatures. The raw GPR data are first preprocessed to enhance the mine signatures for detection. We identify the location of the ground bounce as the signal's peak and align the multiple signals with respect to their peaks. This alignment is necessary because the mounted system cannot maintain the radar antenna at a fixed distance above the ground. Since the system is looking for buried objects, the early time samples of each signal, up to few samples beyond the ground bounce are discarded so that only data corresponding to regions below the ground surface are processed.

Figure 5 displays several preprocessed B-scans (sequences of A-scans) both down-track (formed from a time sequence of A-scans from a single sensor channel) and cross-track (formed from each channels response in a single sample) at the position indicated by a line in the down-track. The objects scanned are (a) a high-metal

Table 2 CPU time per iteration for the MCE/GPD training

MCE/GPD	Baseline CHMM	MSCHMM^{Lm}	MSCHMM^{Ls}
Time per iteration	1.27 s	3.5 s	3.8 s

Table 3 Performance of the different CHMM structures on the multi-stream data

Classifier	Baum-Welch	MCE	BW and MCE
Baseline CHMM	63.25 %	65.75 %	68.85 %
MSCHMM^{Lm}	70.35 %	72.75 %	79.65 %
MSCHMM^{Ls}	71.65 %	71.25 %	80.00 %
MSCHMM^{G_M}	-	-	70.65 %
MSCHMM^{G_S}	-	-	72.00 %

content anti-tank mine, (b) a high-metal content anti-personnel mine, and (c) a wood block. The reflections between depths 50 and 125 in these figures are the artifact of preprocessing and data alignment. The strong reflections between cross-track scans 15 and 20 are due to Electromagnetic interference (or EMI). The preprocessing artifacts and the EMI can add considerable amounts of noise to the signatures and make the detection problem more difficult.

4.2.2 *Feature extraction*

As it can be seen in Figure 6, landmines (and other buried objects) appear in time domain GPR as hyperbolic shapes (corrupted by noise), usually preceded and followed by a background area. Thus, the feature representation adopted by the HMM is based on the degree to which edges occur in the diagonal and antidiagonal directions, and the features are extracted to accentuate these edges.

Each alarm has over 516 depth values, however, the mine signature is not expected to cover all the depth values. Typically, depending on the mine type and burial depth, the mine signature may extend over 40–200 depth values, i.e., it may cover no more than 10% of the extracted data cube. For example, in Figure 5b, the signature essentially extends from depth index 170 to depth index 200. There is a little or no evidence that a mine is present in depth bins above or below this region. Thus, extracting one global feature from the alarm may not discriminate between mine and clutter signatures effectively. To avoid this limitation, we extract the features from a small window with $W_d = 45$ depth values. Since the ground truth for the depth value (z_s) is not provided, we visually inspect all training mine signatures and estimate this value. For

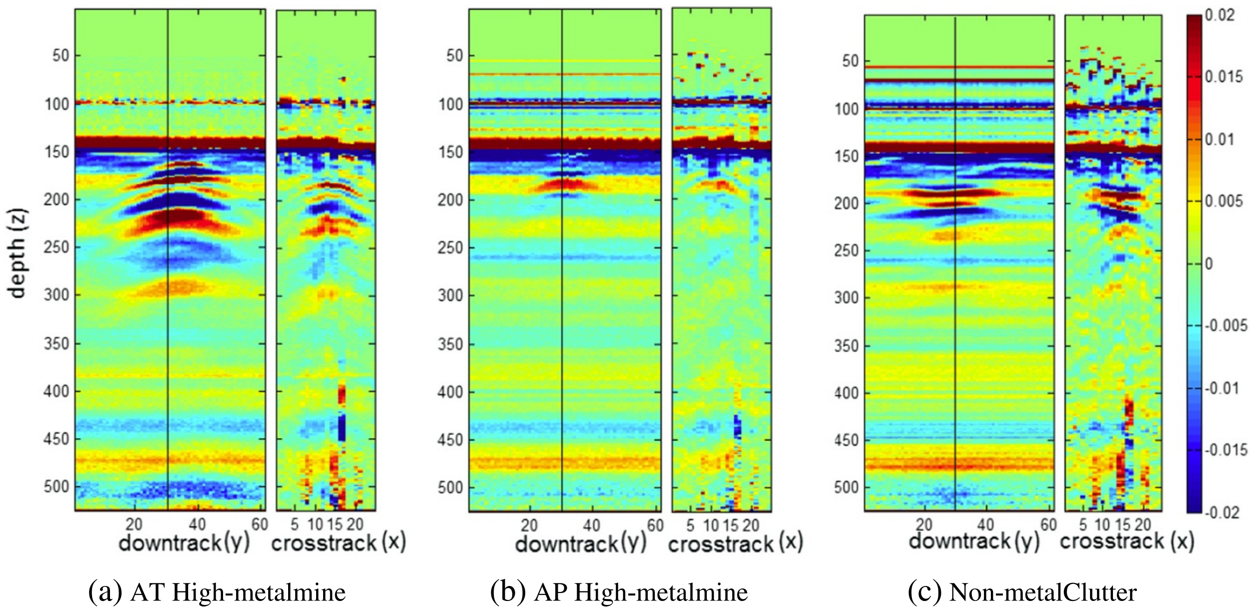

Figure 5 NIITEK radar down-track and cross-track (at position indicated by a line in the down-track) B-scans pairs for (a) an anti-tank (AT) mine, (b) an anti-personnel (AP) mine, and (c) a non-metal clutter alarm.

