

Land-Mine Detection With Ground-Penetrating Radar Using Multistream Discrete Hidden Markov Models

Oualid Missaoui, *Member, IEEE*, Hichem Frigui, *Member, IEEE*, and Paul Gader, *Senior Member, IEEE*

Abstract—We propose a multistream discrete hidden Markov model (DHMM) framework and apply it to the problem of land-mine detection using ground-penetrating radar (GPR). We hypothesize that each signature (mine or nonmine) can be characterized better by multiple synchronous sequences representing features that capture different environments and different radar characteristics. This paper is motivated by the fact that mines and clutter objects can have different characteristics depending on the mine type, soil and weather conditions, and burial depth. Thus, ideally different sets of specialized feature extraction mechanisms may be needed to achieve high detection and low false alarm rates. In order to fuse the different modalities, a multistream DHMM that includes a stream relevance weighting component is developed. The relevance weight of each stream depends on the symbols and the states. We reformulate the Baum–Welch and the minimum classification error/gradient probabilistic descent learning algorithms to include stream relevance weights and partial state probabilities. We generalize their objective functions and derive the necessary conditions to update all model parameters simultaneously. The results on a synthetic data set and a collection of GPR signatures show that the proposed multistream DHMM framework outperforms the basic single-stream DHMM where all the streams are treated equally important.

Index Terms—Baum–Welch (BW), discrete hidden Markov model (DHMM), discriminative training, generalized probabilistic descent (GPD), maximum likelihood, minimum classification error (MCE), multistream DHMM (MSDHMM), stream weighting.

I. INTRODUCTION

DETECTION, localization, and subsequent neutralization of buried land-mines are worldwide humanitarian and military problems. The latest statistics show that, as of August 2009, more than 70 states were believed to be mine affected.

Manuscript received November 13, 2009; revised June 14, 2010 and September 7, 2010; accepted October 17, 2010. Date of publication December 22, 2010; date of current version May 20, 2011. This work was supported in part by the U.S. Army Research Office under Grants W911NF-08-0255 and W911NF-07-1-0347 and in part by the National Science Foundation under Awards CBET-0730802 and CBET-0730484. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office, Army Research Laboratory, or the U.S. Government.

O. Missaoui is with Pipeline Financial Group, Inc., New York, NY 10165 USA.

H. Frigui is with the Computer Engineering and Computer Science Department, University of Louisville, Louisville, KY 40292 USA (e-mail: h.frigui@louisville.edu).

P. Gader is with the University of Florida, Gainesville, FL 32611 USA.

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TGRS.2010.2090886

Landmine Monitor has identified at least 73 576 casualties of land-mines, explosive remnants of war, and victim-activated improvised explosive devices in 119 states and areas in the past ten years [1]. Detection and removal of land-mines is therefore a significant problem and has attracted several researchers in recent years [2]–[7]. Although systems can achieve high detection rates, they have done so at the expense of high false alarm rates. Ground-penetrating radar (GPR) and electromagnetic induction (EMI) are two of the extensively investigated sensors due to their complementary features. For instance, GPR has the advantage of detecting mines with plastic or low-metal-content mines that are difficult to detect by traditional metal detectors. On the other hand, detectors using EMI sensors can outperform those using GPR in detecting small antipersonnel (AP) mines and deeply buried mines [8]. In fact, even for the same sensor, different detection algorithms may be more effective for different mine types, burial depths, and soil conditions [8]. For instance, in [8], it was shown that a detector that uses edge-based features [9] outperforms the spectral detector [10] for antitank (AT) mines and shallow mines. On the other hand, the spectral detector outperforms the edge-based detector for weak scattering plastic mines [10]. In [11], using a large GPR data set collected from four different sites, it was shown that the relative performance of four different detection algorithms is highly dependent on the site.

A typical land-mine detection system, like most automated detection algorithms, has three main components: 1) preprocessing; 2) feature extraction; and 3) classifier design. The preprocessing step relies on algorithms that perform tasks such as normalization of the data, corrections for variations in height and speed, removal of stationary effects due to the system response, etc. Methods that have been used to perform this task include wavelets and Kalman filters [12], subspace methods and matching to polynomials [13], and subtracting optimally shifted and scaled reference vectors [14]. The feature extraction step uses algorithms that reduce the preprocessed raw data to form a lower dimensional salient set of measures that represent the data. Principal component transforms [15], [16] are common tools that have been used to achieve this task. Other feature analysis approaches include wavelets [17], image processing methods of derivative feature extraction [18], and curve analysis using Hough and Radon transforms [19] as well as model-based methods. The classifier design phase includes three main steps: 1) a model learning step that consists of modeling the mine/nonmine signatures; 2) a confidence assignment step that

uses models to assign a confidence that a mine is present at a point; and 3) a decision making step that postprocesses the confidence map data to remove spurious responses and makes a final mine/nonmine decision. Over the past few years, various modeling and classification methods have been investigated for various land-mine detection systems. Examples include fuzzy logic [20], rules and order statistics [21], nearest neighbor classifiers [9], and hidden Markov models (HMMs) [22]–[24]. In particular, HMMs have proved to be very effective and were adapted to various GPR sensors [18], [22], [23].

In [18], hidden Markov modeling was proposed for detecting both metal and nonmetal mine types using GPR data. This (baseline) system uses both continuous and discrete HMMs (DHMMs) trained using the maximum likelihood estimation (MLE) technique and has proved that HMM techniques are feasible and effective for land-mine detection. In [25], Zhao *et al.* modified the discrete baseline HMM system to use the minimum classification error (MCE) training scheme. After initial training, this algorithm uses a sequential generalized probabilistic descent (GPD) algorithm that minimizes an empirical loss function to estimate the mine/background model parameters. The MCE-based DHMM was shown to achieve a significant performance improvement over the baseline system [25]. In [22], a complete real-time software system for land-mine detection using GPR was proposed and evaluated. The classifier component of this system is based on the continuous HMM and is an improved version of the baseline system in [18]. In particular, an improved model training approach that is based on a corrective training procedure to adjust the initial parameters to minimize the number of misclassification sequences was proposed.

Early work on land-mine detection using HMMs has relied on simple gradient edge features [18], [22], [25]. In [26], the authors proposed the use of alternative representation based on Gabor wavelet features. These features were extracted based on the expansion of the signature using a bank of Gabor filters at different scales and orientations. This choice was motivated by the fact that these features do not impose an explicit structure on the signature.

In fact, for any complex classification problem, such as land-mine detection, multiple features extracted from multiple sensors may be needed to achieve a satisfactory performance. However, using multiple sets of different features raises another important issue: How to combine them? Treating all features equally important may degrade the performance of the HMM detector. This is because mine and clutter objects can have different characteristics depending on the object type (e.g., high metal content versus low metal content and AT versus AP), burial depth (deep versus shallow), and soil and weather conditions.

The continuous HMM, through the use of multiple Gaussian components for each state, addresses the aforementioned problem to a certain degree. The learned covariance matrix for each component can encode the feature relevance weights and their correlation. However, learning the mixture of Gaussian can be a difficult task that could lead to erroneous results (i.e., incorrect modeling with incorrect covariance matrices) when some features are irrelevant for some components. This limitation is

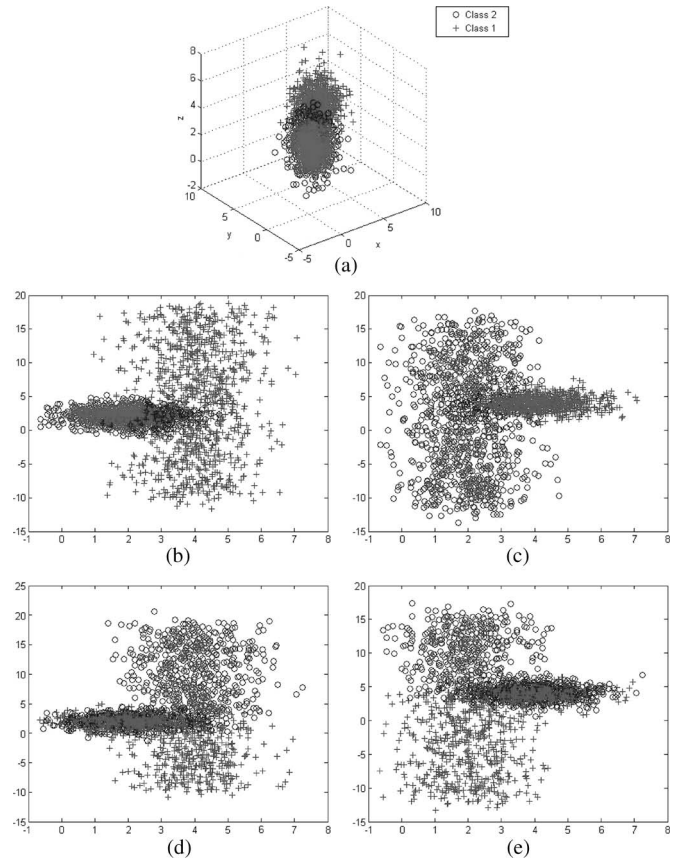


Fig. 1. Learning a mixture of Gaussian components in the presence of component dependent noise. (a) Three-dimensional data set with two Gaussian components. (b) Projection of the data in (a) after corrupting the y feature of component 1 by additive noise on the x – y plane. (c) Projection of the data in (a) after corrupting the z feature of component 2 by additive noise on the x – z plane. Projection of the data partitioned by the EM algorithm on the (d) x – y and (e) x – z planes. Points assigned to cluster 1 are shown by “+” signs, and points assigned to cluster 2 are shown by “o” signs.

illustrated in Fig. 1. Fig. 1(a) shows a 3-D data set generated from two Gaussian components. To simulate the scenario where all features are not equally important in characterizing both components, we corrupt the y feature of component 1 by adding random noise (i.e., feature y is irrelevant for component 1). Similarly, we corrupt the z feature and make it irrelevant for component 2. Fig. 1(b) shows the projection of the corrupted data on the x – y plane where only component 2 is well defined, and Fig. 1(c) shows the projection of the corrupted data on the x – z plane where only component 1 is well defined. Running a standard clustering algorithm, such as the EM, on these data would result in the two components shown in Fig. 1(d) and (e). As it can be seen, the EM fails to identify the two components, and the learned covariance matrices are erroneous and do not represent the true feature relevance weights and their correlation. We should note here that the incorrect partition is a result of treating relevant and irrelevant features equally important and not a consequence of the EM converging to a local minimum.

The drawbacks highlighted in Fig. 1 could be mitigated, for the case of static data, using recent unsupervised algorithms that perform clustering and feature discrimination simultaneously [27]. For sequential data, a multistream DHMM (MSDHMM)

that is trained in two independent steps was proposed in [23]. The first step consists of learning the relevance weights using unsupervised clustering and feature discrimination. The second step consists of learning the rest of the parameters of the DHMM using the standard Baum–Welch (BW) algorithm [28]. This initial MSDHMM was generalized further in [24] where the two steps were combined and optimized simultaneously.

In this paper, the MSDHMM structures proposed in [23] and [24] are revisited and built upon to develop a generalized MSDHMM framework. We propose two distinct structures that integrate a stream relevance weight for each symbol in each state. The first one uses a linear aggregation scheme while the second one uses a geometric aggregation. For each structure, we generalize the BW and the MCE/GPD training algorithms. In particular, we modify the objective function to include the stream relevance weights and derive the necessary conditions to optimize all of the model parameters simultaneously.

The organization of the rest of the paper is as follows. In Section II, we outline the baseline DHMM with maximum likelihood and discriminative training as well as existing HMM-based structures for multisensor fusion. In Section III, we present our MSDHMM structures which integrate a stream weighting component, and we derive the necessary conditions to optimize the parameters using BW and MCE/GPD learning. Section IV has the experimental results that compare the proposed MSDHMM with the baseline DHMM and the continuous HMM using a mixture of Gaussian components. Section V contains the conclusions and future directions.

II. RELATED WORK

A. DHMM

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, \dots, 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 (pdfs): the initial probabilities (π), the transition probabilities (\mathbf{A}), and the state pdfs (\mathbf{B}). Let T be the length of the observation sequence (i.e., number of time steps), let $O = [o_1, \dots, o_T]$ be the observation sequence, where each observation vector o_i is characterized by p features (i.e., $o_i \in \mathbb{R}^p$), and let $Q = [q_1, \dots, q_T]$ be the state sequence. The compact notation

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

is generally used to indicate the complete parameter set of the HMM model. In (1), $\pi = [\pi_i]$, where $\pi_i = \Pr(q_1 = s_i)$ denotes the initial state probabilities; $\mathbf{A} = [a_{ij}]$ is the state transition probability matrix, where $a_{ij} = \Pr(q_t = j | q_{t-1} = i)$ for $i, j = 1, \dots, N_s$; and $\mathbf{B} = \{b_i(o_t), i = 1, \dots, N_s\}$, where $b_i(o_t) = \Pr(o_t | q_t = i)$ is the set of observation probability distributions in state i .

For the DHMM, the observation vectors are commonly quantized into a finite set of M symbols v_1, \dots, v_M called codebook. Each state is represented by a discrete pdf, and each

symbol has a probability of occurring given that the system is in a given state. In other words, \mathbf{B} becomes a simple set of fixed probabilities for each class, i.e., $b_i(o_t) = b_{ij} = \Pr(v_j | q_t = i)$, where v_j is the nearest symbol o_t .

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 DHMM λ_c . Let $\mathbb{O} = [O^{(1)}, \dots, O^{(R)}]$ 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 $\{= 1, 2, \dots, C\}$. That is

$$\Gamma(O) = I, \quad \text{iff } I = \arg \max_{c=1 \dots C} g_c(O). \quad (2)$$

Two main approaches were considered for the training of the HMM parameters. The first one consists of maximizing the likelihood of the data correspondent to each model. The second approach consists of discriminative training that aims at minimizing the classification error over all classes.

