Weakly-supervised Dictionary Learning

Zeyu You, Student member, IEEE, Raviv Raich, Senior Member, IEEE,
Xiaoli Z. Fern, Member, IEEE, and Jinsub Kim, Member, IEEE

Abstract—We present a probabilistic modeling and inference framework for discriminative analysis dictionary learning under a weak supervision setting. Dictionary learning approaches have been widely used for tasks such as low-level signal denoising and restoration as well as high-level classification tasks, which can be applied to audio and image analysis. Synthesis dictionary learning aims at jointly learning a dictionary and corresponding sparse coefficients to provide accurate data representation. This approach is useful for denoising and signal restoration, but may lead to sub-optimal classification performance. By contrast, analysis dictionary learning provides a transform that maps data to a sparse discriminative representation suitable for classification. We consider the problem of analysis dictionary learning for time-series data under a weak supervision setting in which signals are assigned with a global label instead of an instantaneous label signal. We propose a discriminative probabilistic model that incorporates both label information and sparsity constraints on the underlying latent instantaneous label signal using cardinality control. We present the expectation maximization (EM) procedure for maximum likelihood estimation (MLE) of the proposed model. To facilitate a computationally efficient E-step, we propose both a chain and a novel tree graph reformulation of the graphical model. The performance of the proposed model is demonstrated on both synthetic and real-world data.

Index Terms—weakly-supervision learning, convolutive analysis dictionary, chain inference, tree inference.

I. INTRODUCTION

Synthesis dictionary learning, in which the dictionary and a corresponding sparse representation are jointly learned from data, aims at minimizing a reconstruction error. An overcomplete synthesis dictionary learning with sparse coding is introduced in [1]. Various state-of-the-art approaches have been introduced to solve the dictionary learning problem including K-SVD [2], matrix factorization [3], Lagrangian dual gradient descent and feature-sign search [4]. For audio or spectral image analysis, convolutive dictionary learning has been proposed [5]–[7]. Analysis dictionary learning and transform learning offer an alternative to dictionary learning [8], [9], which produces a sparsified outcome after applying the analysis dictionary to the original data. As stated in [8], the analysis approach shows a significant advantage over synthesis and other denoising approach in terms of signal recovery for random, piecewise-constant and natural signal data. Having linear dependencies between sets of rows in the dictionary improves the recovery quality of the pursuit algorithm. The linear dependencies can be incorporated by forcing sparse and zero-mean rows during the training. Both synthesis and analysis dictionary learning are used for solving problems such as reconstruction, denoising, and sparse coding. However, without an additional supervision component, such methods have been reported to perform sub-optimally in classification tasks [10]. This is due to the fact that both approaches aim at reconstruction rather than classification. Nevertheless, both approaches can be applied to classification by modifying the objective to include a label fit term that renders the learned dictionary as discriminative as possible. A more detailed discussion on supervised learning approaches is provided in subsection II-C.

In this paper, we consider a weak-supervision setting for analysis dictionary learning that is suitable for classification. In the weak-supervision setting, for each data portion containing multiple data points, a label set describing the classes present or absent is provided while the individual label of each instance remains unavailable. A motivating example for this setting is the problem of in-situ bio-acoustic monitoring, where audio recordings of varying signal-to-noise ratio (SNR) are collected by unattended microphones. Even in fairly short intervals, it is common to have multiple simultaneous vocalizations from multiple species in addition to other noise sources such as wind, rain, stream, and nearby vehicles. Consequently, intervals of audio data are associated with multiple class labels. Since the audio signals are multi-labeled at the interval level, the precise location and class of each pattern contained in the signal are unknown. Hence, isolating an individual pattern or absent is provided while the individual label of each instance remains unavailable. A motivating example for this setting is the problem of in-situ bio-acoustic monitoring, where audio recordings of varying signal-to-noise ratio (SNR) are collected by unattended microphones. Even in fairly short intervals, it is common to have multiple simultaneous vocalizations from multiple species in addition to other noise sources such as wind, rain, stream, and nearby vehicles. Consequently, intervals of audio data are associated with multiple class labels. Since the audio signals are multi-labeled at the interval level, the precise location and class of each pattern contained in the signal are unknown. Hence, isolating an individual pattern or assigning the appropriate label as a training example in the traditional supervised learning setting is not trivial.

Discriminative dictionary learning approaches have been considered when the full label information is available, i.e. each data example is associated with a label (binary or multiclass) [11], [12]. For weakly-supervised setting, learning a dictionary given a set of observed signals/images and their binary labels has been considered where non-convolutive, synthesis dictionary learning approaches are introduced [13]–[15]. To the best of our knowledge, such approaches cannot be easily extended to the convolutive analysis dictionary learning setting.

This work focuses on dictionary learning under weak supervision, where we learn a dictionary given a set of signals and their label sets. Focusing on time-series, we propose an algorithm for weakly-supervised convolutive analysis dictionary learning. Our contributions are as follows. (i) We develop a novel discriminative probabilistic model for analysis dictionary learning under the weak-supervision setting. (ii) We present an alternative approach for cardinality (or sparsity) constraints.

*All authors are with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, 97331-5501, USA e-mail: [youz, raich, xfern, kimjinsu]@oregonstate.edu. This work is partially supported by the National Science Foundation grants CCF-1254218, DBI-1356792 and IIS-1055113.
as implicit observations in a graphical model as opposed to commonly used norm regularization. This approach allows for localization of the patterns-of-interest. (iii) We introduce a novel framework for efficient message passing using a reformulation of the proposed graphical model both as a chain and as a tree. This reformulation yields a near-linear exact probability calculation that alleviates the need for approximate inference. Our preliminary work on this topic was introduced in [16].}

II. BACKGROUND AND RELATED WORK

A. Generative synthesis dictionary learning

In synthesis dictionary learning [1], [2], [17], the goal is to simultaneously find a dictionary and corresponding coefficients to represent a set of \( n \) signals \( x_1, x_2, \ldots, x_n \in \mathbb{R}^m \). A dictionary is a collection of atoms \( \mathbf{D} = \{ \mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_K \} \), where \( \mathbf{d}_k \in \mathbb{R}^m \) is the \( k \)th dictionary atom (or word). The \( i \)th signal can be approximated by a linear combination over the dictionary \( \mathbf{D} \) by

\[
\mathbf{x}_i \approx \mathbf{D} \alpha^i = \sum_{k=1}^{K} \mathbf{d}_k \alpha_{ik}, \quad \text{for } i = 1, 2, \ldots, n,
\]

where \( \alpha^i = [\alpha_{i1}, \ldots, \alpha_{iK}]^T \) is the coefficient vector associated with the \( i \)th signal. Synthesis dictionary learning is typically formulated as an optimization problem, where the goal is to find \( \mathbf{D} \) and sparse coefficients \( \{\alpha^i\}_{i=1}^n \) that minimize the reconstruction error.