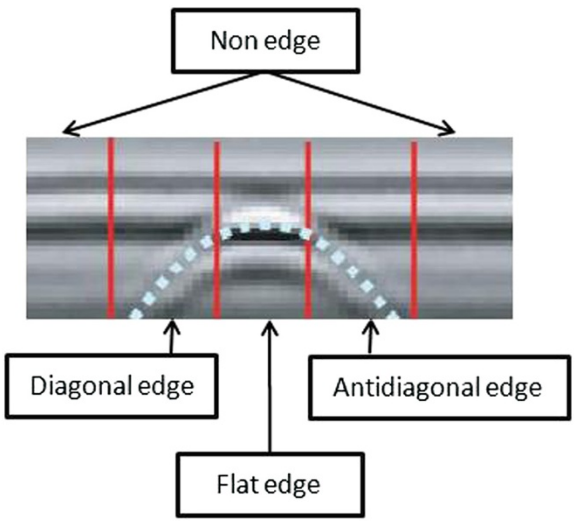

Figure 6 Shape of a typical mine signature and the interpretation of the four states of the DHMM structure.

the clutter signatures, this process is not trivial as clutter objects can have different characteristics and their signature can extend over a different number of samples. Instead, for each clutter signature, we extract five training signatures at equally spaced depths covering the entire depth range. Also, out of the 24 GPR channels, we process only the middle 7 channels as it is unlikely that the target signatures extend beyond this range. Thus, each training signature s consists of 45(depth)×15(scans)×7(channels) volume extracted from the aligned GPR data.

Figure 6 displays a hyperbolic curve superimposed on a preprocessed mine signature (only 45 depths) to illustrate the features of a typical mine signature. This figure also justifies the choice of $N_s = 4$ states in the adopted CHMM structure. State 1 corresponds to non-edge activity (i.e., background), state 2 corresponds to diagonal edge, state 3 corresponds to a flat edge, and state 4 corresponds to an anti-diagonal edge.

We adopt the Homogeneous Texture Descriptor [32] to capture the spatial distribution of the edges within the 3D GPR alarms. We extract features by expanding the signature's B-scan using a bank of Gabor filters at 4 scales and 4 orientations. Let $S(x, y, z)$ denotes the 3D GPR data volume of an alarm. To keep the computation simple, we use 2D filters (in the $y - z$ plane) and average the response over the third dimension. Let $S_x(y, z)$ be the xth plane of the 3D signature $S(x, y, z)$. Let $SG_x^{(k)}(y, z)$, $k = 1, \ldots, 16$ denotes the response of $S_x(y, z)$ to the 16 Gabor filters. Figure 7 displays a strong signature of a typical metal mine and its response to the 16 Gabor filters. As it can be seen, the signature has a strong response to the θ_2 (45°) filters (especially scale 1 and scale 2 to a lesser degree) on the left part of the signature (rising edge), and a strong response to the θ_4 (135°) filters on the right part of the signature (falling edge). Similarly, the middle of signature has a strong response to the θ_3 (horizontal) filters (flat edge). Figure 7b displays a weak mine signature and its response to the Gabor filters. For this signature, the edges are not as strong as those in Figure 7a. As a result, it has a weaker response at all scales (scale 2 has the strongest response), especially for the falling edge. Figure 7c displays a clutter signature (with high energy) and its response. As it can be seen, this signature has strong response to the θ_4 (135°) degree filters. However, this response is not localized on the right side of the signatures.

In our HMM models, we take the down-track dimension as the time variable (i.e., y corresponds to time in the HMM model). Our goal is to produce a confidence that a mine is present at various positions, (x, y), on the surface being traversed. To fit into the HMM context, a sequence of observation vectors must be produced for each signature. We define the observation sequence of $S_x(y, z)$, at a fixed depth z, the sequence

$$[\,\mathbf{O}(x, y-7, z), \mathbf{O}(x, y-6, z), \ldots, \mathbf{O}(x, y-1, z), \mathbf{O}(x, y, z), \mathbf{O}(x, y+1, z), \ldots, \mathbf{O}(x, y+7, z)\,], \tag{66}$$

where

$$\mathbf{O}(x, y, z) = [\,O^1(x, y, z), \ldots, O^{16}(x, y, z)\,], \tag{67}$$

and

$$O^k(x, y, z) = \frac{1}{45}\sum_{z=1}^{45} SG_x^{(k)}(y, z), \tag{68}$$

encodes the response of $S(x, y, z)$ to the kth Gabor filters.

4.2.3 Learning HMM parameters

We construct and train multiple landmine detectors using the proposed HMM structures. Each detector has one model for background (learned using non-mine training signatures) and another for mine (learned using trained mine signatures). Each model produces a probability value by backtracking through model states using the Viterbi algorithm. The probability value produced by the mine (background) model can be thought of as an estimate of the probability of the observation sequence given that there is a mine (background) present.

For all CHMM structures, we assume that each model has $N_s = 4$ states. The states representatives, v_k, are obtained by clustering the training data into four clusters using Fuzzy C-Means [33]. The learning procedures used for the other parameters depend on the HMM structures and are outlined below.