1) *DHMM With MLE*: The BW algorithm [28] is an MLE algorithm commonly used to learn the HMM parameters. It consists of adjusting the parameters of each model λ to maximize the likelihood $\Pr(O|\lambda)$. Maximizing $\Pr(O|\lambda)$ is equivalent to maximizing the auxiliary function

$$\mathbb{Q}(\lambda, \bar{\lambda}) = \sum_Q \Pr(Q|O, \lambda) \ln (\Pr(O, Q|\bar{\lambda})) \quad (3)$$

where λ is the initial guess and $\bar{\lambda}$ is the subject of optimization. In fact, it has been proved in [29] that $\partial \Pr(O|\lambda) / \partial \lambda = (\partial \mathbb{Q}(\lambda, \bar{\lambda}) / \partial \bar{\lambda})|_{\bar{\lambda}=\lambda}$.

Using Lagrange optimization, the estimates of the parameters \mathbf{A} and \mathbf{B} could be computed iteratively using [28]

$$a_{ij} = \frac{\sum_{r=1}^R \sum_{t=1}^{T_r-1} \xi_t^{(r)}(i, j)}{\sum_{r=1}^R \sum_{t=1}^{T_r-1} \gamma_t^{(r)}(i)} \quad (4)$$

$$b_{ij} = \frac{\sum_{r=1}^R \sum_{t=1}^{T_r} \gamma_t^{(r)}(i) \delta(Q_V(o_t^r), j)}{\sum_{r=1}^R \sum_{t=1}^{T_r} \gamma_t^{(r)}(i)}. \quad (5)$$

In the aforementioned equation, Q_V is the quantization operation defined on an observation vector as

$$Q_V(o_t) = \arg \min_{1 \leq j \leq M} d(o_t, v_j) \quad (6)$$

where $d()$ is a distance measure, taken as the Euclidean distance in this paper. In (4) and (5)

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N_s} \sum_{j=1}^{N_s} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \quad (7)$$

$$\gamma_t(i) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{j=1}^{N_s} \alpha_t(j) \beta_t(j)} \quad (8)$$

$$\delta(i, j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

$$\alpha_t(j) = \Pr(o_1 o_2 \dots o_t, q_t = j | \lambda) \quad (10)$$

$$\beta_t(j) = \Pr(o_{t+1} o_{t+2} \dots o_T, | q_t = j, \lambda). \quad (11)$$

The variables $\alpha_t(j)$ and $\beta_t(j)$ are computed using the forward and backward algorithms [28], respectively.

2) *DHMM 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 [30] 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 [31]. In fact, it has been reported since the mid-1990s that discriminative training techniques were more successful, particularly for automatic speech recognition (ASR) [31]. The optimization of the error function is generally carried out by the GPD algorithm [31], a gradient descent-based optimization, and results in a classifier with minimum error probability.

Let

$$g_c(O, \Lambda) = \log \left[\max_Q g_c(O, Q, \Lambda) \right] \quad (12)$$

be the discriminant function, associated with classifier λ , which indicates the degree to which O belongs to class c . In (12), Q is a state sequence correspondent to the observation sequence O , Λ includes the model parameters, and

$$g_c(O, Q, \Lambda) = P(O, Q; \lambda_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \prod_{t=1}^T b_{q_t}^{(c)}(o_t). \quad (13)$$

In the aforementioned equation, $b_{q_t}^{(c)}(o_t) = b_{q_t Q_V(o_t)}^{(c)}$. 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 [32].

The misclassification measure of the sequence O is defined by

$$d_c(O) = -g_c(O, \Lambda) + \log \left[\frac{1}{C-1} \sum_{j \neq c} \exp[\eta g_j(O, \Lambda)] \right]^{\frac{1}{\eta}} \quad (14)$$

where η is a positive number. A positive $d_c(O)$ implies misclassification while a negative $d_c(O)$ indicates a correct decision. The misclassification measure is embedded in a smoothed zero-one loss function, defined as

$$l_c(O, \Lambda) = l(d_c(O)) \quad (15)$$

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

$$l(d) = \frac{1}{1 + \exp(-\zeta d + \theta)}. \quad (16)$$

In (16), θ is normally set to zero, and ζ is set to a number larger than one. Correct classification corresponds to loss values in $[0, 1/2)$, and misclassification corresponds to loss values in $(1/2, 1]$. 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

$$l(O; \Lambda) = \sum_{c=1}^C l_c(O; \Lambda) \mathbb{I}(O \in C_c) \quad (17)$$

where $\mathbb{I}(\cdot)$ is the indicator function.

Given a set of training observation sequences O_r , $r = 1, \dots, R$, an empirical loss function on the training data set is defined as

$$\mathbf{L}(\Lambda) = \sum_{r=1}^R \sum_{c=1}^C l_c(O^{(r)}; \Lambda) \mathbb{I}(O^{(r)} \in C_c). \quad (18)$$

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

$$a_{ij} \rightarrow \tilde{a}_{ij} = \log a_{ij} \quad b_{ij} \rightarrow \tilde{b}_{ij} = \log b_{ij}. \quad (19)$$

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

$$a_{ij} = \frac{\exp \tilde{a}_{ij}}{\sum_{j'=1}^{N_s} \exp \tilde{a}_{ij'}} \quad b_{ij} = \frac{\exp \tilde{b}_{ij}}{\sum_{j'=1}^M \exp \tilde{b}_{ij'}}. \quad (20)$$

Using a batch estimation mode, the DHMM parameters are iteratively updated using

$$\tilde{\Lambda}(\tau + 1) = \tilde{\Lambda}(\tau) - \epsilon \nabla_{\tilde{\Lambda}} \mathbf{L}(\Lambda)|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)}. \quad (21)$$

It can be shown that $\tilde{a}_{ij}^{(c)}$ and $\tilde{b}_{jk}^{(c)}$ need to be updated using

$$\tilde{a}_{ij}^{(c)}(\tau + 1) = \tilde{a}_{ij}^{(c)}(\tau) - \epsilon \left. \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{a}_{ij}^{(c)}} \right|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)} \quad (22)$$

$$\tilde{b}_{ij}^{(c)}(\tau + 1) = \tilde{b}_{ij}^{(c)}(\tau) - \epsilon \left. \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{b}_{ij}^{(c)}} \right|_{\tilde{\Lambda}=\tilde{\Lambda}(\tau)} \quad (23)$$

where

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

In the aforementioned equations

$$\frac{\partial d_c(O)}{\partial g_m(O, \Lambda)} = \begin{cases} -1, & \text{if } c = m \\ \frac{\exp[\eta g_c(O, \Lambda)]}{\sum_{j, j \neq c} \exp[\eta g_j(O, \Lambda)]}, & \text{if } c \neq m. \end{cases} \quad (24)$$

B. Multisensor Fusion

For complex classification systems, data are usually gathered from multiple sources of information that have varying reliability. For the problem of land-mine detection, the multiple sources could be different sensors, such as GPR and wideband EMI (WEMI), or different features and classifiers for one of the sensors. Assuming that the different sources have the same relevance in describing all the data might lead to a suboptimal solution. For instance, in [8], it was shown that local and context-dependent fusion (CDF) of multiple sensors and multiple algorithms outperforms global fusion. This is because the CDF approach can adapt the relevance of the different sources of information to different contexts (such as mine type, mine depth, soil type, etc.). In this paper, instead of fusing the results of the multiple algorithms that use different classifiers, we are interested in combining multiple sources of information at the feature level, i.e., feature-level CDF for the HMM classifier.

In the context of HMMs and for most real applications, different modalities could contribute to the generation of the sequence. These sources of information usually represent heterogeneous types of data. Multimodalities 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 for automatic audio–video speech recognition systems [33]. In fact, both speech and lip movement (possibly captured as video) are available when someone speaks. Natural modalities also appear in sign language recognition application where multistream HMM (MSHMM), based on the hand positions and movements, has been used [34]. In the second category, the modalities are synthesized by several feature extraction techniques with different characteristics and expressiveness such as the mel-frequency cepstral coefficients and formantlike features used to form ASR [35]. Synthesized modalities have also been used to combine upper contour features and lower contour features as two streams for offline handwritten word recognition [36].

Approaches toward the combination of the different modalities can be divided into three main categories: feature-level fusion or direct identification, decision-level fusion or separate identification (also known as late integration), and model-level fusion (early/intermediate integration) [37]. In the feature-level fusion, multiple features are concatenated into a large feature vector, and a single HMM model is trained [38]. This type of fusion has the drawback of treating heterogeneous features equally important. It also cannot easily represent the loose timing synchronicity between different modalities. In the decision-level fusion, the modalities are processed separately to build independent models [39]. This approach completely ignores the correlation between features and allows complete asynchrony between the streams. Moreover, it is computationally heavy since it involves two layers of decision. In the model-level fusion, an HMM model that is more complex than a standard one is sought. This additional complexity is needed to handle the correlation between modalities and the loose synchronicity between sequences. Several HMM structures have been proposed for this purpose. Examples include factorial HMM [40], coupled HMM [41], and MSHMM [42]. Both factorial and

coupled HMM structures allow asynchrony between sequences since a state sequence is assigned to each stream [33].

MSHMM is an HMM-based structure that handles multiple modalities for temporal data. It is used when the modalities (streams) are synchronous and independent. Since the streams are supposed to be synchronous, MSHMM assumes that, for each time slot, there is a single hidden state from which different streams interpret different observations. The independence of the streams means that their interpretation of the hidden state and their generation of the observations are performed independently.

III. MSDHMM

In this section, we propose various MSDHMM structures that integrate a stream relevance weight. For each structure, we generalize the BW and the MCE/GPD training algorithms. We generalize the objective function to include the stream relevance weights and derive the necessary conditions to update the parameters. 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 thus represented by a different subset of features. Instead of treating the streams equally important or using user-specified weights, the proposed MSDHMM structure introduces a built-in component to learn the relevance weight to each stream. We also propose two different data-driven methods to learn the relevance weights. The first one is based on distance weighting, and the second one is based on probability weighting. In the distance-based approach, a weight is assigned to each feature subset (i.e., each stream), and the distance computation between samples becomes a weighted aggregation of the partial distances from the different streams. In the probability-based approach, a partial probability is assigned to each stream of each symbol, and the overall observation probability of each symbol is computed as an aggregation between the stream relevance weights and the partial probabilities. For the probability-based method, we propose two types of aggregation: arithmetic weighting and geometric weighting aggregations.

A. Distance-Based Approach to Learn Multistream Relevance Weights

The proposed distance-based MSDHMM (referred to as MSDHMM^D) structure is defined as

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

where π , \mathbf{A} , and \mathbf{B} are the state prior probabilities, the transition probabilities, and the observation probabilities, respectively, as in the baseline DHMM structure. The additional parameter $\mathbf{W} = [w_{jk}]$ is an $M \times L$ matrix that represents the relevance weight of each symbol with respect to each stream. In particular, a stream relevance weight w_{jk} is assigned to each symbol j to indicate the relevance of stream k for this symbol. The proposed structure assumes a dependence between the streams and the states.

The optimization of MSDHMM^D parameter λ is achieved in two steps. The first step combines the initialization and the learning of \mathbf{W} . The second step uses the standard BW

algorithm [28] to learn the \mathbf{A} and \mathbf{B} parameters. For each model λ_c , the initialization step consists of learning the N_s states, learning the codebook, and assigning initial probabilities to each symbol. The states and the codebook could be obtained by partitioning and quantizing the training data. Any clustering algorithm, such as the k -means [43] or the fuzzy c -means (FCM) [44], could be used for this task. In our application, we use the simultaneous clustering and attribute discrimination (SCAD) [27]. The SCAD can perform clustering and feature discrimination simultaneously and in an unsupervised manner. The feature relevance weights learned by SCAD have two main advantages. First, they guide the clustering process in identifying more meaningful clusters by learning relevance weights to the different feature subsets and identifying clusters in subspaces of the original high-dimensional feature space. Second, the learned feature weights could be used as the relevance weights of the symbols with respect to the different streams.

Let d_{ij}^k be the partial distance between data vector x_i and cluster j using the feature from the k th stream only. The total distance d_{ij} between x_i and cluster j is then computed by aggregating the partial degrees of similarities from the L streams and their weights. That is, we let

$$d_{ij}^2 = \sum_{k=1}^L w_{ik} d_{ijk}^2. \quad (26)$$

The SCAD is an unsupervised algorithm that minimizes the sum of intracluster distances. It is an iterative algorithm that alternates updating between the cluster centers, feature relevance weights, and membership assignment of each sample to each cluster. We refer the reader to [27] for more details about the SCAD.