Convolutional dictionary learning is often considered [18], [19] for time invariant signals such as speech and audio. In convolutional dictionary learning, the \( i \)th signal \( x_i \) is assumed to be formed by combining the convolution of dictionary words \( \mathbf{d}_1, \ldots, \mathbf{d}_K \), where \( \mathbf{d}_k \in \mathbb{R}^m \), with their corresponding sparse activation signals \( \alpha_1^i, \ldots, \alpha_K^i \):

\[
\mathbf{x}_i \approx \sum_{k=1}^{K} \mathbf{d}_k * \alpha_k^i, \quad \text{for } i = 1, 2, \ldots, n.
\]

In this approach, a signal may contain multiple time-shifted copies of the same dictionary signal, convolutional dictionary learning eliminates the need to use additional dictionary words to model multiple shifts of the same dictionary word. The joint recovery of the dictionary and the sparse activation signals can be achieved by minimizing the reconstruction error subject to sparsity constraints on the activation signals.

B. Analysis dictionary learning

In analysis dictionary learning [8], [9], given a signal \( x_i \), we are looking for an analysis dictionary \( \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K] \in \mathbb{R}^{m \times K} \) and an estimated noiseless signal \( \mathbf{x}_i' \) (\( x_i \approx \mathbf{x}_i' \)) such that the resulting analyzed signal

\[
\mathbf{W}^T \mathbf{x}_i' = [\mathbf{w}_1^T \mathbf{x}_i', \mathbf{w}_2^T \mathbf{x}_i', \ldots, \mathbf{w}_K^T \mathbf{x}_i']^T
\]

\[\text{is sparse. For a noisy signal model, the analysis dictionary learning can be formulated as minimizing a quadratic error between all } x_i \text{'s and } x_i' \text{'s subject to the resulting analyzed signals are sparse.}

Convolutional analysis dictionary learning convolves the analysis dictionary \( \mathbf{W} \) with estimated noiseless signals \( x_1', x_2', \ldots, x_n' \) so that the resulting signals \( w_1 * x_1', \ldots, w_K * x_1', \ldots, w_1 * x_n', \ldots, w_K * x_n' \) are sparse [20], [21].

C. Discriminative dictionary learning

Several approaches have been proposed for dictionary learning in the presences of labels. For supervised dictionary learning, the following approaches have been proposed: (i) Learn one dictionary per class [22]–[25]; (ii) Prune large dictionaries [27], [28]; (iii) Jointly learn dictionary and classifier [11], [29]–[31]; (iv) Embed class labels into the learning of sparse coefficients [32]–[36], and (v) Learn a histogram of dictionary elements over signal constituents [23], [27]–[43].

For weakly supervised dictionary learning, several max margin based, non-convolutive, synthesis dictionary learning approaches are proposed [13]–[15]. Other approaches propose to learn a discriminative synthesis dictionary by fully exploiting visual attribute correlations rather than label priors [44], [45]. In our preliminary work [16], a convolutive analysis dictionary learning approach under weak-supervision has been proposed. Here, we expand on the approach of [16]. We simplify the previous graphical model and re-derive all formulations. In addition, we propose a novel inference approach with a tree reformulation of the graphical model allowing for a near linear processing time of the probabilistic calculations.

III. PROBLEM FORMULATION AND SOLUTION APPROACH

Throughout the paper, we denote signals in lower case, e.g., \( x(t) \) or \( y(t) \). Similarly, we use lower case letters to represent indexes such as \( i \) or \( j \). For simplicity, we omit the dependence on time for signals, e.g., we also use \( x \) to denote signal \( x(t) \). In some cases, we use the time-evaluation operator \([\cdot]_t\) to denote evaluation of a signal at time \( t \), e.g., \( x|_t = x(t) \). We denote vectors by boldfaced lower case, e.g., \( \mathbf{x} \) or \( \mathbf{y} \). We use upper case letters to denote sets, e.g., \( Y \), or constants such as \( C \). All signals in this paper are assumed to be in discrete time. Consequently, we use the convolution operator \(*\) to denote discrete time convolution \( w * x|_t = \sum_{u=-\infty}^{\infty} x(t-u)w(u) \).

A. Problem statement

In the sparsity-driven setting of convolutive analysis dictionary learning, we are given a set of discrete-time signals \( \{x_1, x_2, \ldots, x_N\} \) and look for an analysis dictionary
\{w_1, w_2, \ldots, w_C\} such that for each \(n \in \{1, 2, \ldots, N\}\), each analysis signal of \(x_n\) in \(\{w_1 * x_n, w_2 * x_n, \ldots, w_C * x_n\}\) is sparse [20], [21]. Without loss of generality, we assume that the support of \(x_n\) is included in \(\{0, \ldots, T_n - 1\}\), i.e., \(x_n(t) = 0\) if \(t \notin \{0, \ldots, T_n - 1\}\).

In the fully-supervised setting, for each \(x_n\), a potentially sparse time-instance label signal \(y_n\) is provided and the observed data is of the form \(\{(x_1, y_1), (x_2, y_1), \ldots, (x_N, y_N)\}\). The signal \(y_n\) can be viewed as a fine-grain label indicating the presence of particular class patterns at each point in time. In this setting, location of a pattern from a given class can be used to extract examples for that class to train a classifier.