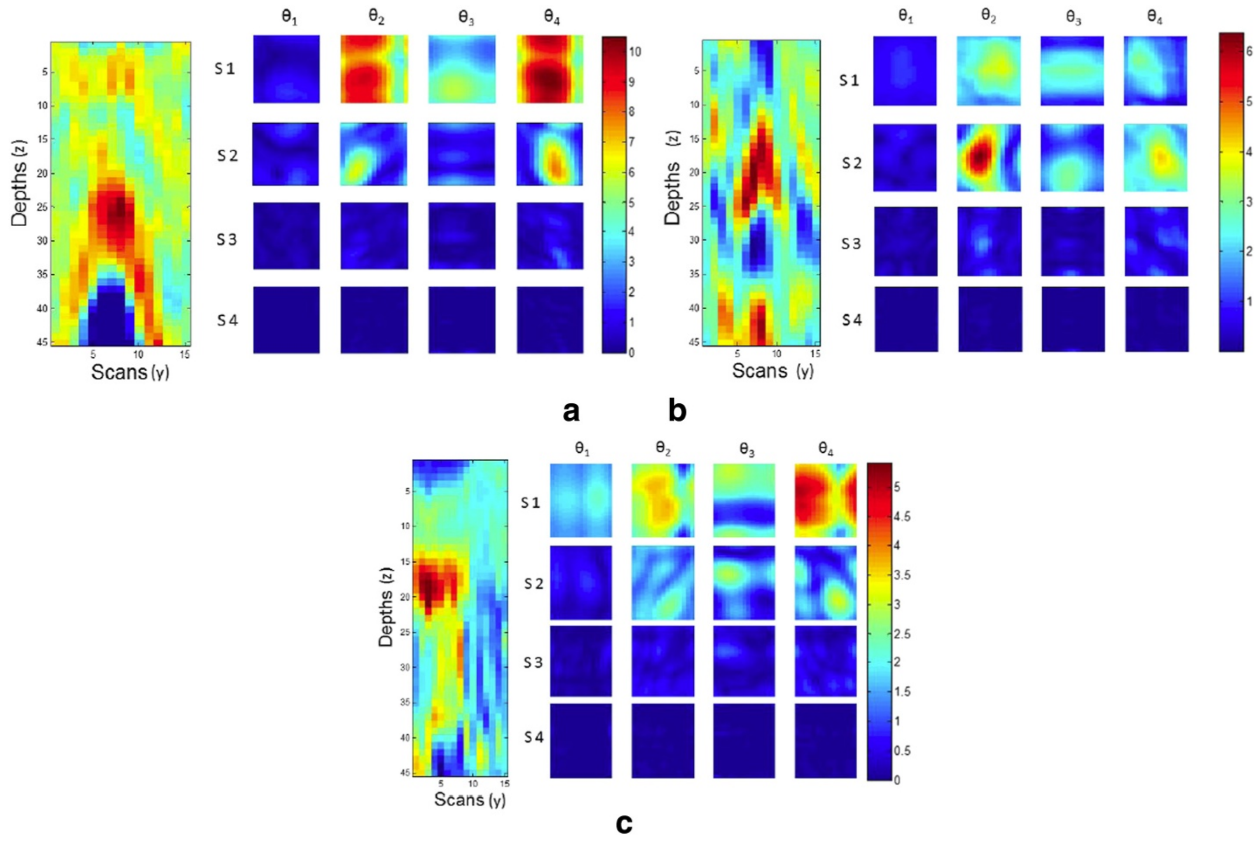

Figure 7 Response of three alarms to the 16 Gabor filters at different scales and orientations. (a) Strong mine signature, **(b)** weak mine signature, and **(c)** clutter signature with high energy.

Baseline (single stream) CHMM For the baseline CHMM, we treat all features (responses of the 16 Gabor filters) equally important. To generate the state components, we cluster the training data relative to each state into $M = 4$ clusters using FCM algorithm [33]. The transition probabilities **A**, the mixing coefficients **U**, and the component parameters could be estimated using Baum-Welch algorithm [1], the MCE/GPD algorithm [18], or few iteration of Baum-Welch followed by the MCE/GPD algorithm. Our results have indicated that the combination of the two learning algorithms provides the best classification accuracy. Thus, due to the space constraint, only those results are reported in this article.

Multi-stream CHMM The Gabor features used within the baseline continuous HMM assume that all scales and orientations contribute equally in characterizing alarm signatures. However, this assumption may not be valid for most cases. For instance, some alarms may be better characterized at a lower scale, while others may be better characterized at a higher scale. The different scales could then be treated as different sources of information, i.e., different streams.

Since it is not possible to know a priori which scale is more discriminative, we propose considering the different Gabor scales as different streams of information and use the training data to learn multi-stream CHMMs (mixture and state level). Thus, we use four streams where each stream (Gabor response at a fixed scale) produces a 4D feature vectors (Gabor response at the different orientations). To generate the state components, we cluster the training data relative to each state in $M = 4$ clusters using SCAD [29] and learn initial stream relevance weights for each state and component. The state transition probabilities **A**, the mixing coefficients **U**, and the component parameters and the observation probabilities **B** are learned using the generalized Baum-Welch (see Sections 3.1.1 and 3.2.1), the generalized MCE/GPD (see Sections 3.1.2 and 3.2.2), or a combination of the two.

4.2.4 Confidence value assignment

The confidence value assigned to each observation sequence, Conf(O), depends on: (1) the probability assigned by the mine model (λ^m), $\Pr(O|\lambda^m)$; (2) the probability assigned by the background model (λ^c), $\Pr(O|\lambda^c)$; and (3) the optimal state sequence. In particular, we use:

$$\mathrm{Conf}(O) = \begin{cases} \max\left(\log \frac{\Pr(O|\lambda^m)}{\Pr(O|\lambda^c)}, 0\right) & \text{if } \#\{s_t = 1, t = 1, \ldots, T\} \leq T_{\max} \\ 0 & \text{otherwise} \end{cases} \quad (69)$$

Since each alarm has over 300 depth values (after preprocessing) and only 45 depths are processed at a time, we divide the test alarm into 10 overlapping sub-alarms and test each one independently to obtain 10 partial confidence values. These values could be combined using various fusion methods such as averaging, artificial neural networks [34], or an order-weighted average (OWA) [35]. In this article, we report the results using the average of the top three confidences. This simple approach has been successfully used in [36].

4.2.5 *Experimental results*

We use a 5-fold cross validation scheme to evaluate the proposed MSCHMM structures and compare them to the baseline CHMM and to MSCHMM^G (Section 2.2). For each cross-validation, we use a different subset of the data that has 80% of the alarms for training and test on the remaining 20% of the alarms. The scoring is performed in terms of probability of detection (PD) versus probability of false alarms (PFA). Confidence values are thresholded at different levels to produce the receiver operating characteristics (ROC) curve.