In our DHMM framework, we use SCAD to initialize the codebook by partitioning the training data into M clusters ($v_j, j = 1, \dots, M$) and assigning a feature relevance weight w_{jk} to each symbol v_j with respect to each stream k . After learning the codebook, the DHMM requires associating a probability value with each symbol in each state. The probability of v_j in state i represents its likelihood in that state. We use an FCM-type [44] membership function to initialize these probabilities, i.e., we let

$$b_{ij} = \frac{1/d_{ij}^2}{\sum_{s=1}^{N_s} 1/d_{sj}^2}. \quad (27)$$

The closer v_j is to s_i , the higher its likelihood is in state i , which explains the usage of the inverse of the distance in the numerator of (27). The denominator in (27) is a normalizing factor. To satisfy the requirement that $\sum_{j=1}^M b_{ij} = 1$, we scale b_{ij} using

$$b_{ij} \leftarrow \frac{b_{ij}}{\sum_{k=1}^M b_{ik}}. \quad (28)$$

The remaining HMM model parameters \mathbf{A} , \mathbf{B} , and π are then estimated using the standard BW algorithm [28] as outlined in the previous section with a minor modification. Recall that the learning equation of b_{ij} in the discrete BW is

$$b_{ij} = \frac{\sum_{t=1}^T \gamma_t(i) \delta(Q_V(o_t), j)}{\sum_{t=1}^T \gamma_t(i)} \quad (29)$$

where $\gamma_t(i)$ and $\delta(\cdot, \cdot)$ are as defined in (8) and (9), respectively, and Q_V is the quantization operation defined on an observation vector o_t as the index of the closest symbol to o_t . In our case, to identify the closest symbol to an observation, we take advantage of the stream relevance weights associated with each symbol. That is, the closest symbol to o_t is the symbol which index $Q_V(o_t)$ satisfies

$$Q_V(o_t) = \arg \min_{1 \leq j \leq M} \sum_{k=1}^L w_{jk} \|o_{tk} - v_{jk}\|^2. \quad (30)$$

In (30), w_{jk} emphasizes the contribution of the stream k in the decision of the closest symbol to o_t . Thus, these learned weights affect the b_{ij} update equation in the BW algorithm. The steps of the resulting training procedure of the different parameters of each λ_c are outlined in Algorithm 1. The stopping criteria are either the increase of the likelihood $\Pr(O|\lambda)$ is not significant or the number of iterations exceeds a predefined number.

Algorithm 1 BW training of the distance-based MSDHMM

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L .

Ensure:

- 1: Cluster training data into N_s clusters using SCAD, and let s_i , the center of each cluster, be the representative of state i
 - 2: Cluster the training data (using SCAD) to quantize it into M symbols, and learn the stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.
 - 3: **while** stopping criteria not satisfied **do**
 - 4: Compute the closest observation to o_t using (30);
 - 5: update \mathbf{A} using (4);
 - 6: update \mathbf{B} using (5);
 - 7: **end while**
-

In this version of the MSDHMM, the stream relevance weights are learned during the initial clustering step and are not updated in the HMM parameter learning. Thus, the discriminative training version of this MSDHMM is carried out in exactly the same way as that for the baseline DHMM. 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 distance-based MSDHMM

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L for each model λ_c .

Ensure:

- 1: For each λ_c , cluster training data into N_s clusters using SCAD, and let s_i , the center of each cluster, be the representative of state i .
- 2: For each λ_c , cluster the training data (using SCAD) to quantize it into M symbols, and learn the stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.

3: **while** stopping criteria not satisfied **do**
 4: Compute the closest observation to each o_t using (30);
 5: Compute the loss function of each sequence O using (17);
 6: update \mathbf{A} of each λ_c using (22);
 7: update \mathbf{B} of each λ_c using (23);
 8: **end while**

The distance weighting approach provides a simple structure of the MSDHMM. However, it has two main limitations. First, the stream weights are independent of the states. Second, the weights are learned independently from the rest of the DHMM parameters and do not maximize the likelihood estimates. To overcome these limitations, we propose an alternative approach that is based on assigning partial probabilities to the different streams.

B. Probability-Based Approach to Learn Multistream Relevance Weights

Let \mathbf{W} be an $N_s \times M \times L$ stream relevance weight matrix. In particular, we assume that a stream relevance weight w_{ijk} is assigned to each symbol j of each stream k within each state i . This choice takes into account the additional dependence between the streams and the states. We call this new structure of MSDHMM the probability-based MSDHMM or simply MSDHMM^P. In this version, each symbol j of each stream k , i.e., v_{jk} , is assigned a partial probability b_{ijk} in each state i . The stream relevance weights w_{ijk} and the probabilities b_{ijk} are combined to form the observation state probabilities b_{ij} . Two different combination methods are considered in this paper. The first one uses a linear combination of the weights and the partial probabilities, while the second one uses a geometric combination.

1) *Linear Aggregation*: In this method, the observation state probabilities are computed using

$$b_{ij} = \sum_{k=1}^L w_{ijk} b_{ijk} \quad (31)$$

subject to

$$\sum_{k=1}^L w_{ijk} = 1 \quad \sum_{j=1}^M b_{ijk} = 1. \quad (32)$$

This linear form of the observation probability is motivated by the following probabilistic reasoning:

$$\begin{aligned} b_{ij} &= \Pr(o_t | q_t = i; \lambda) \\ &= \sum_{k=1}^L \Pr(v_j, f_t = k | q_t = i; \lambda) \Pr(f_t = k | q_t = i; \lambda). \end{aligned} \quad (33)$$

In (33), the introduced random variable f_t represents the index of the most relevant source of information at time t . That is, at time t , one of the L streams is 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 a “collective” way where all the sources contribute (each with a small

portion) to the characterization of the raw data. In (33), v_j is the j th symbol and has feature components from all L sources, i.e., $v_j = [v_j^{(1)}, \dots, v_j^{(L)}]$. Since we are assuming that, at any instant t , there is one source of information that is significantly more relevant than the others, b_{ij} can be approximated using

$$b_{ij} \approx \sum_{k=1}^L \Pr(v_j^{(k)}, f_t = k | q_t = i; \lambda) \Pr(f_t = k | v_j, q_t = i; \lambda). \quad (34)$$

It follows then that

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

We will refer to this probability-based MSDHMM with linear aggregation as MSDHMM^{P1}.

As in the previous method, the learning starts by clustering the training data into N_s clusters using SCAD [27]. The center s_i of each cluster is used as the state's representative. Next, SCAD is used to partition the data into a larger number of clusters and build the codebook $V = [v_1, \dots, v_M]$. The SCAD also learns the stream relevance weight w_{jk} for each symbol. Since the MSDHMM^{P1} structure requires a weight in each state, initially, we duplicate the weights computed via SCAD, i.e., we let $w_{ijk} = w_{jk}$ for $i = 1 \dots N_s$. The probability of each symbol v_{jk} in each stream k and within each state i can be represented by the fuzzy membership degree of v_{jk} in this state. That is, we use

$$b_{ijk} = \frac{1/d_{ijk}^2}{\sum_{s=1}^{N_s} 1/d_{sjk}^2} \quad (35)$$

where d_{ijk} is the partial distance between symbol v_j and state s_i , taking into account only the features from stream k . To satisfy the requirement that $\sum_{j=1}^M b_{ijk} = 1$, we scale the values using

$$b_{ijk} \leftarrow \frac{b_{ijk}}{\sum_{m=1}^M b_{imk}}. \quad (36)$$

The overall probability of v_j in state i is then computed using (31).

a) *Learning model parameters with generalized MLE*: After initialization, the model parameters can be tuned using the maximum likelihood approach. Given a sequence of training observation $O = [o_1, \dots, o_T]$, the parameters of λ_c could be learned by maximizing the likelihood of the observation sequence O , i.e., $\Pr(O|\lambda)$. We achieve this by generalizing the BW algorithm to include a stream relevance weight component. We define the generalized BW algorithm by extending the auxiliary function in (3) to

$$\mathbb{Q}(\lambda, \bar{\lambda}) = \sum_Q \sum_F \Pr(Q, F | O, \lambda) \ln (\Pr(O, Q, F | \bar{\lambda})) \quad (37)$$

where $F = [f_1, \dots, f_T]$ is a sequence of random variables representing the stream 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 $\bar{\lambda} = \lambda$, i.e., $\partial \Pr(O|\lambda) / \partial \lambda = (\partial \mathbb{Q}(\lambda, \bar{\lambda}) / \partial \bar{\lambda})|_{\bar{\lambda}=\lambda}$.

Using Lagrange multiplier optimization, it can be shown (see Appendix A) that the feature relevance weights need to be updated iteratively using

$$w_{ijk} = \begin{cases} \frac{\sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t), j)}{\sum_{t=1}^T \gamma_t(i) \delta(Q_V(o_t), j)}, & \text{if } \exists t, \delta(Q_V(o_t), j) = 1 \\ \frac{1}{L}, & \text{otherwise.} \end{cases} \quad (38)$$

In (38), Q_V is the quantization operation defined on an observation vector o_t as

$$Q_V(o_t) = \arg \min_{1 \leq j \leq M} \frac{1}{N_s} \sum_{i=1}^{N_s} \sum_{k=1}^L w_{ijk} \|o_{tk} - v_{jk}\|. \quad (39)$$

The numerator in (38) reflects the quantity of information provided by stream k while the denominator is used for normalization. Similarly, it can be shown (see Appendix A) that the partial probabilities need to be updated using

$$b_{ijk} = \begin{cases} \frac{\sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t), j)}{\sum_{t=1}^T \gamma_t(i, k)}, & \text{if } \exists t, \delta(Q_V(o_t), j) = 1 \\ \frac{1}{M}, & \text{otherwise.} \end{cases} \quad (40)$$

In (40), the numerator represents the contribution of each stream k for each code v_j within state i , and the denominator is a normalization factor. The updating equation for a_{ij} remains the same as that in the standard BW algorithm [i.e., as that in (4)].

In the case of multiple observations $[O^{(1)}, \dots, O^{(R)}]$, it can be easily shown that the learning equations need to be updated using

$$w_{ijk} = \begin{cases} \frac{\sum_{r=1}^R \sum_{t=1}^T \gamma_t^r(i, k) \delta(Q_V(o_t^r), j)}{\sum_{r=1}^R \sum_{t=1}^T \gamma_t^r(i) \delta(Q_V(o_t^r), j)}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \frac{1}{L}, & \text{otherwise} \end{cases} \quad (41)$$

$$b_{ijk} = \begin{cases} \frac{\sum_{r=1}^R \sum_{t=1}^T \gamma_t^r(i, k) \delta(Q_V(o_t^r), j)}{\sum_{r=1}^R \sum_{t=1}^T \gamma_t^r(i, k)}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \frac{1}{M}, & \text{otherwise.} \end{cases} \quad (42)$$

Algorithm 3 outlines the steps of the MLE training procedure of the different parameters of the MSDHMM^{P_l}.

Algorithm 3 Generalized BW training for the probability-based MSDHMM with linear aggregation

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L .

Ensure:

- 1: Cluster training data into N_s clusters, and let the center of each cluster s_i be the representative of state i .
- 2: Quantize the training data into M symbols, and learn initial stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.
- 3: Let $w_{ijk} = w_{jk}$, for $1 \leq i \leq N_s$
- 4: **while** stopping criteria not satisfied **do**
- 5: Compute the closest observation to o_t using (39);

- 6: update \mathbf{A} using (4);
- 7: update \mathbf{W} using (41);
- 8: update \mathbf{B} using (42);
- 9: **end while**

b) Learning model parameters with generalized MCE/GPD: We generalize the MCE/GPD training approach to the case of the MSDHMM. In particular, we extend the discriminant function in (12) to accommodate for the stream relevance weights using

$$g_c(O, q, \Lambda) = P(O, q; \lambda_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \times \prod_{t=1}^T \left[\sum_{k=1}^L w_{q_t k}^{(c)}(o_t) b_{q_t k}^{(c)}(o_t) \right] \quad (43)$$

where $b_{q_t k}^{(c)}(o_t) = b_{q_t Q_V(o_t) k}^{(c)}$, $w_{q_t k}^{(c)}(o_t) = w_{q_t Q_V(o_t) k}^{(c)}$, and Q_V is defined in (39).

Using (43) in the loss function in (16), we obtain the generalized objective function of the discriminative MSDHMM^{P_l}. It can be shown (see Appendix C) that minimization of this empirical loss results in the following update equations:

$$\tilde{w}_{ijk}^{(c)}(\tau + 1) = \tilde{w}_{ijk}^{(c)}(\tau) - \epsilon \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} \Big|_{\tilde{\Lambda} = \tilde{\Lambda}(\tau)} \quad (44)$$

$$\tilde{b}_{ijk}^{(c)}(\tau + 1) = \tilde{b}_{ijk}^{(c)}(\tau) - \epsilon \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} \Big|_{\tilde{\Lambda} = \tilde{\Lambda}(\tau)} \quad (45)$$

where

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

In the aforementioned equations, $\partial d_c(O_r) / \partial g_m(O_r, \Lambda)$ is the same as that in (24), and \tilde{w}_{ijk} and \tilde{b}_{ijk} are mapped versions of w_{ijk} and b_{ijk} , respectively. The mapping is similar to the ones used for a_{ij} and b_{ij} in (20). The update equation for \tilde{a}_{ij} is the same as that in (22). Algorithm 4 outlines the steps needed to learn the parameters of all the models λ_c in the MCE/GPD framework.

Algorithm 4 MCE/GPD training of the probability-based MSDHMM with linear aggregation

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L for each model λ_c .

Ensure:

- 1: For each λ_c , cluster training data into N_s clusters, and let the center of each cluster s_i be the representative of state i .

- 2: For each λ_c , quantize the training data into M symbols, and learn initial stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.
- 3: Let $w_{ijk} = w_{jk}$, for $1 \leq i \leq N_s$
- 4: **while** stopping criteria not satisfied **do**
- 5: Compute the closest observation to each o_t using (30).
- 6: Compute the loss function of each sequence O using (17);
- 7: update \mathbf{A} of each λ_c using (22);
- 8: update \mathbf{B} of each λ_c using (45);
- 9: update \mathbf{W} of each λ_c using (44);
- 10: **end while**

To test a new observation sequence $O = [o_1, \dots, o_T]$, we need to compute $\Pr(O|\lambda_c)$ with respect to each model λ_c . This computation can be performed efficiently using the Viterbi algorithm [32]. The Viterbi algorithm computes the correspondent optimal state sequence $[q_1, \dots, q_T]$ to O . This, in turn, requires the computation of $b_i(o_t)$, which can be computed using

$$b_i(o_t) = b_{ij} \quad (46)$$

where $j = Q_V(o_t)$ and is computed using (39).