In the weak supervision setting, \(y_n\) is unknown and we are interested in learning a discriminative version of the convolutional analysis dictionary given the observed data. Under this setting, every signal \(x_n\) is accompanied with a single label set \(Y_n\) containing the classes that are present in signal \(x_n\), e.g., \(Y_n = \{2, 6\}\) if \(x_n\) contains patterns from only classes 2 and 6. Hence the data provided in our setting is of the form \(\{(x_1, Y_1), (x_2, Y_2), \ldots, (x_N, Y_N)\}\). Our goal in this setting, is two-fold: (i) to develop a time-instance-level classifier that predicts the latent label signal \(y(t)\) value for a previously unseen signal \(x(t)\) based on training data \(D\); and (ii) to develop a signal-level classifier that predicts the label set \(Y\) for a previously unseen signal \(x(t)\) based on training data \(D\). We proceed with the proposed formulation of this problem using a probabilistic model approach.

B. Probabilistic graphical model

To solve the weakly-supervised dictionary learning problem, we present a probabilistic model for learning a discriminative convolutional dictionary. We begin by introducing the convolution used in our model and proceed with a graphical representation of the proposed discriminative convolutional dictionary learning model.

Convolutional model: To simplify the exposition of the model using vectors instead of signals, we convert each signal \(x_n\) to a set of \(T_n + T_w - 1\) vectors such that each vector is a \(T_w\) width windowed portion of the signal. For simplicity, we assume \(T_w\) to be odd and denote \(\Delta = (T_w - 1)/2\). This notation allows us to replace the convolution \(w_c * x_n|_t\) with \(w_c^T x_{nt}\) for \(t = -\Delta, -\Delta + 1, \ldots, T_n - 1 + \Delta\) such that

\[
x_n * w_c|_t = \sum_{\tau=-\Delta}^{\Delta} x_n(t-\tau)w_c(\tau) = x_{nt}^T w_c,
\]

where \(x_{nt} \in \mathbb{R}^{T_w}\) is defined as

\[
x_{nt} = [x_n(t+\Delta), x_n(t+\Delta-1), \ldots, x_n(t-\Delta)]^T,
\]

and \(w_c \in \mathbb{R}^{T_w}\) is given by

\[
w_c = [w_c(-\Delta), w_c(-\Delta+1), \ldots, w_c(\Delta)]^T.
\]

The aforementioned one-dimensional signal model can be extended to a two-dimensional signal model in which convolution analysis dictionary is applied only on the time dimension. In the two-dimensional setting, \(x_n\) denotes \(x_n(f, t)\) and the \(c\)-th analysis dictionary word signal \(w_c\) denotes \(w_c(f, t)\). Using a 2-D window with size \(F \times T_w\), the analysis signal \(x_n * w_c\) with the time-convolution of the two signal is given by

\[
x_n * w_c|_t = \sum_{f=1}^{F} \sum_{\tau=-\Delta}^{\Delta} x_n(f, t-\tau)w_c(f, \tau) = x_{nt}^T w_c,
\]

where each windowed portion of the signal is

\[
x_{nt} = [x_n(1, t+\Delta), x(1, t+\Delta-1), \ldots, x(F, t-\Delta)]^T,
\]

and

\[
w_c = [w_c(1, -\Delta), w_c(1, -\Delta+1), \ldots, w_c(F, \Delta)]^T.
\]

While it is possible to develop a model that can handle boundary effects, such models are not time-invariant and hence may not benefit from the simplicity of the convolutional model.

Figure 2: The proposed graphical model for WSCADL

1) Model assumptions: To develop our model, we introduce additional assumptions to the aforementioned setting to explain the link between the analysis result and the signal label. Specifically, we assume that

A.1 Convolutional instance labeler: for each signal \(x_n\), a hidden discrete-value label signal \(y_n(t) \in \{0, 1, \ldots, C\}\) is produced given only the analysis result at time \(t\), i.e., the probability \(P(y_n(t) = c|x_n)\) depends on signal \(x_n\) only through \([w_0 * x_n|_t, w_1 * x_n|_t, \ldots, w_C * x_n|_t]\), the analysis result evaluated at time \(t\):

\[
P(y_n(t) = c|x_n) = f_c(w_0 * x_n|_t, \ldots, w_C * x_n|_t)
\]

for \(c = 0, 1, \ldots, C\), where \(f_c\) is an arbitrary multivariate function such that \(f_c: \mathbb{R}^{C+1} \rightarrow [0, 1]\) and \(\sum_c f_c = 1\).

A.2 Sparse instance label signals: the instance label signal \(y_n\) is sparse with the number of nonzero values at most \(\hat{N}_n\):

\[
\sum_{t=-\Delta}^{T_n-1+\Delta} \mathbb{I}(y_n(t) \neq 0) \leq \hat{N}_n.
\]

A.3 Signal label union assumption: the signal label \(Y_n\) is produced by taking the union of all the nonzero values of \(y_n\)

\[
Y_n = \bigcup_{t=-\Delta}^{T_n-1+\Delta} \{y_n(t)\}
\]

Note that this assumption makes it possible to have an empty signal label set in the case that all instantaneous labels are zero and no positive class is associated with any time instance. For simplicity, we remove the braces of \(y_n(t)\) and change to \(y_n(t)\) to represent a set of union of time instances.
2) Model description: The probabilistic graphical model for the weakly-supervised convolutive analysis dictionary learning (WSCADL) is shown in Figure 2 in which our target is to estimate the model parameters \( \theta = [w, b] \) and \( b = [b_0, b_1, \ldots, b_C]^T \). As explained earlier, the latent label signal at time \( t \) given by \( y_n(t) \) depends on the entire signal \( x_n \) through the convolution \( w_c \cdot x_n \) for \( c = 0, 1, \ldots, C \) evaluated at time \( t \) and hence through signal window \( x_{nt} \). The probabilistic model for \( y_n(t) \) follows the multinomial logistic regression model given by:

\[
P(y_n(t) = c|x_n; w, b) = \frac{e^{w_c^T x_{nt} + b_c}}{\sum_{u=0}^{C} e^{w_u^T x_{nt} + b_u}},
\]

for \( c = 0, 1, \ldots, C \), where \( w_c \) is the \( c \)th analysis word and \( b_c \) is a scalar bias term for the logistic regression model.