Figure 8 compares the ROC curves generated using each of the four streams (Gabor features at each scale) and their combination using simple concatenation (Baseline CHMM), using the proposed MSCHMM and MSCHMM^G (Section 2.2). We only display the ROC segments where the PD is larger than 0.5 to magnify the interesting and practical regions. All results were obtained when the model parameters are learned using Baum-Welch followed by the MCE/GPD training method. First, we note that the CHMM with Gabor features at scale 2 and 4 outperform all other features (for FAR $\leq$ 40). Second, the baseline CHMM with all 4 scales is not much better than the CHMM at scale 2 and 4 especially for FAR $\leq$ 30. In fact, for some FAR, the performance can be worse. This is due mainly to the way the four scales are combined equally. Third, we note that all MSCHMM structures outperform the baseline CHMM. Moreover, the MSCHMM with mixture level streaming outperforms the other structures. Fourth, the proposed MSCHMM structures outperform the MSCHMM^G (Section 2.2). This is due to the fact that for the latter approach, the stream relevance weights are learned separately from the rest of the model parameters. These results are consistent with those obtained with the synthetic data in Table 3. Figure 8 also compares the performance of the proposed continuous MSCHMM structures with our previously published discrete version [17]. As expected with most HMM classifiers, the continuous versions have slightly better performance.

To illustrate the advantages of combining the different Gabor scales into a MSCHMM structure and learning stream dependent relevance weights, in Figure 9, we display a scatter plot of the confidence values generated by the baseline CHMM that uses Gabor feature at scale 1 and scale 2, separately. As it can be seen, for many alarms, the confidence values generated by both CHMMs are comparable (i.e., alarms along the diagonal). However, there are different regions in the confidence space where one scale is more reliable than the other. For instance, alarms highlighted in region R_3 include more mine signatures than false alarms, and these signatures have higher confidence values using scale 1. Thus, for this region, scale 1 is a better detector than scale 2. The alarm shown in Figure 7a is one of those alarms, and as it can be seen, the alarm's response to scale 1 Gabor filters is more dominant. Similarly, region R_1 include mainly mine signatures that have high confidence values using scale 1 and low confidence values using scale 1. Thus, for this group of alarms, the scale 2 detector is more reliable than scale 1 detector. The alarm shown in Figure 7b is one of those alarms and has a stronger response to scale 2. This difference in behavior exists for both target and non-target alarms. For instance, region R_2 highlights both target and non-target alarms that are detected at scale 2 but not detected at scale 1 using an 80 % PD threshold (=4.2).

5 Conclusions

We have proposed novel multi-stream continuous Hidden Markov models structures that integrate stream relevance weighting component for the classification of temporal data. These structures allow learning component or state dependent stream relevance weights. In particular, we modified the probability density function that characterizes the standard continuous HMM to include state and component dependent stream relevance weights. For both methods, we generalized the Baum-Welch and MCE/GPD learning algorithms and derived the update equations for all model parameters are derived. Results on synthetic data set and a library of GPR signatures show that the proposed multi-stream CHMM structures improve the discriminative power and thus, the classification accuracy of the CHMM. The introduction of stream relevance weights also causes the training error to decrease faster and for the training algorithm to converge faster.

The discriminative training performed in this article uses batch mode training. Sequential training could be

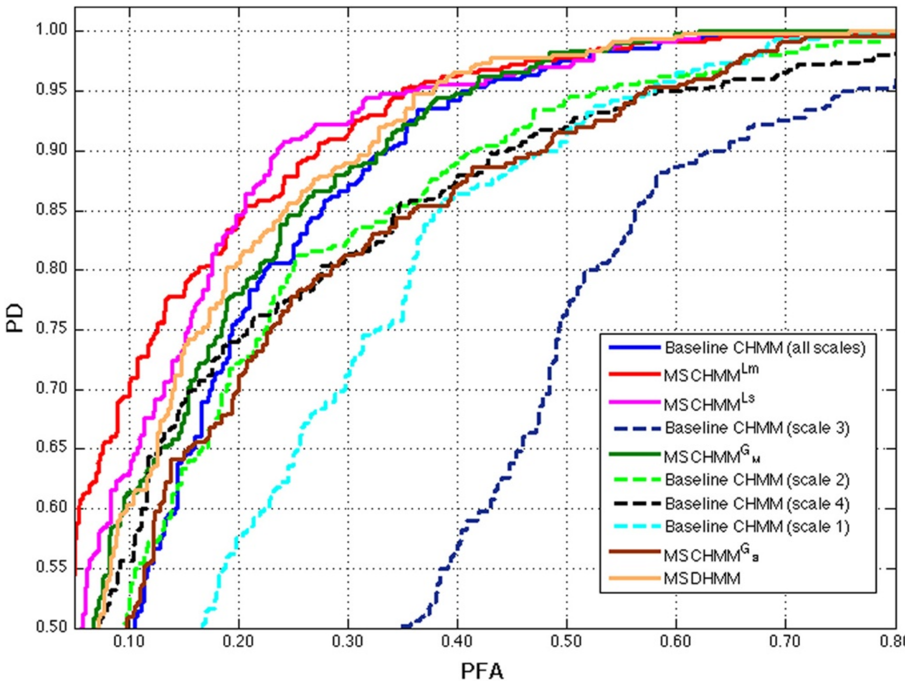

Figure 8 Performance of the proposed multi-stream CHMM compared to the baseline CHMM and the state of the art MSCHMM when the PD >= 0.5.

investigated and combined with a boosting framework. In order to control the complexity of the proposed structures, a regularization mechanism could be investigated. In addition, this study could be extended to the Bayesian case that is relevant in situations where training data is limited. The application to landmine detection could be extended to include streams from different feature extraction methods or even from different sensors.