2) *Geometric Aggregation*: In this method, the partial probabilities are combined using

$$b_{ij} = \prod_{k=1}^L [b_{ijk}]^{w_{ijk}}, \quad \text{subject to } \sum_{s=1}^L w_{ijs} = \kappa \text{ and } \sum_{j=1}^M b_{ijk} = 1. \quad (47)$$

The geometric form of the observation probability in (47) is motivated by the following probabilistic reasoning:

$$b_{ij} = \Pr(v_j|q_t = i; \lambda) = \Pr(v_j^{(1)}, \dots, v_j^{(L)}|q_t = i; \lambda). \quad (48)$$

Assuming that the L sources of information are independent, (48) can be rewritten as

$$b_{ij} = \prod_{k=1}^L \Pr(v_j^{(k)}|q_t = i; \lambda). \quad (49)$$

In (49), b_{ij} does not encode the relevance of each source of information. In fact, a feature $v_j^{(k)}$ may be probable (high relative frequency) but not relevant (low discrimination power). Therefore, we assume the existence of an additional variable w_{ijk} that characterizes the relevance of each source and approximate b_{ij} using

$$b_{ij} \approx \prod_{k=1}^L \left[\Pr(v_j^{(k)}|q_t = i; \lambda) \right]^{w_{ijk}}.$$

It follows that

$$b_{ijk} = \Pr(v_j^{(k)}|q_t = i; \lambda).$$

We will refer to this probability-based MSDHMM with geometric aggregation as MSDHMM^{P_g}. In (47), κ is a constant, usually set to one, and $\nu \in (1, \infty)$ is an exponent that controls the discrimination between the different streams. We follow the same steps as those used for the linear combination for the initialization of the model parameters; the only difference resides in using (47) instead of the linear combination in (31).

a) *Learning model parameters with generalized MLE*: It can be shown (see Appendix B) that the weights need to be updated using

$$\bar{w}_{ijk} = \begin{cases} \left[\kappa \frac{[D_{ijk}]^{\frac{\nu}{\nu-1}}}{\sum_{k=1}^L [D_{ijk}]^{\frac{\nu}{\nu-1}}} \right]^{\frac{1}{\nu}}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \left(\frac{\kappa}{L} \right)^{\frac{1}{\nu}}, & \text{otherwise} \end{cases} \quad (50)$$

where $D_{ijk} = \sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t^r), j) \log(b_{ijk})$. Similarly, it can be shown (see Appendix B) that the update equation for the partial probabilities is (51), shown at the bottom of the page. In the case of multiple observations $[O^{(1)}, \dots, O^{(R)}]$, it can be easily shown that the update equations become (52) and (53), shown at the bottom of the page.

$$\bar{b}_{ijk} = \begin{cases} \frac{\sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t^r), j) w_{ijk}}{\sum_{j'=1}^M \sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t^r), k) w_{ij'k}}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \frac{1}{M}, & \text{otherwise} \end{cases} \quad (51)$$

$$\bar{w}_{ijk} = \begin{cases} \left[\kappa \frac{[\sum_{l=1}^S D_{ijk}]^{\frac{\nu}{\nu-1}}}{\sum_{r=1}^R \sum_{k=1}^L [D_{ijk}]^{\frac{\nu}{\nu-1}}} \right]^{\frac{1}{\nu}}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \left(\frac{\kappa}{L} \right)^{\frac{1}{\nu}}, & \text{otherwise} \end{cases} \quad (52)$$

$$\bar{b}_{ijk} = \begin{cases} \frac{\sum_{r=1}^R \sum_{j'=1}^M \sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t^r), j) w_{ijk}}{\sum_{r=1}^R \sum_{j'=1}^M \sum_{t=1}^T \gamma_t(i, k) \delta(Q_V(o_t^r), k) w_{ij'k}}, & \text{if } \exists t, \delta(Q_V(o_t^r), j) = 1 \\ \frac{1}{M}, & \text{otherwise} \end{cases} \quad (53)$$

Algorithm 5 Generalized BW training for the probability-based MSDHMM with geometric aggregation

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L .

Ensure:

- 1: Cluster training data into N_s clusters, and let the center of each cluster s_i be the representative of state i .
- 2: Quantize the training data into M symbols, and learn initial stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.
- 3: Let $w_{ijk} = w_{jk}$, for $1 \leq i \leq N_s$
- 4: **while** stopping criteria not satisfied **do**
- 5: Compute the closest observation to o_t using (39);
- 6: update **A** using (4);
- 7: update **W** using (52);
- 8: update **B** using (53);
- 9: **end while**

b) Learning model parameters with generalized MCE/GPD: For this model, we generalize the discriminant function in (12) to accommodate for the stream relevance weights using

$$g_c(O, q, \Lambda) = P(O, q; \lambda_c) = \pi_{q_0^{(c)}} \prod_{t=1}^{T-1} a_{q_t q_{t+1}}^{(c)} \times \prod_{t=1}^T \prod_{k=1}^L \left[b_{q_t k}^{(c)}(o_t) \right]^{w_{q_t k}^{(c)}(o_t)} \quad (54)$$

where $b_{q_t k}^{(c)}(o_t) = b_{q_t Q_V(o_t)k}^{(c)}$, $w_{q_t k}^{(c)}(o_t) = w_{q_t Q_V(o_t)k}^{(c)}$, and Q_V is defined in (39). Using this function in the loss function in (16), we obtain the new objective function of the discriminative MSDHMM^{P_g}. It can be shown (see Appendix D) that minimization of this empirical loss results in the following update equations:

$$\tilde{w}_{ijk}^{(c)}(\tau + 1) = \tilde{w}_{ijk}^{(c)}(\tau) - \epsilon \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} \Big|_{\tilde{\Lambda} = \tilde{\Lambda}(\tau)} \quad (55)$$

$$\tilde{b}_{ijk}^{(c)}(\tau + 1) = \tilde{b}_{ijk}^{(c)}(\tau) - \epsilon \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} \Big|_{\tilde{\Lambda} = \tilde{\Lambda}(\tau)} \quad (56)$$

where

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} &= \sum_{r=1}^R \sum_{m=1}^C \sum_{t=1}^T \zeta l_m(O_r, \Lambda) (1 - l_m(O_r, \Lambda)) \\ &\quad \times \delta(q_t^r = i, Q_V(o_t^r) = j) \left[\tilde{w}_{ijk}^{(c)} \right]^{\nu-1} w_{ijk}^{(c)} \\ &\quad \times \left(1 - \left[w_{ijk}^{(c)} \right]^\nu \right) \log b_{ijk}^{(c)} \times \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)} \end{aligned} \quad (57)$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} &= \sum_{r=1}^R \sum_{m=1}^C \sum_{t=1}^T \zeta l_m(O_r, \Lambda) (1 - l_m(O_r, \Lambda)) \\ &\quad \times \delta(q_t^r = i, Q_V(o_t^r) = j) b_{ijk}^{(c)} \\ &\quad \times \left(1 - b_{ijk}^{(c)} \right) \frac{w_{ijk}^{(c)}}{b_{ijk}^{(c)}} \frac{\partial d_c(O_r)}{\partial g_m(O_r, \Lambda)}. \end{aligned} \quad (58)$$

In the aforementioned equations, $\partial d_c(O_r)/\partial g_m(O_r, \Lambda)$ is the same as that in (24), and \tilde{w}_{ijk} and \tilde{b}_{ijk} are mapped versions of w_{ijk} and b_{ijk} , respectively. The mapping is similar to the ones used for a_{ij} and b_{ij} in (20). The update equation for \tilde{a}_{ij} is the same as that in (22).

Algorithm 6 MCE/GPD training of the probability-based MSDHMM with geometric aggregation

Require: Training data $[O_1, \dots, O_R]$, $O_i = [o_1, \dots, o_T]$. Fix the variables N_s , M , and L for each model λ_c .

Ensure:

- 1: For each λ_c , cluster training data into N_s clusters, and let the center of each cluster s_i be the representative of state i .
- 2: For each λ_c , quantize the training data into M symbols, and learn initial stream relevance weights w_{jk} . The center of each cluster v_j is a symbol.
- 3: Let $w_{ijk} = w_{jk}$, for $1 \leq i \leq N_s$
- 4: **while** stopping criteria not satisfied **do**
- 5: Compute the closest observation to each o_t using (30).
- 6: Compute the loss function of each sequence O using (17);
- 7: update **A** of each λ_c using (22);
- 8: update **B** of each λ_c using (56);
- 9: update **W** of each λ_c using (55);
- 10: **end while**

IV. EXPERIMENTAL RESULTS

In this section, we illustrate the performance of the proposed MSDHMM architectures. We first use synthetically generated data sets to outline the advantages of the proposed structures and learning algorithms. We also apply the proposed architectures to the problem of land-mine detection using GPR sensor data.

A. Synthetic Data

1) Data Generation: To validate the proposed MSDHMM structures, we generate two synthetic data sets using two discrete DHMMs to simulate a two-class problem. The first set is single-stream sequential data, and the second one is multistream ones. We follow an approach similar to the one used in [45] to generate sequential data with $N_s = 4$ states and $M = 120$ symbols with four dimensions. We start by fixing N_s different vectors $s_i \in \mathbb{R}^4$ to represent the different states. Then, we randomly sample M/N_s vectors from each normal distribution with mean s_i and identity covariance matrix. A codebook with M symbols ($v_j, j = 1, \dots, M$) is then formed. For each symbol, the membership in each state is computed using

$$b_{jk} = \frac{1/\|s_j - v_k\|}{\sum_{i=1}^{N_s} 1/\|s_i - v_k\|} \quad (59)$$

and then scaled using

$$b_{jk} \leftarrow \frac{b_{jk}}{\sum_{l=1}^M b_{jl}}. \quad (60)$$

The initial state probability distribution and the state transition probability distribution, for both models, are generated randomly from a uniform distribution in the interval $[0, 1]$ and scaled to satisfy the stochastic constraints. Then, we generate R sequences of length $T = 15$ observations for each of the two classes using Algorithm 7.