To encode the notion of sparsity in the instance label \( y_n(t) \), we introduce class 0 following the novel class concept of [40]. To integrate a constraint on the number of non-zero instances in the \( n \)th signal (i.e., the sparsity of \( y_n(t) \)) into our probabilistic model, we introduce the latent random variable \( I_n \), an indicator that takes the value 1 if the number of non-zero \( y_n(t) \) is less than or equal a sparsity upper bound \( N_n \) and zero otherwise. We treat \( N_n \) as a tuning (or hyper-) parameter of the graphical model. The smaller the value of \( N_n \), the sparser the label signal \( y_n(t) \) is. The probability model for sparsity indicator \( I_n \) of label signal \( y_n(t) \) is given by

\[
P(I_n = 1|y_n; N_n) = \frac{1}{(\sum_{\Delta = -\Delta}^{\Delta = 1} 1)(y_n(t) = 0) \leq N_n}.
\]

Using this notation, the sparsity constraint can be encoded as observing \( I_n = 1 \).

Since the class 0 is not represented in the signal label set, to obtain the signal label \( Y_n \) from \( y_n(-\Delta), \ldots, y_n(T_n - 1 + \Delta) \), we consider two possibilities. First, if the label signal \( y_n(t) \) does not contain zeros then we expect \( Y_n = \cup \{y_n(t)\} \). Alternatively, if the label signal \( y_n(t) \) contains zeros then we expect \( Y_n \cup \{0\} = \cup \{y_n(t)\} \). Consequently, the corresponding probabilistic model for the signal label \( Y_n \) is given by:

\[
P(Y_n|y_n) = \prod_{t=-\Delta}^{T_n-1+\Delta} \left( \mathbb{I}(y_n(t) = 0) \right) + \mathbb{I}(y_n(t) = y_n(t))
\]

C. Model parameter estimation

Given the parametric representation of our proposed model it is natural to consider estimating the model parameter using a maximum likelihood estimation (MLE). Since the model contains hidden variables, we adopt an expectation-maximization (EM) framework to facilitate the MLE estimator. We continue with the derivation of the complete and incomplete data likelihood.

1) Complete and incomplete data likelihood: Define the observed data as \( \mathcal{D} = \{X, Y, I_1, \ldots, I_N = 1\} \), the hidden data as \( \mathcal{H} = \{y_1, \ldots, y_N\} \), the unknown parameters as \( \theta = [w^T, b^T]^T \), and the tuning parameters as \( \phi = [N_1, \ldots, N_N]^T \).

According to the graphical model shown in Figure 2, the complete data likelihood \( P(\mathcal{D}, \mathcal{H}; \theta, \phi) \) is computed as

\[
P(X) = \prod_{n=1}^{N} \prod_{y_n=0}^{T_n+1-\Delta} \left( \mathbb{I}(y_n=0) \right) + \mathbb{I}(y_n(t) = y_n(t))
\]

and

\[
P(I_n = 1|y_n; N_n) = \prod_{t=-\Delta}^{T_n-1+\Delta} P(y_n(t)|x_{nt}; w, b).
\]

The incomplete data likelihood is calculated by marginalizing out the hidden variables as

\[
L(\theta) = \sum_{y_n(-\Delta) = 0} \ldots \sum_{y_n(T_n + \Delta) = 0} \mathbb{I}(y_n(t) = 0)
\]

Taking the natural logarithm of \( L(\theta) \) and plugging in the probability with \( (4) \) produces the incomplete log-likelihood:

\[
l(\theta) = \log P(X) + \sum_{n=1}^{N} \log \left( \mathbb{I}(y_n(-\Delta) = 0) \ldots \mathbb{I}(y_n(T_n + \Delta) = 0) \right)
\]

Calculating the incomplete data likelihood in \( (5) \) involves enumerating all possible instance labels, which is computationally intractable especially when the number of instance per signal is large.

2) Expectation maximization: Exact inference which computes the likelihood in a brute force manner by marginalizing over all instance value combinations is intractable. To resolve this, we consider an expectation maximization (EM) approach [47], where the proposed approach is very similar to the one in [46]. Specifically, the EM algorithm alternated between the expectation over the hidden variable and maximization of the auxiliary function as the following two steps:

**E-step:** Compute \( Q(\theta; \theta^0) = E_{\mathcal{H}|\mathcal{D}, \theta} \log P(\mathcal{H}, \mathcal{D}; \theta) \).

**M-step:** \( \theta^{t+1} = \arg \max \theta \) \( Q(\theta; \theta^0) \).

The auxiliary function for the proposed model is given by

\[
Q(\theta; \theta^0) = \sum_{n=1}^{N} \sum_{t=\Delta}^{T_n+1-\Delta} \sum_{c=0}^{C} \mathbb{I}(y_n(t) = c|\mathcal{D}; N_n, \theta^0) 
\]

The derivation of the auxiliary function for our model is explained in Appendix A. The maximization of the auxiliary function \( Q(\theta; \theta^0) \) provides an update rule for both dictionary words \( w \) and bias terms \( b \) with a learning rate \( \gamma \):

\[
w_{c}^{t+1} = w_{c}^{t} + \gamma \frac{\partial Q(\theta; \theta^0)}{\partial w_{c}} \big|_{\theta = \theta^t},
\]

\[
b_{c}^{t+1} = b_{c}^{t} + \gamma \frac{\partial Q(\theta; \theta^0)}{\partial b_{c}} \big|_{\theta = \theta^t},
\]
To guarantee the boundedness of the solution, we add an L$_2$-regularization term $-\lambda_r/2 \sum_{c=0}^C \|w_c\|^2$ in the M-step.