Appendix 1

Generalized Baum-Welch for the mixture level MSCHMM

The objective function in (29) involves the quantity $\Pr(O, Q, E, F|\bar{\lambda})$ which could be expressed analytically as:

$$\Pr(O, Q, E, F|\bar{\lambda}_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a^{(c)}_{q_t q_{t+1}} \prod_{t=1}^{T} u^{(c)}_{q_t e_t} w^{(c)}_{q_t e_t f_t} b^{(c)}_{q_t e_t f_t}(o_t) \tag{70}$$

Thus, the objective function in (29) can be expanded as follows:

$$\begin{aligned}\mathbb{Q}(\lambda, \bar{\lambda}) =& \sum_{Q}\sum_{E}\sum_{F} \Pr(Q, E, F|O, \lambda) \log \bar{\pi}_{q_1} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{E}\sum_{F} \Pr(Q, E, F|O, \lambda) \log \bar{a}_{q_t q_{t+1}} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{E}\sum_{F} \Pr(Q, E, F|O, \lambda) \log \bar{u}_{q_t e_t} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{E}\sum_{F} \Pr(Q, E, F|O, \lambda) \log \bar{w}_{q_t e_t f_t} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{E}\sum_{F} \Pr(Q,E,F|O,\lambda) \log \mathcal{N}(o_t^{(f_t)}, \bar{\mu}_{q_t e_t f_t}, \bar{\Sigma}_{q_t e_t f_t})\end{aligned} \tag{71}$$

After the estimation step, the maximization step consists of finding the parameters of $\bar{\lambda}$ that maximize the function in (71). The expanded form of the function $\mathbb{Q}(\lambda, \bar{\lambda})$ in (71) has 5 terms involving $\bar{\pi}$, $\bar{a}$,and $(\bar{w}, \bar{b})$ independently. To find the values of $\bar{\pi}_i$, $\bar{a}_{ij}$, $\bar{w}_{ijk}$, and $\bar{b}_{ijk}$ that maximize $\mathbb{Q}(\lambda, \bar{\lambda})$, we consider the terms in (71) that depend on $\bar{\pi}$, $\bar{a}$, $\bar{w}$, and $\bar{b}$. In particular, the first and second terms in (71) depend on $\bar{\pi}$ and $\bar{a}$, and they have the same analytical expressions sketched in the case of the baseline CHMM (refer to (2.1)). It follows that the update equations for $\bar{\pi}_i$, $\bar{a}_{ij}$, and $\bar{u}_{ij}$ are the same as in the standard CHMM. That is,

$$\bar{\pi}_i = \gamma_1(i),$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)},$$

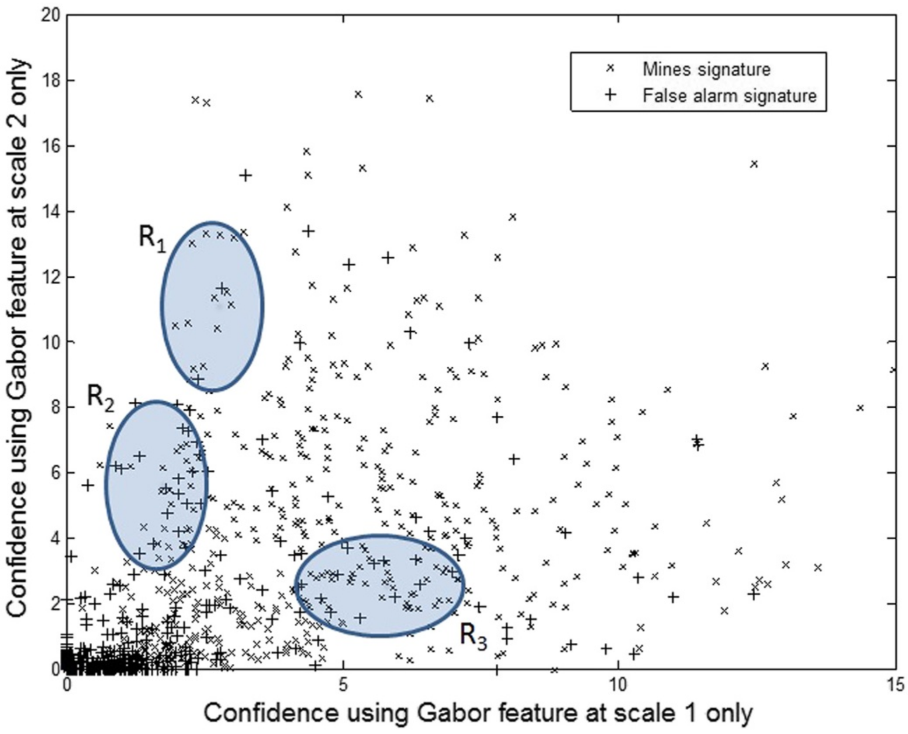

Figure 9 **Scatter plot of the confidence values generated using two baseline DHMM that use Gabor features at scales 1 and 2.**

and

$$\bar{u}_{ij} = \frac{\sum_{t=1}^{T} \Pr(q_t = i, e_t = j|o, \lambda)}{\sum_{t=1}^{T} \Pr(q_t = i|o, \lambda)}.$$

To find the value of $\overline{w}_{ijk}$ that maximizes the auxiliary function $\mathbb{Q}(.,.)$, only the fourth term of the expression in (71) is considered since it is the only part of $\mathbb{Q}(.,.)$ that depends on $\overline{w}_{ijk}$. This term can be expressed as:

$$\sum_{t=1}^{T}\sum_{Q}\sum_{E}\sum_{F}\Pr(Q,E,F|O,\lambda)\log\bar{w}_{q_te_tf_t} = \sum_{t=1}^{T}\sum_{i}\sum_{j}\sum_{k}\log(\bar{w}_{ijk}) \times\sum_{Q}\sum_{E}\sum_{F}\Pr(Q,E,F|O,\lambda)\delta(i,q_t)\delta(j,e_t)\delta(k,f_t), \tag{72}$$

where $\delta(i,q_t)\delta(j,e_t)\delta(k,f_t)$ keeps only those cases for which $q_t = i$, $e_t = j$ and $f_t = k$. That is,

$$\sum_{Q}\sum_{E}\sum_{F}\Pr(Q,E,F|O,\lambda)\delta(i,q_t)\delta(j,e_t)\delta(k,f_t) = \Pr(q_t = i, e_t = j, f_t = k|o_t,\lambda), \tag{73}$$

therefore:

$$\sum_{t=1}^{T}\sum_{Q}\sum_{E}\sum_{F}\Pr(Q,E,F|O,\lambda)\log\bar{w}_{q_te_tf_t} = \sum_{t=1}^{T}\sum_{i=1}^{N_s}\sum_{j=1}^{M}\sum_{k=1}^{L}\Pr(q_t=i,e_t=j,f_t=k|o_t,\lambda)\log\bar{w}_{q_te_tf_t} \tag{74}$$