Algorithm 7 Single-stream sequential data generation for each class

```

for  $r = 1$  to  $R$  do
    Select the initial state according to the initial state
    probability distribution  $\pi$ 
    Randomly pick a vector  $v$  from the  $M$  symbols among
    those representing the selected state
    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 probability transition
        matrix  $A$ ,
        Randomly pick a symbol  $v$  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 multistream case, we assume that the sequential data are synthesized by $L = 2$ streams and that each stream k is described by N_s states, where each state i , $s_i \in \mathbb{R}^4$. For each state, 30 symbols are generated from each stream and concatenated to form a double-stream set of symbols. To simulate streams with various relevance weights, we create three groups of symbols in each state. The first group is formed by concatenating ten symbols from each stream by just appending the features (i.e., both streams are relevant). The second group of symbols is formed by concatenating noise (instead of stream 2 features) to stream 1 features (i.e., stream 1 is relevant and stream 2 is irrelevant). The last group of symbols is formed by concatenating noise (instead of stream 1 features) to stream 2 features (i.e., stream 1 is irrelevant and stream 2 is relevant). Thus, for each state i , we have a set of double-stream symbols where the streams have different degrees of relevance. Once the set of double-stream symbols is generated, a state transition probability distribution is generated, and the double-stream sequential data are generated using Algorithm 7.

2) *Results*: In the first experiment, we apply the baseline DHMM and the proposed MSDHMM structures to the single-stream sequential data where the features are generated from one homogeneous source of information. The MSDHMM architectures treat the single-stream sequential data as double-stream ones (each stream is assumed to have 2-D observation vectors). In this experiment, all models are trained using standard BW (for the baseline DHMM and distance-based MSDHMM) and the generalized BW (for the probability-based MSDHMM), the standard and generalized MCE/GPD

TABLE I
CLASSIFICATION RATES OF THE DIFFERENT DHMM
STRUCTURES OF THE SINGLE-STREAM DATA

Classifier	Baum-Welch	MCE	BW and MCE
Baseline DHMM	90.08%	90.5%	91.25%
MSDHMM ^D	91.075%	92.00%	93.75%
MSDHMM ^{P_a}	91.25%	92.25%	98.75%
MSDHMM ^{P_g}	90.25%	92.50%	95.75%

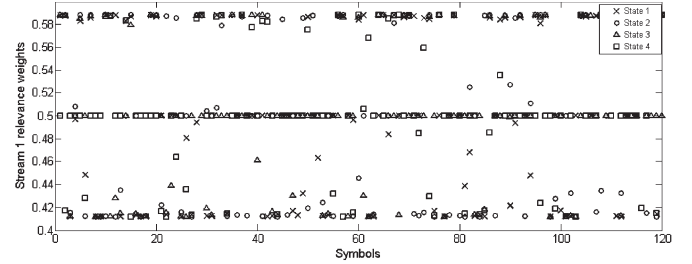


Fig. 2. Stream 1 relevance weights of the symbols in all four states, learned by the MSDHMM^{P₁} model for the single-stream sequential data. Since both streams were generated to be equally relevant for these data and since weights of the two streams must sum to one, all learned weights are clustered around 0.5.

algorithms, or a combination of the two (BW followed by MCE/GPD). The results of this experiment are reported in Table I. As it can be seen, the performance of the proposed MSDHMM structures and the baseline DHMM is 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 and the MSDHMM reduces to the baseline DHMM. Fig. 2 shows the weights of stream 1 in all four states. 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 weights of both streams must sum to one, both weights are equally important for all symbols.

The second experiment involves applying both the baseline DHMM and the proposed MSDHMM to the double-stream sequential data where the features are generated from two different streams. In this experiment, the various models were trained using BW, MCE, and BW followed by MCE training algorithms. First, we note that, using stream relevance weights, the generalized BW and MCE training algorithms converge faster and result in a small error. Fig. 3 shows the number of misclassified samples versus the number of iterations for the baseline DHMM and the proposed MSDHMM using MCE/GPD training. As it can be seen, learning the stream relevance weights causes the error to drop faster. In fact, at each iteration, the classification error for the MSDHMM structure is lower than that for the baseline DHMM. In particular, for the probability-based linear MSDHMM, the error reaches the minimum after only two iterations.

The testing results are reported in Table II. First, we note that all proposed MSDHMMs outperform the baseline DHMM 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 DHMM treats both streams equally important. The proposed MSDHMMs, on the other hand, learn the optimal relevance weights for each symbol

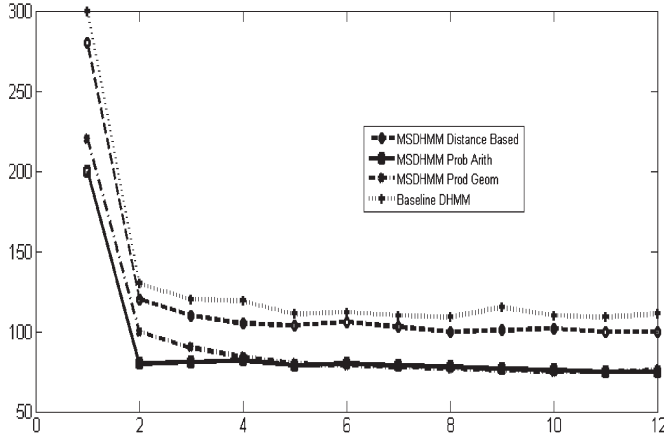


Fig. 3. Number of misclassified sequences versus MCE/GPD iteration number for the baseline DHMM and MSDHMM.

TABLE II
COMPARISON OF BW AND MCE ALGORITHMS FOR
THE DIFFERENT DHMM STRUCTURES

Classifier	Baum-Welch	MCE	BW and MCE
Baseline DHMM	54.075%	59.075%	60.025%
MSDHMM ^D	62.075%	64.075%	71.25%
MSDHMM ^{P_a}	60.25%	70.25%	72.65%
MSDHMM ^{P_g}	58.25%	65.25%	75.00%

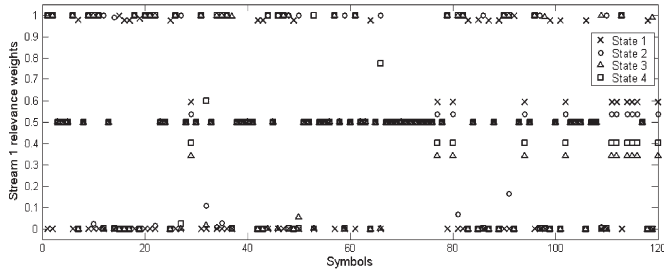


Fig. 4. Stream 1 relevance weights of the symbols in all four states learned by the MSDHMM^{P1} model for the double-stream sequential data. Weights close to 0.5 correspond to symbols where both streams are relevant. Weights close to one (or zero) correspond to symbols where stream 2 data (or stream 1) were replaced with random features.

within each state. The learned weights for stream 1 by the MSDHMM^{P1} are shown in Fig. 4. As it can be seen, some symbols are highly relevant (weights that are close to one) in some states, while others are completely irrelevant (weights that are close to zero). The latter ones correspond to the symbols where stream 1 features were replaced by noise in the data generation, and the former ones correspond to the symbols where stream 2 features were replaced by noise. Symbols with weights close to 0.5 correspond to those where neither streams were replaced by random noise.

From Table II, we also notice that the probability-based MSDHMMs outperform the distance-based MSDHMM. This can be attributed to two main factors. First, the MSDHMM^D learns an initial set of relevance weights and does not optimize these weights in the subsequent learning phase. Second, these weights are not state dependent. The results also indicate that using the generalized BW followed by the MCE to learn the

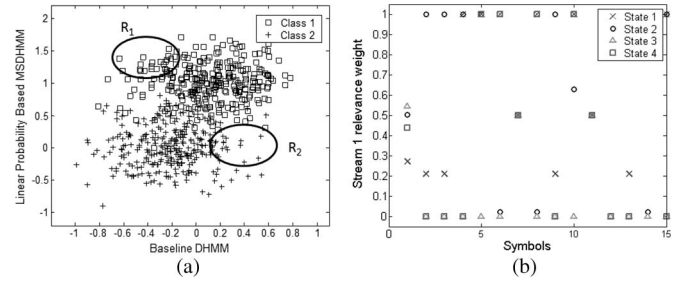


Fig. 5. (a) Scatter plot of the confidences of the two-class data in the baseline DHMM and the MSDHMM^{P1}. Region R_1 (R_2) highlights sequences from class 1 (class 2) that will most likely be classified correctly by MSDHMM^{P1} and misclassified by the baseline DHMM. (b) Stream 1 relevance weights of the closest symbols of a sequence in R_1 .

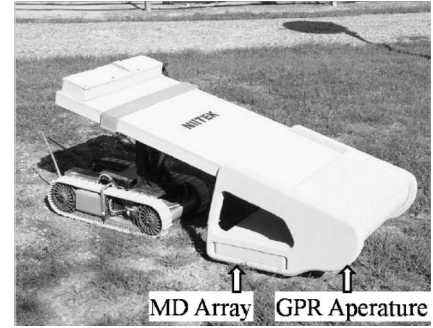


Fig. 6. NIITEK autonomous mine detection system.

model parameters is a better strategy. This is consistent with what has been reported for the baseline HMM [31].

To illustrate the advantages of the MSDHMM further, in Fig. 5, we show a scatter plot of the baseline DHMM versus the MSDHMM^{P1} confidence values. As it can be seen, the confidence values are highly correlated for most sequences. However, for few sequences (e.g., highlighted regions R_1 for class 1 and R_2 for class 2) the MSDHMM^{P1} outperforms the baseline DHMM. For instance, the MSDHMM^{P1} assigns relatively higher confidence values to the sequences highlighted in region R_1 . Since these sequences belong to class 1, they will be classified correctly by the MSDHMM^{P1} and misclassified by the baseline DHMM. Similarly, the MSDHMM^{P1} assigns relatively lower confidence values to the sequences highlighted in region R_2 , and these sequences will be classified correctly by the MSDHMM^{P1} and misclassified by the baseline DHMM. To verify that this difference is attributed to the learned relevance weights, in Fig. 5, we show the learned stream 1 relevance weights for the symbols associated with the 15 observations for one of the sequences in region R_1 . As it can be seen, only four symbols have equal relevance weights in all four states.

B. Application to Land-Mine Detection

In this experiment, we apply the proposed DHMM structures to the problem of land-mine detection using a GPR sensor.

1) *Data Collection:* We use data collected with a GPR mounted on a NIITEK, Inc., robotic mine detection system. This system includes a GPR and a WEMI sensor and is shown in Fig. 6. Each sensor collects data as the system moves. Only data collected by the GPR sensor are used in our experiments.

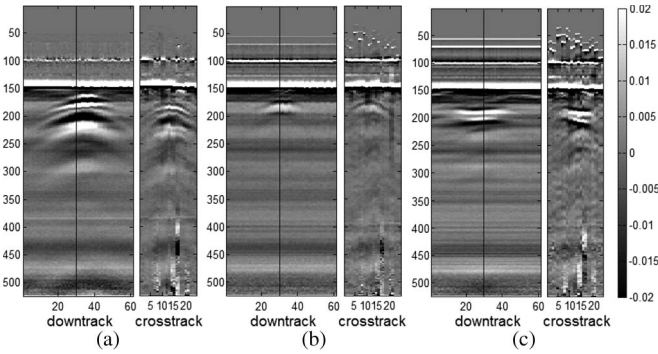


Fig. 7. NIITEK radar down-track and cross-track (at position indicated by a line in the down track) B-scan pairs for (a) an AT mine, (b) an AP mine, and (c) a nonmetal clutter alarm.

The GPR sensor [46] collects 24 channels of data. Adjacent channels are spaced approximately 6 cm apart in the cross-track direction, and sequences (or scans) are taken at approximately 1-cm down-track intervals. The system uses a V-dipole antenna that generates a wideband pulse ranging from 200 MHz to 7 GHz. Each A-scan, i.e., the measured waveform that is collected in one channel at one down-track position, contains 516 time samples at which the GPR signal return is recorded. Each sample corresponds to roughly 8 ps. We often refer to the time index as depth. Thus, we model an entire collection of input data as a 3-D matrix of sample values $S(z; x; y)$, where $z = 1, \dots, 516$, $x = 1, \dots, 24$, and $y = 1, \dots, T$ with T being the total number of collected scans and the indices z , x , and y representing the depth, cross-track position, and down-track positions, respectively.

The mine detection system (shown in Fig. 6) was used to acquire large collections of GPR data from two geographically distinct test sites with natural soil in the eastern U.S. The two sites are partitioned into grids with known mine locations. In all, there are 28 distinct mine types that can be classified into four categories: AT metal, AT with low metal content, AP metal, and AP with low metal content. The targets were buried up to 5 in deep. Multiple data collections were performed at each site resulting in a large and diverse collection of mine and clutter signatures.

For our experiment, we use a subset of the data collection that includes 600 mine and 600 clutter signatures. These signatures are first preprocessed to enhance detection. First, 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, which are up to few samples above the ground bounce, are discarded so that only the data corresponding to regions below the ground surface are processed.

Fig. 7 shows 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 channel response in a single sample) at the position indicated by a line in the down track. The objects scanned are a high-metal-content AT mine (a), a high-metal-content AP

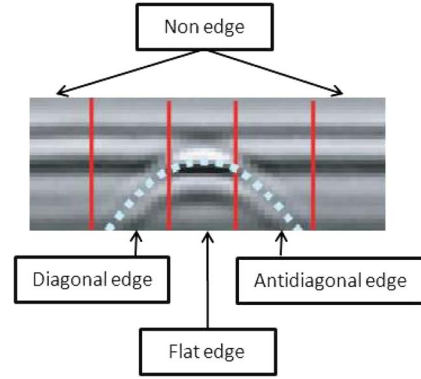


Fig. 8. Shape of a typical mine signature and the interpretation of the four states of the DHMM structure.

mine (b), and a wood block (c). 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. The preprocessing artifacts and the electromagnetic interference can add considerable amounts of noise to the signatures and make the detection problem more difficult.

2) *Feature Extraction*: As it can be seen in Fig. 8, land mines (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 DHMM-based system 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 Fig. 7, 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 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. Thus, each training signature s consists of $45(\text{depth}) \times 15(\text{scans}) \times 7(\text{channels})$ volume extracted from the aligned GPR data.