3) Key challenge: The computation of the gradient in (7) and (8) involves the computation of the posterior probability $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \theta^i)$ for each $y_n$ signal. This term presents one of the challenges of EM inference for the proposed model. Brute-force calculation requires marginalization over all other instance level labels, i.e., $y_n(s)$ for $s \neq t$. This marginalization is exponential in the number of instances per signal and hence makes the brute-form calculation prohibitive. In the following, we present the proposed efficient approach for calculating the posterior instance level label probability.

IV. GRAPHICAL MODEL REFORMULATION FOR THE E-STEP

The goal of the E-step is to obtain the posterior probability $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \theta^i)$, which first requires the calculation of the prior probability. To efficiently compute the prior probability $P(y_n(t)|x_n; w, b)$ as a function of $t$ for each signal $x_n$, $C + 1$ convolutions of the form $w_c \ast x_n$ are performed to obtain the values of $w_c^T x_{nt}$ in (1) for $t = -\Delta, -\Delta + 1, \ldots, T_n - 1 + \Delta$. Under some settings, the fast Fourier transform (FFT) and its inverse can be used as a computationally efficient implementation of the convolution.

We proceed with two efficient procedures for calculating the posterior probability of $y_n(t)$ given the prior probabilities.

A. Chain model reformulation

Consider the label portion of the original graphical model in Figure 3(a). Enumerating over the set of all possible values of $(y_n(-\Delta), \ldots, y_n(T_n - 1 + \Delta))$ to compute the posterior is exponential with respect to the number of time instances. The v-structure graph in Figure 3(a) does not offer an immediate efficient approach for computing the posterior. Hence, we propose a reformulation of the model as follows. We denote

$$Y_n^t = \bigcup_{i=-\Delta}^{t} \{y_n(i)\}, \quad N_n^t = \sum_{i=-\Delta}^{t} \mathbb{I}(y_n(i) \neq 0)$$

as the label set and the number of non-zero class instances of the first $t$ instances in $n$th signal. Both of the aforementioned newly introduced variables follow a recursive rule

$$Y_n^{t+1} = Y_n^{t} \cup \{y_n(t + 1) \mid y_n(t + 1) \neq 0\} \quad (9)$$
$$N_n^{t+1} = N_n^{t} + \mathbb{I}(y_n(t + 1) \neq 0) \quad (10)$$

Regularization: To guarantee the boundedness of the solution, we add an $L_2$-regularization term $-\lambda_r/2 \sum_{c=0}^C \|w_c\|^2$ in the M-step.
for \( t = -\Delta, \ldots, T_n - 1 + \Delta - 1 \). This reformulation gives rise to the chain model in Figure 3(b). We proceed with a linear time procedure for calculating the posterior probabilities that takes advantage of the reformulation of our model as a chain.

**Forward and backward message passing on the chain:**

Given the prior probability of \( y_n(t) = c \) for \( c \in \{0, 1, \ldots, C\} \) and \( t = -\Delta, \ldots, T_n - 1 + \Delta \), our goal is to obtain the posterior probability of \( y_n(t) = c \). This can be done by first computing the joint probability defined by 
\[
P_{nte} = P(y_n(t) = c, Y_n, I_n | x_n; \tilde{N}_n, \theta^t)
\]
and then applying Bayes rule as
\[
P(y_n(t) = c | y_n, x_n, I_n; \tilde{N}_n, \theta^t) = \frac{P_{nte}}{\sum_{c=0}^{C} P_{nte}}. \tag{11}
\]

We compute the joint probabilities \( P_{nte} \) using a dynamic programming approach that is summarized in the following three steps:

**Step 1: Forward message passing.** In this step, the goal is to compute the joint state probability \( P(Y_n, I_n = 1 | Y_n, N_n^t, x_n; \theta^t, \tilde{N}_n) \) for \( t = -\Delta, \ldots, T_n - 1 + \Delta \). Denote an element in the power set of all class labels in the \( n \)th signal by \( \mathbb{L} \in 2^{Y_n} \). A forward message is defined as
\[
\alpha_t(\mathbb{L}, l) \triangleq P(Y_n^t = \mathbb{L}, N_n^t = l | x_n; \theta^t).
\]
The first message is initialized as Figure 4(a) first step shows
\[
\alpha_1(\mathbb{L}, l) = \begin{cases} 
P(y_n(-\Delta) = 0 | x_n; w^i, b^i), & l = 0, \mathbb{L} = \emptyset; \\
P(y_n(-\Delta) = c | x_n; w^i, b^i), & l = 1, \mathbb{L} = \{c\}, c \in Y_n; \\
0, & \text{Otherwise},
\end{cases}
\]
The update equation for the forward message of the \( t \)th instance is calculated by marginalizing over the \((t-1)\)th state and the \( t \)th instance label as Figure 4(a) update step shows:
\[
\alpha_t(\mathbb{L}, l) = \alpha_{t-1}(\mathbb{L}, l) P(y_n(t) = 0 | x_n; w^i, b^i) + \sum_{c=1}^{C} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{L}_{(c \neq 0)}) \alpha_{t-1}(\mathbb{L} \setminus c, l). \tag{12}
\]
In the final step,
\[
P(Y_n = \mathbb{L}, I_n = 1 | x_n; \theta^t) = \sum_{l=1}^{N_n^*} \alpha_{t-1}(\mathbb{L}, l).
\]

**Step 2: Backward message passing.** In this step, the goal is to compute the conditional joint state probability defined as
\[
P(Y_n, I_n = 1 | Y_n, N_n^t, x_n; \theta^t, \tilde{N}_n).
\]
We denote a backward message as
\[
\beta_t(\mathbb{L}, l) \triangleq P(Y_n, I_n = 1 | Y_n^t = \mathbb{L}, N_n^t = l, x_n; \theta^t, \tilde{N}_n).
\]
According to the graphical model in Figure 4(b) such that \( Y_n, I_n \) is only dependent on \( Y_n^t, N_n^t \), the first backward message is initial at
\[
\beta_{T_n-1+\Delta}(\mathbb{L}, l) = \mathbb{I}_{(1 \leq \tilde{N}_n)}(\mathbb{L} = Y_n).
\]
The update equation for the \( t-1 \)th backward message is calculated by marginalizing over the \( t \)th backward message as Figure 4(b) update step shows
\[
\beta_{t-1}(\mathbb{L}, l) = \sum_{c=0}^{C} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{L}_{(c \neq 0)}) P(y_n(t) = c | x_n; w^i, b^i) \tag{13}
\]