To find the update equation of $\overline{w}_{ijk}$ we use the Lagrange multipliers optimization with the constraint in (28), and obtain

$$w_{ijk} = \frac{\sum_{t=1}^{T}\gamma_t(i,j,k)}{\sum_{t=1}^{T}\gamma_t(i,j)}, \tag{75}$$

where

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N_s}\alpha_t(j)\beta_t(j)}, \tag{76}$$

$$\kappa_t(i,j) = \gamma_t(i)\frac{u_{ij}b_{ij}(o_t)}{b_i(o_t)}, \tag{77}$$

and

$$\nu_t(i,j,k) = \gamma_t(i)\frac{u_{ij}w_{ijk}\mathcal{N}(o_t^{(k)},\mu_{ijk},\Sigma_{ijk})}{b_j(o_t)}. \tag{78}$$

Similarly, it can be shown that the update equations for the rest of the parameters are:

$$\mu_{ijkd} = \frac{\sum_{t=1}^{T} \nu_t(i,j,k) o_{td}^{(k)}}{\sum_{t=1}^{T} \nu_t(i,j,k)}, \tag{79}$$

and

$$\Sigma_{ijk} = \frac{\sum_{t=1}^{T} \nu_t(i,j,k)(\mathbf{o}_t^{(k)} - \boldsymbol{\mu}_{ijk})^t(\mathbf{o}_t^{(k)} - \boldsymbol{\mu}_{ijk})}{\sum_{t=1}^{T} \nu_t(i,j,k)}. \tag{80}$$

Appendix 2

Generalized Baum-Welch for the state level MSCHMM

The MSCHMM^{Ls} model parameters can be learned using a maximum Likelihood approach. Given a sequence of training observation $O = [o_1, \ldots, o_T]$, the parameters of λ could be learned by maximizing the likelihood of the observation sequence O, i.e., $\Pr(O|\lambda)$. We achieve this by generalizing the Baum-Welch algorithm to include a stream relevance weight component. We define the generalized Baum-Welch algorithm by extending the auxiliary function in (5) to

$$\mathbb{Q}(\lambda, \bar{\lambda}) = \sum_{Q}\sum_{F}\sum_{E} \Pr(Q,F,E|O,\lambda) \ln \Pr(O,Q,F,E|\bar{\lambda}), \tag{81}$$

where $F = [f_1, \ldots, f_T]$ and $E = [e_1, \ldots, e_T]$ are two sequences of random variables representing, respectively, the stream and component indices for each time step. It can be shown that a critical point of $\Pr(O|\lambda)$, with respect to λ, is a critical point of the new auxiliary function $\mathbb{Q}(\lambda, \bar{\lambda})$ with respect to $\bar{\lambda}$ when $\bar{\lambda} = \lambda$, that is:

$$\frac{\partial \Pr(O|\lambda)}{\partial \lambda} = \frac{\partial \mathbb{Q}(\lambda, \bar{\lambda})}{\partial \bar{\lambda}}\Big|_{\bar{\lambda}=\lambda}. \tag{82}$$

Similar to the discrete and mixture level cases, it could be shown that the formulation of the maximization of the likelihood $\Pr(O|\lambda)$ through maximizing the auxiliary function $\mathbb{Q}(\lambda, \bar{\lambda})$ is an EM [37] type optimization that is performed in two steps: the estimation step and the maximization step. The estimation step consists of computing the conditional expectation in (81) and writing it in an analytical form. The objective function in (81) involves the quantity $\Pr(O,Q,F,E|\bar{\lambda})$ which could be expressed analytically as

$$\Pr(O,Q,F,E|\bar{\lambda}_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^{T} w_{q_t f_t}^{(c)} u_{q_t f_t e_t}^{(c)} b_{q_t f_t e_t}^{(c)}(o_t) \tag{83}$$

Thus, the objective function in (81) can be expanded as

$$\begin{aligned}\mathbb{Q}(\lambda, \bar{\lambda}) &= \sum_{Q}\sum_{E}\sum_{F} \Pr(Q,E,F|O,\lambda) \log \bar{\pi}_{q_1} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{F}\sum_{E} \Pr(Q,F,E|O,\lambda) \log \bar{a}_{q_t q_{t+1}} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{F}\sum_{E} \Pr(Q,F,E|O,\lambda) \log \bar{w}_{q_t f_t} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{F}\sum_{E} \Pr(Q,F,E|O,\lambda) \log \bar{u}_{q_t f_t e_t} \\ &+ \sum_{t=1}^{T-1}\sum_{Q}\sum_{F}\sum_{E} \Pr(Q,F,E|O,\lambda) \log \mathcal{N}(o_t^{(f_t)}, \bar{\mu}_{q_t f_t e_t}, \bar{\Sigma}_{q_t f_t e_t})\end{aligned} \tag{84}$$