Fig. 8 shows 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 DHMM structure. State 1 corresponds to a nonedge activity (i.e., background), state 2 corresponds to a diagonal edge, state 3 corresponds to a flat edge, and state 4 corresponds to an antidiagonal edge.

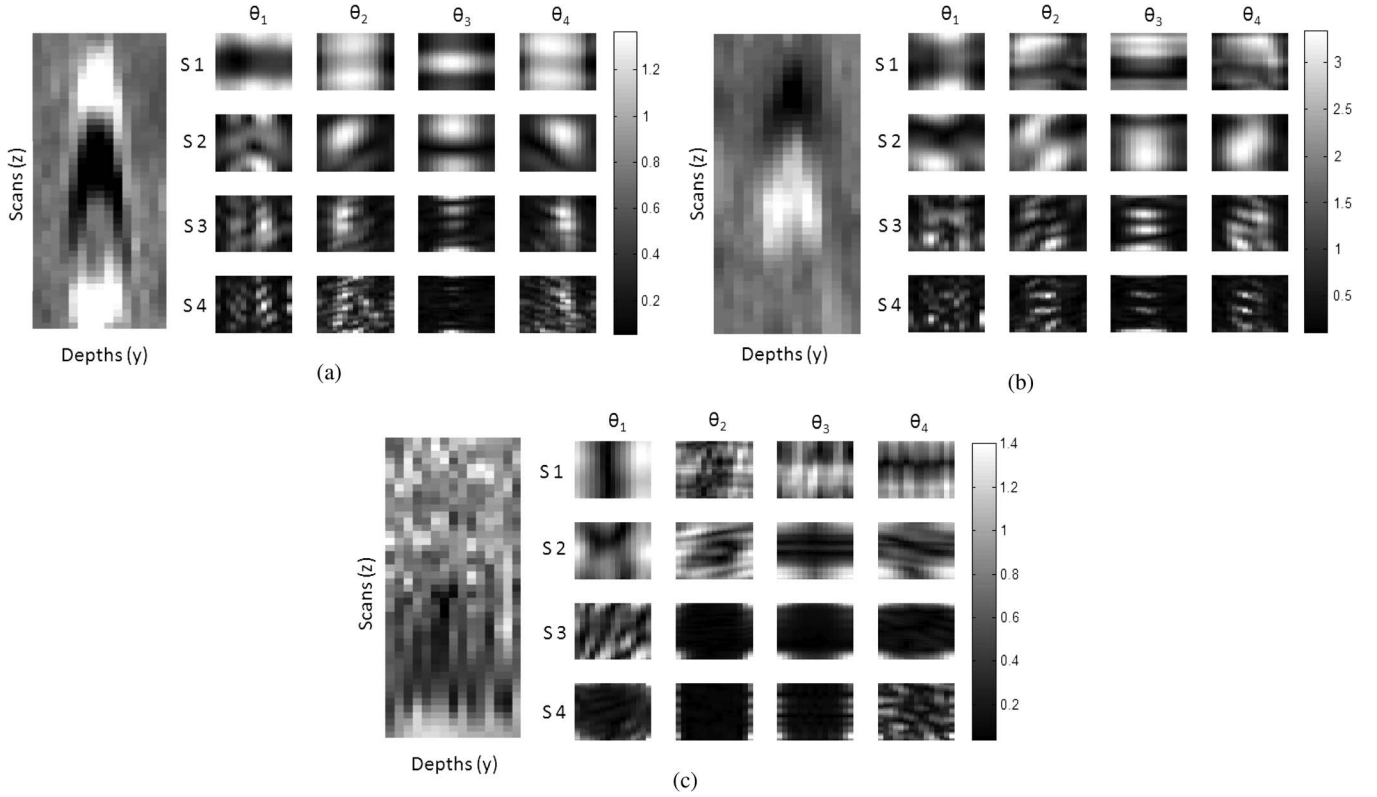


Fig. 9. Response of three alarms to the 16 Gabor filters at different scales and orientations. (a) Strong mine signature. (b) Weak mine signature. (c) Clutter signature with high energy.

For our application, to capture the spatial distribution of the edges within the 3-D GPR alarms, we adopt two feature extraction mechanisms: Gabor wavelets [26] and edge histogram descriptors (EHDs) [9].

a) Gabor feature extraction: We use the homogeneous texture descriptor [26] to capture the spatial distribution of the edges within the 3-D GPR alarms in the frequency domain. In particular, we extract features by expanding the signature's B-scan using a bank of scale and orientation selective Gabor filters. We fix the number of scales to four and the number of orientations to four at 45° intervals.

Let $S(x, y, z)$ denote the 3-D GPR data volume of an alarm. To keep the computation simple, we use 2-D filters (in the y - z plane) and average the response over the third dimension. Let $S_x(y, z)$ be the x th plane of the 3-D signature $S(x, y, z)$. Let $SG_x^{(k)}(y, z)$, $k = 1, \dots, 16$, denote the response of $S_x(y, z)$ to the 16 Gabor filters. Fig. 9(a) shows 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 (particularly 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 the signature has a strong response to the θ_3 (horizontal) filters (flat edge). Fig. 9(b) shows a weak mine signature and its response to the Gabor filters. For this signature, the edges are not as strong as those in Fig. 9(a). As a result, it has a weaker response at all scales (scale 2 has the strongest response), particularly for the falling edge. Fig. 9(c) shows a clutter signature (with high

energy) and its response. As it can be seen, this signature has a strong response to the θ_4 (135°) degree filters. However, this response is not localized on the right side of the signature as it is the case for most mine 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 point. The observation sequence of $S_x(y, z)$ at a fixed depth z is the sequence of 15 observation vectors

$$\mathbf{O}(x, y - 7, z), \mathbf{O}(x, y - 6, z), \dots, \mathbf{O}(x, y - 1, z), \mathbf{O}(x, y, z), \\ \mathbf{O}(x, y + 1, z), \dots, \mathbf{O}(x, y + 7, z) \quad (61)$$

where

$$\mathbf{O}(x, y, z) = [O^1(x, y, z), \dots, O^{16}(x, y, z)] \quad (62)$$

$$O^k(x, y, z) = \frac{1}{45} \sum_{z=1}^{45} SG_x^{(k)}(y, z) \quad (63)$$

encode the response of $S(x, y, z)$ to the k th Gabor filters.

b) EHDs: The EHDs [9] capture the salient properties of the 3-D alarms in a compact and translation-invariant representation. They extract edge histograms capturing the frequency

of occurrence of five types of edges: vertical, horizontal, 45° diagonal, 135° antidiagonal, and isotropic (nonedges) orientations in the data associated with a ground position. First, for each $S_x(y, z)$, we compute four categories of edge strengths: vertical, horizontal, 45° diagonal, 135° antidiagonal. If the maximum of the edge strengths exceeds a certain preset threshold θ , the corresponding pixel is considered to be an edge pixel. Otherwise, it is considered a nonedge pixel.

At each position (x, y, z) , we compute an observation vector $H_{zy}^x \in \mathbb{R}^5$ that represents a five-bin edge histogram. As with the Gabor features, we take the down-track dimension as the time variable, and at each position (x, y, z) , we define a sequence of 15 observation vectors as

$$O(x, y, z) = [\bar{H}_{zy-7}, \dots, \bar{H}_{zy}, \dots, \bar{H}_{zy+7}] \quad (64)$$

where \bar{H}_{zy_i} is the cross-track average of the edge histograms of subimage $S_x(y_i, z)$ over N_C channels, i.e.,

$$\bar{H}_{zy_i} = \frac{1}{N_C} \sum_{x=1}^{N_C} H_{zy}^x. \quad (65)$$

Using the Gabor and EHD features, we perform two different experiments to highlight the advantages of the proposed MSDHMM structures. First, we treat the Gabor features at each scale as different streams. We show that, by learning relevance weights for the different streams, the MSDHMM outperforms the baseline DHMM that treats features from the four scales equally important. In the second experiment, we compare the performance of the proposed MSDHMM, baseline DHMM, and baseline continuous HMM when Gabor features at scale 2 and EHD features are used as two streams.

3) *Learning HMM Parameters*: We construct and train multiple land-mine detectors using various HMM structures. Each detector has one model for clutter and another for mine. Each model produces a probability value by backtracking through model states using the Viterbi algorithm. The probability value produced by the mine (clutter) model can be thought of as an estimate of the probability of the observation sequence given that there is a mine (clutter) present.

For all HMM structures, we assume that each model has $N_s = 4$ states. The state representatives v_k are obtained by clustering the training data into four clusters using the FCM [44]. The learning procedures used for the other parameters depend on the HMM structures and are outlined hereinafter.

a) *Baseline DHMM*: For the baseline DHMM, we treat all features equally important and simply concatenate them to form a single feature vector. To generate the codebook, we cluster the training data into $M = 100$ clusters using the FCM [44]. The transition probabilities \mathbf{A} and the observation probabilities \mathbf{B} are then estimated using the BW algorithm [28], the MCE/GPD algorithm [31], or a combination of the two.

b) *MSDHMM*: The baseline HMM assumes that all features 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 Gabor 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. Similarly, some alarms may be better characterized by Gabor features while others may be better characterized by EHD features.

Since it is not possible to know *a priori* which feature is more discriminative, we propose considering the different features as different streams of information and use the training data to learn MSDHMMs (distance based and probability based). For the first experiment, we use four streams where each stream (Gabor response at a fixed scale) produces 4-D feature vectors (Gabor response at the different orientations). For the second experiment, we use two streams. The first one is a Gabor feature at scale 2, and the second one is the EHD feature. For both experiments, to generate the codebook, we cluster the training data in $M = 100$ clusters using SCAD [27] and learn initial stream relevance weights for each symbol. The state transition probabilities \mathbf{A} , the observation probabilities \mathbf{B} , and the stream relevance weights \mathbf{W} (for the case of probability-based MSDHMM) are learned using the generalized BW (see Section III-B), the generalized MCE/GPD (see Section III-B2b), or a combination of the two.

4) *Confidence Value Assignment*: The confidence value assigned to each observation sequence $\text{Conf}(O)$ depends on the following: 1) the probability assigned by the mine model $\Pr(O|\lambda^m)$; 2) the probability assigned by the clutter model $\Pr(O|\lambda^c)$; and 3) the optimal state sequence assigned by the mine model. In particular, if the optimal state sequence has at least one observation assigned to state 2 (rising edge) in the left half of the sequence and at least one observation assigned to state 4 (falling edge) in the right half, then the confidence is set to $\Pr(O|\lambda^m) - \Pr(O|\lambda^c)$. Otherwise, the confidence is simply set to zero. This condition is needed to prevent background sequences, where all observations are assigned to state 1, from having nonzero confidence values. These conditions were used in [22], and in this paper, we adopt them to all of the DHMM variations.

Since each alarm has over 500 depth values and only 45 depths are processed at a time, we divide the test alarm into ten overlapping subalarms and test each one independently to obtain ten partial confidence values. The average of the top three confidences is then taken as the final confidence value.

5) *Experimental Results*: We use a fivefold cross validation scheme to evaluate the proposed MSDHMM structures and compare them to the baseline DHMM. For each cross validation, we use a different subset of the data that have 80% of the alarms for training, and we test on the remaining 20% of the alarms. The scoring is performed in terms of the probability of detection (PD) versus the probability of false alarms (PFA). Confidence values are thresholded at different levels to produce the receiver operating characteristic (ROC) curve. For the probability-based DHMM with geometric aggregation, we set the values of ν and κ in (47) to 1.25 and 1, respectively. For the MCE/GPD training, the parameter of the sigmoid loss function was empirically chosen as $\zeta = 1$ and $\theta = 0$. In general, in the MCE training, the step size parameter ϵ needs to be carefully chosen to balance the learning rate and convergence behavior. A large ϵ leads to fast learning but may cause divergence, while

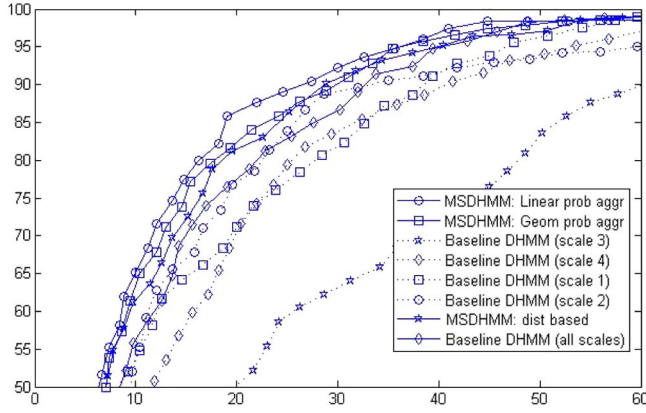


Fig. 10. Performance of the baseline DHMM and MSDHMM using Gabor features at four scales as four streams.

a small ϵ leads to slow learning but is safe in convergence. Our experiments revealed that the best step size ϵ was often data dependent, and it also depended on how well the baseline models fit the data. In this paper, we report the results when the step size is set to 10^{-3} for the first iteration and is increased by a step of 10^{-3} . It has been noticed that the number of iterations required for convergence is around 50.

The results of the first experiment are shown in Fig. 10. In this figure, we compare the ROC curves generated using each of the four streams (Gabor features at each scale) and their combination using simple concatenation (baseline DHMM) and using the MSDHMM. All results were obtained when the model parameters are learned using BW followed by the MCE/GPD training method. First, we note that the DHMM with Gabor features at scale 2 outperforms all other features (for $PFA \leq 40$). Second, the baseline DHMM with all four scales is not much better than the DHMM at scale 2. In fact, for some PFA, the performance can be even worse. This is due mainly to the way that the four scales are combined equally. Third, we note that all MSDHMM structures outperform the baseline DHMM. This is due to different stream relevance weights learned by the MSDHMM structures for the codebook symbols. In fact, the majority of the symbols have higher weights associated with Gabor features at scale 2 (since, globally, it is the best scale). However, there are few symbols where weights to other features are higher. Finally, we note that the MSDHMM with linear aggregation outperforms the other structures. These results are consistent with those obtained with the synthetic data.

The results of the second experiment are shown in Fig. 11. In this figure, we compare the ROC curves generated using the proposed MSDHMM structures and the baseline discrete and continuous HMMs when Gabor features at scale 2 and EHD features were used as two streams. First, we note that, for a high PD, all of the proposed MSDHMM structures outperform the baseline HMM. Second, the MSDHMM with linear aggregation outperforms the other structures. Third, for a PD in the range of $[70, 90]$, the continuous HMM outperforms the DHMM and two variations of the proposed MSDHMM. This may be due to the richer description (continuous pdf) used by this HMM.