**Properties:** To understand the range that should be used in computing the joint probability, we examine the values for which the forward and the backward messages are non-zero. The forward and backward messages for \( t = -\Delta, \ldots, T_n - 1 + \Delta \) have the following properties: (i) \( \alpha_t(\mathbb{L}, l) = 0 \) for \( l = t + 1 \), (ii) \( \beta_t(\mathbb{L}, l) = 0 \) for \( l > \tilde{N}_n \), \( \mathbb{L} \notin 2^{Y_n} \). Where (i) is from the definition of \( N_n^* \) in (10) and (ii) is from the sparsity constraint in (2) and the definition of \( Y_n^t \) in (9) such that each \( Y_n^{t-1} \subseteq Y_n^t \), and \( Y_n^{t-1+\Delta} = \mathbb{L} \).

**Step 3: Joint probability.** Finally, the numerator of (11) for \( t = -\Delta, \ldots, T_n - 1 + \Delta \) is computed using all of the forward messages and the backward messages as
\[
P(y_n(t) = c, Y_n, I_n = 1 | x_n; \theta^t, \tilde{N}_n) = P(y_n(t) = c | x_n; w^i, b^i) \cdot \sum_{\mathbb{L} \in 2^{Y_n}} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{L}_{(c \neq 0)}) \alpha_{t-1}(\mathbb{L}, l). \tag{14}
\]
where \( N_n^* = \min(\tilde{N}_n - \mathbb{L}_{(c \neq 0)}, t) \). Since \( Y_n^{t-\Delta}, N_n^{t-\Delta} \) is only dependent on the first instance \( y_n(-\Delta) \) as Figure 4(b) shows,
\[
P(y_n(-\Delta) = c, Y_n, I_n = 1 | x_n, \tilde{N}_n, \theta^t) = \beta_{-\Delta}(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{L}_{(c \neq 0)}) P(y_n(1) = c | x_n; w^i, b^i).
\]

**Note:** Based on property (i) that \( \alpha_{t-1}(\mathbb{L}, l) = 0 \) when \( l > t \), and property (ii) that \( \beta_t(\mathbb{L}, l) = 0 \) when \( l > \tilde{N}_n \), the
effective calculation and actual need of storing both forward and backward message is for $0 \leq l \leq \min(\bar{N}_n, t)$.

**Step 1: Forward message passing.** In this step, the goal is to compute the joint state probabilities $P(Y_{nt}, N_{nt}|x_n; \theta^i)$ for all $t = 1, 2, \ldots, 2^j$ and $0 \leq j \leq L$. Denote an element in the power set of all class labels in $n$th signal by $L \in 2^{Y_n}$. The forward message is defined as

$$
\alpha^j_t(L, l) \triangleq P(Y_{nt} = l, N_{nt} = \emptyset | x_n; \theta^i).
$$

At the leaf level, the forward messages are initialized as $\alpha^j_t(\emptyset, l) = \begin{cases} 
1, & l = 0, L = \emptyset; \\
0, & \text{Otherwise},
\end{cases}$

for $t = 1, 2, \ldots, T_n + T_w - 1$, and

$$
\alpha^j_t(L, l) = \begin{cases} 
1, & l = 0, L = \emptyset; \\
0, & \text{Otherwise},
\end{cases}
$$

for $t = T_n + T_w, T_n + T_w + 1, \ldots, 2^L$. The update for the forward message of the $t$th node in $j$-th level is calculated by marginalizing over its left parent (the $(2t - 1)$th message in $j$th level) and the right parent ($(2t)$th message in $j$th level) as

$$
\alpha^j_t(L, l) = \sum_{A \subseteq L} \sum_{B \subseteq L} \beta^j_t(A, a) \alpha^j_{2t-1}(B, l - a).
$$

We summarize the forward message step in Figure 6(a).

**Step 2: Backward message passing.** In this step, the goal is to compute the joint state posterior probability $P(Y_{nt}, n = 1|Y_{nt}, N_{nt}, x_n; \theta^i, \bar{N}_n)$. We denote a backward message as $\beta^j_t(L, l) \triangleq P(Y_{nt}, I_n = 1|Y_{nt} = l, x_n; \theta^i, \bar{N}_n)$. The first backward message is initialized as $\beta^0_t(L, l) = \mathbb{I}(l \subseteq \bar{N}_n) \mathbb{I}(l = \emptyset)$. The update equation for the backward messages are calculated as follows:

$$
\beta^j_{2t-1}(A, a) = \sum_{E \subseteq 2^{Y_n}} \beta^{j-1}_{t}(A \cup E, a + e) \alpha^j_{2t}(E, e).
$$

We summarize the backward message step in Figure 6(b).

Note: To efficiently calculate and store the forward and backward messages, we consider the following results: (i) $\alpha^j_t(L, l) = 0$ for $l > 2^L - j + 1$. (ii) $\beta^j_t(L, l) = 0$ for $l > N_n$ or $L \notin 2^{Y_n}$ for $j = 0, 1, \ldots, L$ and $t = 1, 2, \ldots, 2^L$. Where (i) is obtained from the recursive formula of $N_{nt}^j$ in (16) with the initialization of $N_{nt}^j$ and (ii) is obtained from the sparsity constraint in (2) and the definition of $Y_{nt}^j$ in (15) such that each $Y_{nt}^j \subseteq Y_{nt/2}^j$ and $Y_{n1}^0 = L$. Based on summary (i) that $\alpha^j_t(L, l) = 0$ when $l > 2^L - j + 1$, and (ii) that $\beta^j_t(L, l) = 0$ when $l > N_n$, the effective calculation and

---

**B. Tree model reformulation**

When both the cardinality constraints $\bar{N}_n$ and size of signal $T_n$ are large, chain model reformulation become computational-wise inefficient, since the complexity for both time and space grows as $\bar{N}_n \times T_n$ increases. Instead, we propose a complete full binary tree (denoted as $T(S_{nt}, j)$ with depth $L + 1$, where $j$ indicates the tree level, $S_{nt}^j$ is the node variable at index $t$ in $j$th level and $L = \left\lceil \log_2(T_n + T_w - 1) \right\rceil$) graph structure reformulation of the original graphical model in Figure 3(a) to make the E-step calculation more efficient.