After the estimation step, the maximization step consists on finding the parameters of $\bar{\lambda}$ that maximize the function in (84). The expanded form of the function $\mathbb{Q}(\lambda, \bar{\lambda})$ in (84) has five terms involving $\overline{\pi}$, $\overline{a}$, $\overline{w}$, $\overline{u}$, and (μ, Σ). To find the values of $\overline{\pi}_i$, $\overline{a}_{ij}$, $\overline{w}_{ik}$, $\overline{u}_{ikj}$, $\overline{\mu}_{ikjd}$, and $\overline{\sigma}_{ikjd}$ that maximize $\mathbb{Q}(\lambda, \bar{\lambda})$, we consider the terms in (84) that depend on $\overline{\pi}$, $\overline{a}$, $\overline{w}$, $\overline{u}$, and (μ, Σ). In particular, the first and second terms in (71) depend on $\overline{\pi}$ and $\overline{a}$, and they have the same analytical expressions sketched in the case of the baseline CHMM in (5). It follows that the update equations for $\overline{\pi}_i$, and $\overline{a}_{ij}$ are the same as in the standard CHMM. That is,

$$\overline{\pi}_i = \gamma_1(i),$$

and

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)}.$$

To find the value of $\overline{w}_{ik}$ that maximizes the auxiliary function $\mathbb{Q}(.,.)$, only the third term of the expression in (84) is considered since it is the only part of $\mathbb{Q}(.,.)$ that depends on $\overline{w}_{ik}$. This term can be expressed as:

$$\begin{aligned}\sum_{t=1}^{T}\sum_{Q}\sum_{F}\sum_{E} \Pr(Q,E,F|O,\lambda) \log \bar{w}_{q_t f_t} &= \sum_{t=1}^{T}\sum_{Q}\sum_{F} \Pr(Q,F|O,\lambda) \log \bar{w}_{q_t f_t} \\ &= \sum_{t=1}^{T}\sum_{i}\sum_{k} \log(\bar{w}_{ik}) \times \sum_{Q}\sum_{F} \\ &\quad \times \Pr(Q,F|O,\lambda)\delta(i,q_t)\delta(k,f_t),\end{aligned} \tag{85}$$

where $\delta(i, q_t)\delta(k, f_t)$ keeps only those cases for which $q_t = i$, and $f_t = k$. That is,

$$\sum_Q \sum_F \Pr(Q, F|O, \lambda)\delta(i, q_t)\delta(k, f_t) = \Pr(q_t = i, f_t = k|o_t, \lambda), \tag{86}$$

therefore:

$$\sum_{t=1}^{T} \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log \bar{w}_{q_t f_t} = \sum_{t=1}^{T} \sum_{i=1}^{N_s} \sum_{k=1}^{L} \Pr(q_t = i, f_t = k|o_t, \lambda) \log \bar{w}_{q_t f_t} \tag{87}$$

To find the update equation of $\overline{w}_{ik}$ we use the Lagrange multipliers optimization with the constraint in (54), and obtain

$$\overline{w}_{ik} = \frac{\sum_{t=1}^{T} \kappa_t(i, k)}{\sum_{t=1}^{T} \gamma_t(i)}, \tag{88}$$

where,

$$\gamma_t(i) = \Pr(q_t = i|O, \lambda),$$

and

$$\kappa_t(i, k) = \gamma_t(i) \frac{w_{ik} b_{ik}(o_t)}{b_i(o_t)}.$$

Similarly, it can be shown that the update equations for the rest of the parameters are:

$$\overline{u}_{ikj} = \frac{\sum_{t=1}^{T} \nu_t(i, k, j)}{\sum_{t=1}^{T} \kappa_t(i, k)}, \tag{89}$$

$$\overline{\mu}_{ikjd} = \frac{\sum_{t=1}^{T} \nu_t(i, k, j) o_{td}^{(l)}}{\sum_{t=1}^{T} \nu_t(i, k, j)}, \tag{90}$$

$$\overline{\Sigma}_{ikj} = \frac{\sum_{t=1}^{T} \nu_t(i, k, j)(\mathbf{o}_t^{(k)} - \boldsymbol{\mu}_{ijd}^{(k)})^t(\mathbf{o}_t^{(k)} - \boldsymbol{\mu}_{ijd}^{(k)})}{\sum_{t=1}^{T} \nu_t(i, k, j)}, \tag{91}$$

where

$$\nu_t(i, k, j) = \gamma_t(i) \frac{w_{ik} u_{ijk} \mathcal{N}(o_t^{(k)}, \mu_{ijk}, \Sigma_{ijk})}{b_i(o_t)}$$

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This study was supported in part by U.S. Army Research Office Grants Number W911NF-08-0255 and by a grant from the Kentucky Science and Engineering Foundation as per Grant Agreement No. KSEF-2079-RDE-013 with the Kentucky Science and Technology Corporation. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office, or the U.S. Government.

Author details
[1]Portware, LLC., New York 10279, USA. [2]University of Louisville, Louisville, KY 40292, USA. [3]University of Florida, Gainesville, FL 32611, USA.