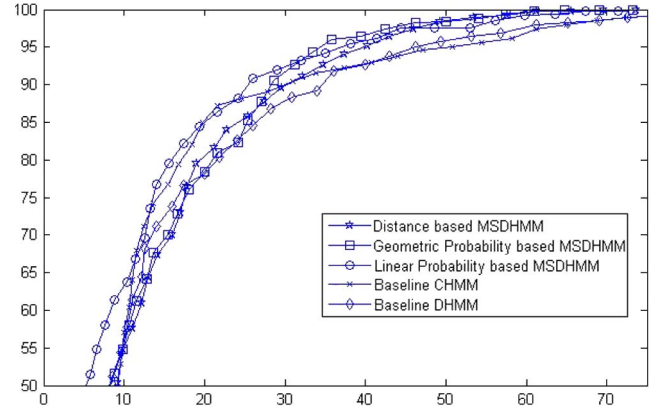


Fig. 11. Performance of the MSDHMM compared to the baseline discrete and continuous HMMs when Gabor features at scale 2 and EHD features were used as two streams.

To gain more insight into the behavior of the different DHMM structures, in Fig. 12, we compare the performance of the different algorithms when different categories of mines are considered. First, we note that the relative performance of the two individual features can vary significantly depending on the category of the target being considered. For instance, for high-metal mines, the EHD features perform much better than the Gabor features at all PDs. On the other hand, for low-metal mines, the EHD feature has a better performance at a high PD, and the Gabor feature has a better performance at a low PFA. A similar behavior can be observed when deep mines (depth > 3 in) and shallow mines are scored separately. Since the category of the target, its burial depth, or the properties of the soil are not known *a priori*, it is not possible to select the optimal features. This was the main motivation of our proposed MSDHMM structures. By assigning context-dependent relevance weights (weights to the different symbols in different regions of the aggregated feature space), the MSDHMM can take advantages of the good streams and yield results better than all single streams.

V. CONCLUSION

We have proposed novel MSHMM structures that integrate a stream relevance weighting component for the classification of temporal data. These structures allow learning symbol and state-dependent stream relevance weights. We have formulated two approaches: the distance-based MSDHMM and the probability-based MSDHMM. The first method involves an unsupervised learning step to learn the stream relevance weights, and the second one integrates the varying stream relevance weight into the overall observation probability of each symbol. This method aggregates the partial observation probabilities (relative to each stream) and their relevance weights. For both methods, we generalized the BW and MCE/GPD learning algorithms and derived the update equations for all model parameters. The results on a synthetic data set and a library of GPR signatures have shown that the proposed MSDHMM structures improve the discriminative power and, thus, the classification accuracy of the DHMM. The introduction of stream relevance weights also causes the training error to decrease faster and for the training algorithm to converge faster.

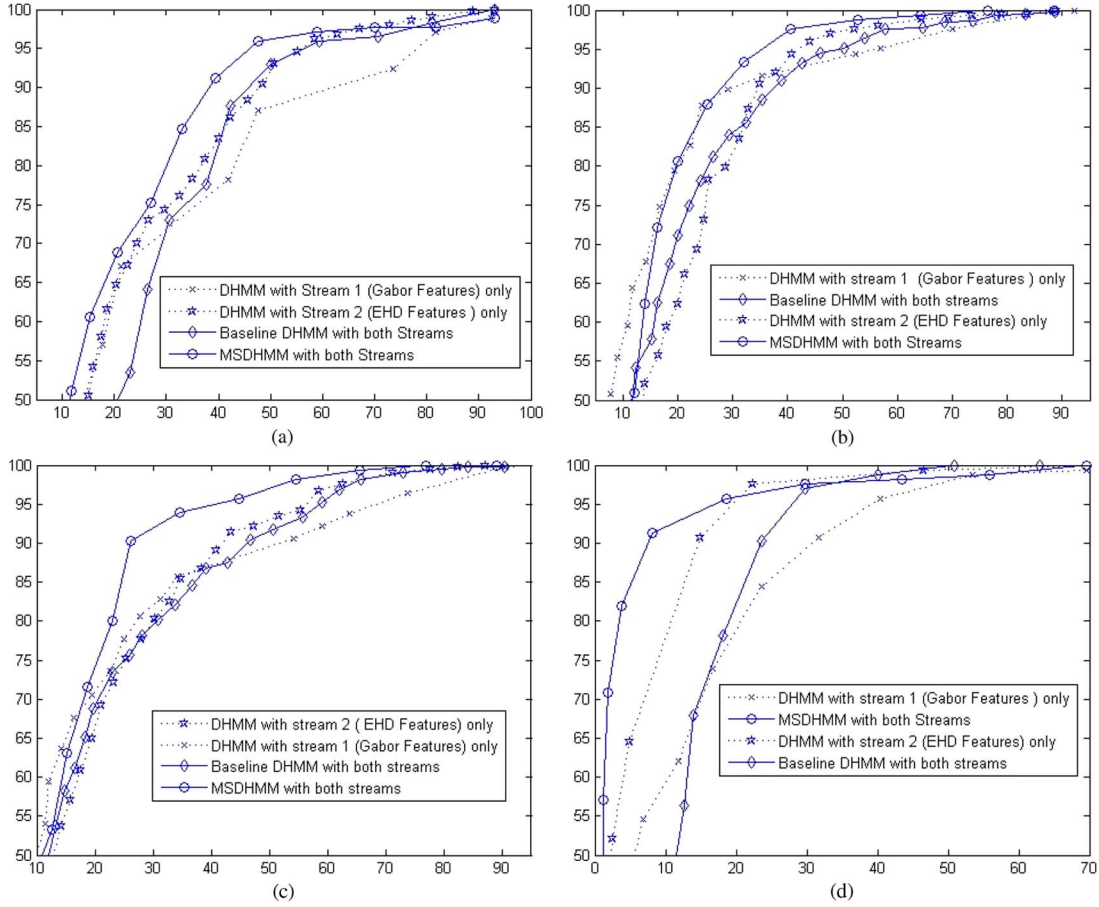


Fig. 12. Comparison of the ROC curves obtained using the DHMM and MSDHMM (with linear probabilities) when different categories of targets are considered.

In this paper, we have proposed the MSDHMM. A continuous version could be developed using a similar approach where stream relevance weights could be learned for the different components of each state. We are currently investigating this extension.

The discriminative training performed in this paper uses batch-mode training. Sequential training could be 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 paper could be extended to the Bayesian case that is relevant in situations where training data are limited. The application to land-mine detection could be extended to include streams from different feature extraction methods or even from different sensors. In particular, for the system shown in Fig. 6, we could use streams from the GPR sensor as well as streams from the WEMI sensor.

APPENDIX A GENERALIZED BW FOR THE MSDHMM^{P1}

To derive (39) and (40), we recall the objective function in (37). This objective function involves the quantity $\Pr(O, Q, F|\bar{\lambda})$ which could be expressed analytically as

$$\Pr(O, Q, F|\bar{\lambda}) = \bar{\pi}_{q_1} \prod_{t=1}^{T-1} \bar{a}_{q_t q_{t+1}} \prod_{t=1}^T \bar{w}_{q_t Q_V(o_t) f_t} \bar{b}_{q_t Q_V(o_t) f_t}. \quad (66)$$

Thus, the objective function in (37) expands as follows:

$$\begin{aligned} \mathbb{Q}(\lambda, \bar{\lambda}) = & \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{\pi}_{q_1}) \\ & + \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{w}_{q_t Q_V(o_t) f_t}) \\ & + \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{b}_{q_t Q_V(o_t) f_t}) \\ & + \sum_{t=1}^{T-1} \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{a}_{q_t q_{t+1}}). \quad (67) \end{aligned}$$

The second term in (75) can be expressed as follows:

$$\begin{aligned} & \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{w}_{q_t Q_V(o_t) f_t}) \\ & = \sum_{t=1}^T \sum_i \sum_j \sum_k \log(\bar{w}_{ijk}) \\ & \quad \times \sum_Q \sum_F \Pr(Q, F|O, \lambda) \delta(i, q_t) \delta(j, o_t) \delta(k, f_t) \quad (68) \end{aligned}$$

where $\delta(i, q_t) \delta(j, o_t) \delta(k, f_t)$ tells us to include only those cases for which $q_t = i$, $o_t = j$, and $f_t = k$. That is

$$\begin{aligned} \Pr(Q, F|O, \lambda) \delta(i, q_t) \delta(j, o_t) \delta(k, f_t) \\ = \Pr(q_t = i, f_t = k|O, \lambda) \delta(o_t, j) \quad (69) \end{aligned}$$

and (76) reduces to

$$\begin{aligned} & \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{w}_{q_t Q_V(o_t) f_t}) \\ &= \sum_{t=1}^T \sum_{i=1}^{N_s} \sum_{j=1}^M \sum_{k=1}^L \Pr(q_t = i, f_t = k|O, \lambda) \\ & \quad \times \delta(o_t, j) \ln(\bar{w}_{ijk}). \end{aligned} \quad (70)$$

To find the value of \bar{w}_{ijk} that maximizes the auxiliary function $\mathbb{Q}(\cdot, \cdot)$, only the expression in (78) needs to be considered since it is the only part of $\mathbb{Q}(\cdot, \cdot)$ that depends on \bar{w}_{ijk} . To find the update equation of \bar{w}_{ijk} , we use the Lagrange multiplier with the constraint in (31) and obtain

$$\bar{w}_{ijk} = \frac{\sum_{t=1}^T \Pr(q_t = i, f_t = k|O, \lambda) \delta(o_t, j)}{\sum_{t=1}^T \Pr(q_t = i|O, \lambda) \delta(o_t, j)} \quad (71)$$

where

$$\Pr(q_t = i, f_t = k|O, \lambda) = \Pr(q_t = i|O, \lambda) \frac{w_{iQ_V(o_t)k} b_{iQ_V(o_t)k}}{b_{iQ_V(o_t)}}. \quad (72)$$

Since $\Pr(q_t = i|O, \lambda) = \gamma_t(i)$ and noting $\gamma_t(i, k) = \Pr(q_t = i, f_t = k|O, \lambda)$, the update equation for \bar{w}_{ijk} becomes

$$\bar{w}_{ijk} = \frac{\sum_{t=1}^T \gamma_t(i, k) \delta(o_t, j)}{\sum_{t=1}^T \gamma_t(i) \delta(o_t, j)}. \quad (73)$$

Similarly, the rest of the parameters' update equations could be derived.

APPENDIX B GENERALIZED BW FOR THE MSDHMM^{Pg}

To derive (51) and (79), we recall the objective function in (37). This objective function involves the quantity $\Pr(O, Q, F|\bar{\lambda})$ which could be expressed analytically as

$$\Pr(O, Q, F|\bar{\lambda}) = \bar{\pi}_{q_1} \prod_{t=1}^{T-1} \bar{a}_{q_t q_{t+1}} \prod_{t=1}^T [\bar{b}_{q_t Q_V(o_t) f_t}]^{\bar{w}_{q_t Q_V(o_t) f_t}}. \quad (74)$$

Thus, the objective function in (37) expands as follows:

$$\begin{aligned} \mathbb{Q}(\lambda, \bar{\lambda}) &= \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{\pi}_{q_1}) \\ &+ \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \bar{w}_{q_t Q_V(o_t) f_t} \\ & \quad \times \log(\bar{b}_{q_t Q_V(o_t) f_t}) \\ &+ \sum_{t=1}^{T-1} \sum_Q \sum_F \Pr(Q, F|O, \lambda) \log(\bar{a}_{q_t q_{t+1}}). \end{aligned} \quad (75)$$

The second term in (75) can be expressed as follows:

$$\begin{aligned} & \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \bar{w}_{q_t Q_V(o_t) f_t} \log(\bar{b}_{q_t Q_V(o_t) f_t}) \\ &= \sum_{t=1}^T \sum_i \sum_j \sum_k \log(\bar{w}_{ijk}) \sum_Q \sum_F \Pr(Q, F|O, \lambda) \\ & \quad \times \delta(i, q_t) \delta(j, o_t) \delta(k, f_t) \end{aligned} \quad (76)$$

where $\delta(i, q_t) \delta(j, o_t) \delta(k, f_t)$ tells us to include only those cases for which $q_t = i$, $o_t = j$, and $f_t = k$. That is

$$\begin{aligned} & \Pr(Q, F|O, \lambda) \delta(i, q_t) \delta(j, o_t) \delta(k, f_t) \\ &= \Pr(q_t = i, f_t = k|O, \lambda) \delta(o_t, j) \end{aligned} \quad (77)$$

and (76) reduces to

$$\begin{aligned} & \sum_{t=1}^T \sum_Q \sum_F \Pr(Q, F|O, \lambda) \bar{w}_{q_t Q_V(o_t) f_t} \log(\bar{b}_{q_t Q_V(o_t) f_t}) \\ &= \sum_{t=1}^T \sum_{i=1}^{N_s} \sum_{j=1}^M \sum_{k=1}^L \Pr(q_t = i, f_t = k|O, \lambda) \delta(o_t, j) \\ & \quad \times \bar{w}_{q_t Q_V(o_t) f_t} \ln(\bar{b}_{ijk}). \end{aligned} \quad (78)$$

To find the value of \bar{w}_{ijk} that maximizes the auxiliary function $\mathbb{Q}(\cdot, \cdot)$, only the expression in (78) needs to be considered since it is the only part of $\mathbb{Q}(\cdot, \cdot)$ that depends on \bar{w}_{ijk} . To find the update equation of \bar{w}_{ijk} , we use the Lagrange multiplier with the constraint in (47) and obtain

$$\bar{w}_{ijk} = \begin{cases} \left[\kappa \frac{[D_{ijk}]^{\frac{\nu-1}{\nu}}}{\sum_{k=1}^L [D_{ijk}]^{\frac{\nu-1}{\nu}}} \right]^{\frac{1}{\nu}}, & \text{if } \exists t, \delta(o_t, j) = 1 \\ \left(\frac{\kappa}{L} \right)^{\frac{1}{\nu}}, & \text{otherwise} \end{cases} \quad (79)$$

where $D_{ijk} = \sum_{t=1}^T \gamma_t(i, k) \delta(o_t, j) \log(b_{ijk})$.

Similarly, the rest of the parameters' update equations could be derived.

APPENDIX C GENERALIZED MCE/GPD FOR MSDHMM^{P1}

In order to derive the update equations in (44) and (45), we need intermediate derivatives. The derivatives $\partial \mathbf{L}(\Lambda) / \partial \tilde{w}_{ijk}^{(c)}$ and $\partial \mathbf{L}(\Lambda) / \partial \tilde{b}_{ijk}^{(c)}$ could be expanded using the chain rule as follows:

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} &= \sum_{r=1}^R \sum_{m=1}^C \frac{\partial l_m(O, \Lambda)}{\partial d_m(O)} \times \frac{\partial d_m(O)}{\partial g_c(O, \Lambda)} \\ & \quad \times \frac{\partial g_c(O, \Lambda)}{\partial w_{ijk}^{(c)}} \times \frac{\partial w_{ijk}^{(c)}}{\partial \tilde{w}_{ijk}^{(c)}} \mathbb{I}(O \in C_c) \end{aligned} \quad (80)$$

$$\begin{aligned} \frac{\partial \mathbf{L}(\Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} &= \sum_{r=1}^R \sum_{m=1}^C \frac{\partial l_m(O, \Lambda)}{\partial d_m(O)} \times \frac{\partial d_m(O)}{\partial g_c(O, \Lambda)} \\ &\times \frac{\partial g_c(O, \Lambda)}{\partial b_{ijk}^{(c)}} \times \frac{\partial b_{ijk}^{(c)}}{\partial \tilde{b}_{ijk}^{(c)}} \mathbb{I}(O \in C_c). \end{aligned} \quad (81)$$

Both (44) and (45) use the following common derivatives:

$$\begin{aligned} \frac{\partial l_m(O, \Lambda)}{\partial d_m(O)} &= \zeta l_m(O, \Lambda) [1 - l_m(O, \Lambda)] \\ \frac{\partial d_m(O)}{\partial g_c(O, \Lambda)} &= \begin{cases} -1, & \text{if } c = m \\ \frac{\exp[\eta g_c(O, \Lambda)]}{\sum_{j, j \neq c} \exp[\eta g_j(O, \Lambda)]}, & \text{if } c \neq m. \end{cases} \end{aligned} \quad (82)$$

The following derivatives are needed for the stream relevance weights:

$$\begin{aligned} \frac{\partial g_c(O, \Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} &= \sum_{t=1}^T \delta(q_t = i, Q_V(o_t) = j) \frac{b_{ijk}^{(c)}}{b_{ij}^{(c)}} \\ \frac{\partial w_{ijk}^{(c)}}{\partial \tilde{w}_{ijk}^{(c)}} &= w_{ijk}^{(c)} [1 - w_{ijk}^{(c)}]. \end{aligned}$$

The following derivatives are needed for the partial observation probabilities:

$$\begin{aligned} \frac{\partial g_c(O, \Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} &= \sum_{t=1}^T \delta(q_t = i, Q_V(o_t) = j) \frac{w_{ijk}^{(c)}}{b_{ij}^{(c)}} \\ \frac{\partial b_{ijk}^{(c)}}{\partial \tilde{b}_{ijk}^{(c)}} &= b_{ijk}^{(c)} [1 - b_{ijk}^{(c)}]. \end{aligned}$$

APPENDIX D

GENERALIZED MCE/GPD FOR MSDHMM^{P1}

For the geometric case, the update equations for the stream relevance weights and the partial observation probabilities are conducted similarly to the linear case. The only difference figures in the following derivatives:

$$\begin{aligned} \frac{\partial g_c(O, \Lambda)}{\partial \tilde{w}_{ijk}^{(c)}} &= \sum_{t=1}^T \delta(q_t = i, Q_V(o_t) = j) \log b_{ijk}^{(c)} \\ \frac{\partial w_{ijk}^{(c)}}{\partial \tilde{w}_{ijk}^{(c)}} &= [\tilde{w}_{ijk}^{(c)}]^{\nu-1} w_{ijk}^{(c)} [1 - [w_{ijk}^{(c)}]^\nu] \\ \frac{\partial g_c(O, \Lambda)}{\partial \tilde{b}_{ijk}^{(c)}} &= \sum_{t=1}^T \delta(q_t = i, Q_V(o_t) = j) \frac{w_{ijk}^{(c)}}{b_{ij}^{(c)}}. \end{aligned}$$

REFERENCES

- [1] "Landmine Monitor Report 2009: Toward a Mine-Free World."
- [2] K. Ho and P. Gader, "A linear prediction land mine detection algorithm for hand held ground penetrating radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 40, no. 6, pp. 1374–1385, Jun. 2002.
- [3] Y. Tan, S. Tantum, and L. Collins, "Kalman filtering for enhanced landmine detection using quadrupole resonance," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 7, pp. 1507–1516, Jul. 2005.
- [4] D. Potin, E. Duflos, and P. Vanheeghe, "Landmines ground-penetrating radar signal enhancement by digital filtering," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 9, pp. 2393–2406, Sep. 2006.
- [5] T. Savelyev, L. Van Kempen, H. Sahli, J. Sachs, and M. Sato, "Investigation of time frequency features for GPR landmine discrimination," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 1, pp. 118–129, Jan. 2007.
- [6] L. Collins, P. Gao, and L. Carin, "An improved Bayesian decision theoretic approach for land mine detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 811–819, Mar. 1999.
- [7] W. Ng, T. Chan, H. So, and K. Ho, "Particle filtering based approach for landmine detection using ground penetrating radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 11, pp. 3739–3755, Nov. 2008.
- [8] H. Frigui, L. Zhang, and P. D. Gader, "Context-dependent multisensor fusion and its application to land mine detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 6, pp. 2528–2543, Jun. 2010.
- [9] H. Frigui and P. Gader, "Detection and discrimination of landmines in ground-penetrating radar based on edge histogram descriptors and a possibilistic K-nearest neighbor classifier," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 1, pp. 185–199, Feb. 2009.
- [10] K. C. Ho, L. Carin, P. D. Gader, and J. N. Wilson, "On using the spectral features from ground penetrating radar for landmine/clutter discrimination," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 4, pp. 1177–1191, Apr. 2008.
- [11] J. N. Wilson, P. Gader, W.-H. Lee, H. Frigui, and K. C. Ho, "A large-scale systematic evaluation of algorithms using ground-penetrating radar for landmine detection and discrimination," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 8, pp. 2560–2572, Aug. 2007.
- [12] D. Carevic, "Kalman filter-based approach to target detection and target-background separation in ground-penetrating radar data," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets IV*, Orlando, FL, 1999, pp. 1284–1288.
- [13] A. Gunatilaka and B. Baertlein, "Subspace decomposition technique to improve GPR imaging of anti-personal mines," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets V*, Orlando, FL, 2000.
- [14] H. Brunzell, "Detection of shallowly buried objects using impulse radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 875–886, Mar. 1999.
- [15] S. Yu, R. Mehra, and T. Witten, "Automatic mine detection based on ground penetrating radar," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets IV*, Orlando, FL, 1999, pp. 961–972.
- [16] H. Frigui, K. Satyanarayana, and P. Gader, "Detection of landmines using fuzzy and possibilistic membership functions," in *Proc. IEEE Conf. Fuzzy Syst.*, St. Louis, MO, 2003, pp. 834–839.
- [17] D. Carevic, "Clutter reduction and target detection in ground penetrating radar using wavelets," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets IV*, Orlando, FL, 1999.
- [18] P. Gader, M. Mystkowski, and Y. Zhao, "Landmine detection with ground penetrating radar using hidden Markov models," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 6, pp. 1231–1244, Jun. 2001.
- [19] S. Tantum, Y. Wei, V. Munshi, and L. Collins, "A comparison of algorithms for landmine detection and discrimination using ground penetrating radar," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets*, Orlando, FL, 2002, pp. 728–735.
- [20] P. Gader, B. Nelson, H. Frigui, G. Vaillette, and J. Keller, "Fuzzy logic detection of landmines with ground penetrating radar," *Signal Process.—Special Issue on Fuzzy Logic in Signal Processing*, vol. 80, no. 6, pp. 1069–1084, Jun. 2000.
- [21] P. Gader, W. Lee, and J. Wilson, "Detecting landmines with ground penetrating radar using feature-based rules, order statistics, and adaptive whitening," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 11, pp. 2522–2534, Nov. 2004.
- [22] H. Frigui, K. Ho, and P. Gader, "Real-time landmine detection with ground-penetrating radar using discriminative and adaptive hidden Markov models," *EURASIP J. Appl. Signal Process.*, vol. 2005, no. 12, pp. 1867–1885, 2005.
- [23] H. Frigui, O. Missaoui, and P. Gader, "Landmine detection with ground penetrating radar using discrete hidden Markov models with symbol dependent features," in *Proc. SPIE Defense Security Symp.*, Orlando, FL, Mar. 2008.
- [24] O. Missaoui and H. Frigui, "Optimal feature weighting for discrete HMM," in *Proc. Int. Conf. Pattern Recogn.*, Tampa, FL, Dec. 2008, pp. 1–4.
- [25] Y. Zhao, P. Gader, P. Chen, and Y. Zhang, "Training DHMMs of mine and clutter to minimize landmine detection errors," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 5, pp. 1016–1024, May 2003.
- [26] H. Frigui, O. Missaoui, and P. Gader, "Landmine detection using discrete hidden Markov models with Gabor features," *Proc. SPIE*, vol. 6553, p. 655 32A, Apr. 27, 2007.

- [27] H. Frigui and S. Salem, "Fuzzy clustering and subset feature weighting," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2003, pp. 857–862.
- [28] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
- [29] X. Li, M. Parizeau, and R. Plamondon, "Training hidden Markov models with multiple observations—A combinatorial method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 4, pp. 371–377, Apr. 2000.
- [30] A. Nadas, "A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-31, no. 4, pp. 814–817, Aug. 1983.
- [31] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 5, no. 3, pp. 257–265, May 1997.
- [32] G. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.
- [33] P. Gerasimos, N. Chalapathy, L. Juergen, and M. Iain, "Audio-visual automatic speech recognition: An overview," in *Audio-Visual Speech Processing*, E. Vatikiotis-Bateson, G. Bailly, and P. Perrier, Eds. Cambridge, MA: MIT Press, 2009.
- [34] M. Masaru, S. Iori, N. Masafumi, H. Yasuo, and K. Shingo, "Sign language recognition based on position and movement using multi-stream HMM," in *Proc. 2nd Int. Symp. Universal Commun.*, 2008, pp. 478–481.
- [35] N. Atta, S. Sid-Ahmed, T. Hesham, and O. Douglas, "Incorporating phonetic knowledge into a multi-stream HMM framework," in *Proc. Can. Conf. Elect. Comput. Eng.*, 2008, pp. 001 705–001 708.
- [36] Y. Kessentini, T. Paquet, and A. M. Benhamadou, "A multi-stream approach to off-line handwritten word recognition," in *Proc. 9th ICDAR*, 2007, pp. 317–321.
- [37] Z. Wu, L. Cai, and H. Meng, "Multi-level fusion of audio and visual features for speaker identification," in *Proc. ICB*, 2006, pp. 493–499.
- [38] C. Chibelushi, J. Mason, and F. Deravi, "Feature level data fusion for bimodal person recognition," in *Proc. ICIP*, 1997, pp. 399–403.
- [39] V. Chatziz, A. Bors, and I. Pitas, "Multimodal decision level fusion for person authentication," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 29, no. 6, pp. 674–680, Nov. 1999.
- [40] Z. Ghahramani and M. Jordan, "Factorial hidden Markov models," in *Proc. NIPS*, 1995, pp. 472–478.
- [41] A. Nefian, L. Liang, T. Fu, and X. Liu, "A Bayesian approach to audio-visual speaker identification," in *Proc. 5th Int. Conf. Audio Video-Based Biometric Person Authentication*, 2003, pp. 761–769.
- [42] S. Dupont and J. Luetin, "Audio-visual speech modeling for continuous speech recognition," *IEEE Trans. Multimedia*, vol. 2, no. 3, pp. 141–151, Sep. 2000.
- [43] J. Hartigan, *Clustering Algorithms*. Hoboken, NJ: Wiley, 1975.
- [44] J. Bezdek, *Pattern Recognition With Fuzzy Objective Function Algorithms*. New York: Plenum, 1981.
- [45] Z. Ghahramani and M. Jordan, "Factorial hidden Markov models," *Mach. Learn.*, vol. 29, no. 2/3, pp. 245–273, Nov./Dec. 1997.
- [46] K. J. Hintz, "SNR improvements in NITEK ground penetrating radar," in *Proc. SPIE Conf. Detection Remediation Technol. Mines Minelike Targets IX*, Orlando, FL, Apr. 2004, pp. 399–408.



Hichem Frigui (S'92–M'98) received the Ph.D. degree in computer engineering and computer science from the University of Missouri, Columbia, in 1997.

From 1998 to 2004, he was an Assistant Professor with The University of Memphis, Memphis, TN. He is currently a Professor and the Director of the Multimedia Research Laboratory with the University of Louisville, Louisville, KY. He has been active in the research fields of fuzzy pattern recognition, data mining, and image processing with applications to content-based multimedia retrieval and land-mine

detection. He has participated in the development, testing, and real-time implementation of several land-mine detection systems. He has published over 135 journal and refereed conference articles.

Dr. Frigui was a recipient of the National Science Foundation Career Award for outstanding young scientists.



Paul Gader (M'87–SM'99) received the Ph.D. degree in mathematics from the University of Florida, Gainesville, in 1986.

He was a Senior Research Scientist with Honeywell Systems and Research Center, a Research Engineer and Manager with the Environmental Research Institute of Michigan, and a Faculty Member with the University of Wisconsin Oshkosh, Oshkosh, and the University of Missouri, Columbia. He is currently a Professor of computer and information science and engineering with the University of Florida.

He led teams involved in real-time handwritten address recognition systems for the U.S. Postal Service and teams that devised and tested several real-time algorithms in the field of mine detection. He has approximately 260 technical publications in the areas of image and signal processing, applied mathematics, and pattern recognition, including approximately 77 refereed journal articles. His research interests include hyperspectral and light detection and ranging image and signal analysis, land-mine detection, handwriting recognition, machine learning/pattern recognition, fuzzy and random sets, and Choquet integration.



Oualid Missaoui (S'98–M'00) received the B.E. degree in signal and systems and the M.S. degree in applied mathematics from the Polytechnic School of Tunisia, La Marsa, Tunisia, in 2003 and 2005, respectively, and the Ph.D. degree in computer science and engineering from the University of Louisville, Louisville, KY.

From August 2003 to December 2005, he was a Design and Verification Engineer with STMicroelectronics. In January 2006, he was with the University of Louisville, where he did research in the fields of

machine learning, land-mine detection, and image processing. Since March 2010, he has been with Pipeline Financial Group, Inc., New York, NY, where he is part of the research team and is in charge of developing a data-mining-and-pattern-recognition-based algorithmic trading mechanism. His research interests include data mining, machine learning, pattern recognition, evolutionary algorithms, stochastic calculus, computational finance, econophysics, and multimedia.