In the complete full binary tree structure, each node of the tree $S_{nt}^j$ is considered as the joint state node ($Y_{nt}^j$ and $N_{nt}^j$). We denote $Y_{nt}^j$ as the label set of all ancestors of node $t$ in level $j$ of the tree and $N_{nt}^j$ as the number of non-zero class instances of all ancestors of node $t$ in level $j$ of the tree. We present the recursive relation At the leaf’s level, we assign the values as:

$$
Y_{nt}^j = \{y_n(t - \Delta - 1)|y_n(t - \Delta - 1) \neq 0\},
$$

$N_{nt}^j = \#(y_n(t - \Delta - 1) \neq 0)$

for $t = 1, 2, \ldots, T_n + T_w - 1$, and $Y_{nt}^L = \emptyset$, $N_{nt}^L = 0$ for $t = T_n + T_w, T_n + T_w + 1, \ldots, 2^L$. The relationship between the child node and its left parent node and its right parent node is using the following recursive formulas:

$$
Y_{nt}^{j-1} = Y_{nt}^{j}(2t-1) \cup Y_{nt}^{j}(2t)\}
$$

and

$$
N_{nt}^{j-1} = N_{nt}^{j}(2t-1) + N_{nt}^{j}(2t)
$$

for $t = 1, 2, \ldots, 2^j - 1$. This reformulation gives rise to the tree model in Figure 5. Note that $Y_{n1}^0 = \bigcup_{t} \{y_n(t)|y_n(t) \neq 0\} = Y_n$ and $N_{n1}^0 = \sum_{t} \#(y_n(t) \neq 0)$ which is used to determine $I_n$.

**Forward and backward message passing on the tree:**

In the tree inference, the target is the same as the chain inference as to compute the posterior probability of $y_n(t) = c$ for $c \in \{0, 1, \ldots, C\}$ and $t = -\Delta, \ldots, T_n - 1 + \Delta$. Using a dynamic programming approach, the joint probabilities $P(y_n(t), Y_n, I_n|x_n; \bar{N}_n, w^i)$ can be computed efficiently with the following three steps:

![Graphical model reformulation as a tree](image-url)
actual storing of both forward and backward message is for $0 \leq l \leq \min(N_n, 2^L - 1)$.

Step 3: Joint probability. Finally, the numerator on the RHS of (11) for $t = 1, \ldots, T_n$ is computed using the backward message $\beta_l^T(L, l)$ such that

$$P(y_n(t) = c, Y_n, I_n = 1|x_n; \theta^t, \bar{N}_n) = \beta_T^t((c), \|_{c \neq 0}) \cdot p(y_n(t) = c|x_n; w^t, b^t).$$

(20)

Convolutional model on tree: Based on update equation of the forward message in (17), if we treat each $\alpha_l^T$ message of a particular set value $L$ as a discrete signal $\alpha_l^T(L, t)$, then the update of each forward message is performing a convolution between $\alpha_{2t-1}^T(A, t)$ and $\alpha_{2t}^T(B, t)$. For the update on the backward message in (18) and (19), the update of backward message signal $\beta_{2t-1}^T(A, t)$ is a convolution between $\beta_{2t}^T(A \cup E, t)$ and $\alpha_{2t}^T(E, t)$ and the update of backward message signal $\beta_{2t}^T(E, t)$ is a convolution between $\beta_{2t}^T(A \cup E, t)$ and $\alpha_{2t-1}^T(A, t)$.

C. Complexity analysis

The complexity analysis can be divided into three parts (i) prior calculation, (ii) posterior calculation in E-step, and (iii) gradient calculation in M-step. We evaluate both prior probability and gradient update by forming $(C + 1) \times F$ number of convolutions in the time domain for the $n$th signal. Therefore, the time complexity for both (i) and (iii) is $O(N_n^C(C + 1)FT_nT_w)$. The space complexity is $O((C + 1)F(T_n + T_w - 1))$ and $O((C + 1)F(T_n + T_w))$ respectively.

On the posterior calculation of the E-step, the chain forward and backward messages require to run over all possible values of $y_n(t)$ and the previous state values of $Y_n^{t-1} \in 2^Y_n$ and $0 \leq N_n^{t-1} \leq \bar{N}_n$. Therefore the overall time and space complexity is $O\left(\sum_{n=1}^{N_n}Y_n\{2^{Y_n}\}T_n + T_w\bar{N}_n\right)$ and $O\left(2^{Y_n}\{T_n + T_w\}\bar{N}_n\right)$ respectively. To formulate the tree forward and backward messages, we need to run over all possible values of the previous two parents’ states. Therefore the resulting time and space complexity is $O\left(\sum_{n=1}^{N_n}Y_n\{2^{Y_n}\}(T_n + T_w)(\log_2 \bar{N}_n)^2\right)$ and $O\left(2^{Y_n}\{T_n + T_w\}(\log_2 \bar{N}_n)\right)$ respectively.

D. Prediction

In addition to identifying the analysis words $w_n$, the discriminative model allows for the prediction of time instance labels $y_n(t)$ for both labeled and unlabeled signals as well for the prediction of the signal label. Given a test signal $x_n^{test}(t)$ for $t = 1, 2, \ldots, T_n$, the goal is to predict the time instance label $\hat{y}_n(t)$ and the signal label $Y_n$.