Received: 29 March 2012 Accepted: 17 December 2012
Published: 2 March 2013

References

1. L Rabiner, in *Proc. of the IEEE*, vol. 77. A tutorial on hidden Markov models and selected applications in speech recognition, (1989), pp. 257–286
2. P Runkle, P Bharadwaj, L Carin, Hidden Markov model multi-aspect target classification. IEEE Trans. Signal Proc. **47**, 2035–2040 (1999)
3. P Baldi, Y Chauvin, T Hunkapiller, M McClure, in *Nat. Acad. Science*, vol. 91. Hidden Markov models of biological primary sequence information (USA, 1994), pp. 1059–1063
4. T Koski, *Hidden Markov Models for Bioinformatics*. (Kluwer Academic Publishers, Netherlands, 2001)
5. H Frigui, K Ho, P Gader, Real-time landmine detection with ground-penetrating radar using discriminative and adaptive hidden Markov models. EURASIP J. Appl. Signal Process. **2005**, 1867–1885 (2005)
6. M Mohamed, P Gader, Generalized hidden Markov models part 2: applications to handwritten word recognition. IEEE Trans. Fuzzy Syst. **8**, 186–194 (2000)
7. H Bunke, T Caelli, *Hidden Markov Models: Applications in Computer Vision*. (World Scientific Publishing Co, Singapore, 2001)
8. W Zhiyong, C Lianhong, M Helen, Multi-level fusion of audio and visual features for speaker identification. Adv. Biometrics, 493–499 (2005)
9. CC Chibelushi, JSD Mason, F Deravi, in *Image Processing and Its Applications, 1997. Sixth International Conference on*, vol. 1. Feature-level data fusion for bimodal person recognition (IET, 1997), pp. 399–403
10. V Chatziz, A Bors, I Pitas, Multimodal decision level fusion for person authentication. IEEE Trans. Syst. Man Cybern. A. **29**, 674–680 (1999)
11. MI Jordan, Z Ghahramani, in *Advances in Neural Information Processing Systems 8: Proceedings of the 1995 Conference*, vol. 8. Factorial Hidden Markov Models (MIT Press, 1996), p. 472
12. N Ara, L Liang, T Fu, X Liu, in *Audio-and Video-Based Biometric Person Authentication*. A Bayesian approach to audio-visual speaker identification (Springer Berlin/Heidelberg, 2003), pp. 1056–1056
13. S Dupont, J Luettin, Audio-visual speech modeling for continuous speech recognition. IEEE Trans. Multimedia. **2**(3), 141–151 (2000)
14. P Gerasimos, N Chalapathy, L Juergen, M Iain, in *Audio-Visual Speech Processing*, vol. 2004, ed. by E Vatikiotis-Bateson, G Bailly, and P Perrier. Audio-visual automatic speech recognition: an overview (MIT Press, 2009), pp. 356–396. ISBN:0-26-222078-4
15. J Hernando, in *IEEE Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2. Maximum likelihood weighting of dynamic speech features for CDHMM speech recognition (Munich, 1997), pp. 1267–1270
16. A Torre, A Peinado, A Rubio, J Segura, C Benitez, Discriminative feature weighting for HMM-based continuous speech recognizers. Speech Commun. **38**, 267–286 (2002)
17. O Missaoui, H Frigui, P Gader, Landmine detection with ground penetrating radar using multistream discrete hidden Markov models. IEEE Trans. Geosci. Rem. Sens. **49**, 2080–2099 (2011)
18. BH Juang, W Chou, CH Lee, Minimum classification error rate methods for speech recognition. Trans. Speech Audio Process. **5**(3), 257–265 (1997)
19. O Missaoui, H Frigui, in *International Conference of Pattern Recognition*, vol. 19. Optimal feature weighting for continuous HMM (Florida, USA, 2008), pp. 1–4
20. X Li, M Parizeau, R Plamondon, Training hidden Markov models with multiple observations-a combinatorial method. IEEE Trans. Pattern Anal. Mach. Intell. **22**(4), 371–377 (2000)
21. A Nadas, A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional vesus conditional maximum likelihood. IEEE Trans. Acoust. Speech Signal Process. **31**(4), 814–817 (1983)
22. G Forney, The Viterbi algorithm. Proc. IEEE. **61**, 268–278 (1973)
23. M Masaru, S Iori, N Masafumi, H Yasuo, K Shingo, in *Universal Communication, 2008. ISUC'08. Second International Symposium on*. Sign language recognition based on position and movement using multi-stream HMM (IEEE, 2008), pp. 478–481

24. N Atta, S Sid-Ahmed, T Hesham, O Douglas, in *Electrical and Computer Engineering, 2008. CCECE 2008. Canadian Conference on*. Incorporating phonetic knowledge into a multi-stream HMM framework (IEEE, 2008), pp. 001705–001708
25. K Yousri, P Thierry, B AbdelMajid, in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, vol. 1. A multi-stream approach to off-line handwritten word recognition (IEEE, 2007), pp. 317–321
26. S Kapadia, Discriminative training of hidden Markov models. PhD thesis, University of Cambridge (1998)
27. G Potamianos, H Graf, in *Proc. of the Inter. Conf. on Acoustics, Speech, and Signal Processing*, vol. 6. Discriminative training of HMM stream exponents for audio-visual seech recognition (Seattle, 1998), pp. 3733–3736
28. G Potamianos, A Potamianos, in *Proc. EUROSPEECH*, vol. 3. Speaker adpatation for audio-visual speech recognition (Budapest, 1999), pp. 1291–1294
29. H Frigui, S Salem, in *Fuzzy Systems, 2003. FUZZ'03. The 12th IEEE International Conference on*, vol. 2. Fuzzy clustering and subset feature weighting (IEEE, 2003), pp. 857–862
30. Z Ghahramani, MI Jordan, Factorial hidden Markov models. Mach. Lear. **29**(2), 245–273 (1997)
31. KJ Hintz, in *Proceedings of the SPIE Conference on Detection and Remediation Technologies for Mines and Minelike Targets*, vol. IX. SNR improvements in NIITEK ground penetrating radar (Orlando, FL, USA, 2004), pp. 399–408
32. H Frigui, O Missaoui, P Gader, in *Proc. SPIE*, vol. 6553. Landmine detection using discrete hidden Markov models with Gabor features (Orlando, 2007). doi:10.1117/12.722241
33. J Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. (Plenum Press, New York, 1981)
34. RO Duda, PE Hart, DG Stork, *Pattern Classification (2nd Edition)*. (Wiley-Interscience, 2000)
35. P Gader, R Grandhi, W Lee, J Wislon, D Ho, in *SPIE Conf. Detect. Remediation Technol. Mines Minelike Targets*, vol. 5415. Feature analysis for the NIITEK ground-penetrating radar using order weighted averaging operators for landmine detection (Orlando, FL, 2004), pp. 953–962
36. H Frigui, P Gader, Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic K-Nearest neighbor classifier. IEEE Trans. Fuzzy Syst. **17**(1), 185–199 (2009)
37. AP Dempster, NM Laird, DB Rubin, Maximum likelihood from incomplete data via the EM algorithm. J. Royal Stat. Soc. Series B (Methodological). **39**(1), 1–38 (1977)

doi:10.1186/1687-6180-2013-40
Cite this article as: Missaoui *et al.*: **Multi-stream continuous hidden Markov models with application to landmine detection.** *EURASIP Journal on Advances in Signal Processing* 2013 **2013**:40.