1) Time instance prediction:

$$\hat{y}_n^{test}(t) = \arg \max_{0 \leq l \leq C} P(y_n(t) = c|x_n^{test}; w, b).$$

2) Signal label prediction: For an unlabeled test signal $x_n^{test}$, the predicted signal label set using the union rule is

$$\hat{Y}_n = \cup_{t=1}^{T_n} \{\hat{y}_n^{test}(t) | \hat{y}_n^{test}(t) \neq 0\}.$$ Alternatively, the signal label can be predicted by maximizing a posterior probability (MAP) rule:

$$\hat{Y}_n = \arg \max_{\Lambda \in \{0,1\}^C} P(Y_n = \Lambda, I_n = 1|x_n^{test}; \bar{N}_n, w, b) = \sum_{l=0}^{\bar{N}_n} \alpha_{T_n - 1 + \Delta}(\Lambda, l).$$

V. EXPERIMENTS

In this section, we first present a runtime comparison between chain and tree model reformulations of the E-step inference. We continue by evaluating the performance of the proposed approach using synthetic datasets and real world datasets.

A. Run time analysis

The computational complexity is due to three main calculations namely the prior calculation, the posterior calculation and the gradient calculation in the M-step. Since the posterior calculation dominates the computational complexity and since we have focused on developing an efficient computation for this step, the following results are on the runtime analysis of the posterior calculation during the E-step based on the chain and the tree reformulations of our model. We used a randomly generated prior probability as an input to the posterior calculation. We illustrate the relationships between the E-step posterior calculation time and the number of classes per signal $|Y_n|$, the number of time instances per signal $T_n$, the sparsity regularization per signal $\bar{N}_n$, we vary $|Y_n| \in \{1, 2, 3, 4, 5\}$, $T_n \in \{5, 10, 20, 50, 100, \ldots, 10000\}$ and $\bar{N}_n/T_n \in \{0.1, 0.2, \ldots, 1.0\}$.
E-step calculation as a function of $T$ with respect to $T$ based on the tree reformulation grows in a nearly-linear rate. A constant runtime with respect to the sparsity factor $\bar{N}/n$ is small or when $|Y_n|$ is large. Even though Figure 7(b) shows the posterior calculation time per signal $|$ for four scenarios.

Figure 7(a) shows the posterior calculation time per signal based on the tree reformulation grows in a nearly-linear rate with respect to $T_n$ when setting the sparsity level to 0.2$T_n$. In addition, it shows the chain based inference time grows quadratically in $T_n$ when $T_n > 100$. However, the chain reformulation is more efficient than the tree approach when $T_n$ is small or when $|Y_n|$ is large. Even though Figure 7(b) shows the posterior calculation time is exponential with respect to $|Y_n|$, the number of classes per signal is usually a small number in practice (see [48]). Figure 7(c) exhibits a near-constant runtime with respect to the sparsity factor $\bar{N}/n$ for the tree inference. Runtime values for both models are shown in Table 1.

### B. Synthetic datasets and settings

In designing the synthetic datasets, our goal is to test the performance of the proposed algorithms on different types of data both in terms of the dimension of the data and whether a generative or discriminative approach is taken for the data generation mechanism.

1) Data generation: Below we describe the two synthetic datasets.

**Gabor basis dataset:** This dataset is constructed with nine different Gabor filters as 1-D signal templates as shown in Figure 8(a):

$$s_{(a,f)}(t) = \cos(2\pi ft) e^{-\frac{t^2}{2a^2}},$$

for $t = -20, -19, \ldots, 20$

by setting $a = 1, 2, 3$ and $f = 0.1, 0.2, 0.3$. Each of the nine templates is used to generate signal of a particular class.

The data is generated as follows. 1) First, we generate the label sets using a fixed proportion such that 50% contains only a single label, 20% contain two, 20% contain three, and 10% contain no label and are pure noise. The labels in the label set are generated by sampling uniformly with replacement from the nine classes until the target size is reached. 2) Given a non-empty label set $Y_n$, to generate its signal $x_n$, we first decide $m_n$, the number of active time instances in the signal that contain true templates, by randomly choosing between $|Y_n|$ and $|Y_n| + 1$ for $|Y_n| \leq 2$ and sampling uniformly from $|Y_n|$ to 10 for $|Y_n| > 2$. 3) For each active time instance $k \in \{1, \ldots, m_n\}$, its exact location $t_k$ is sampled uniformly without replacement from 1 to $T_n = 200$, its class label $c_k$ is uniformly sampled from $Y_n$, and its scaling factor $A_k$ is sampled from $\U[1, 2]$. 4) We then generate $y_n^c(t) = \sum_{k=1}^{m_n} A_k x_n^c(t)$ for each class $c$ and generate the signal $\hat{x}_n = \sum_{c=1}^{9} y_n^c$.

For $s^c$, where $s^c(t) = s_{(1,0,1)}(t), s^2(t) = s_{(1,0,2)}(t), \ldots, s^9(t) = s_{(3,0,3)}(t)$. 5) Lastly, we generate the final $x_n$ by adding to $\hat{x}_n$ the white Gaussian noise, whose variance is set to $\sigma^2 = E/(T_n \text{SNR}) = E/(T_n 10^{-\text{SNR}_{dB}/10})$. Here $E$ is the average signal energy of all $\hat{x}_n$s and we use $\text{SNR}_{dB}$ to control the signal to noise ratio for our final data.

**Binary patterns dataset:** For this dataset, we work with 2-D signals and generate the data following the discriminative assumption. First we randomly generated 200 binary sequences of size $3 \times 200$ as synthetic 2-D signals ($N = 200$, $F = 3$, $|Y_n|$ = 2, $\bar{N}/n = 0.5$, $|Y_n| = 2$, $\bar{N}/n = 0.5$, $|Y_n| = 2$). Table 1: Runtime values for the chain-based and the tree-based E-step calculation as a function of $T_n$ for four scenarios.

| $T_n$ | 5  | 50  | 500  | 5000  | 10000  |
|-------|----|-----|------|-------|--------|
| $N_n/T_n$ = 0.2, $|Y_n| = 2$ | chain | 0.024ms | 0.08ms | 4.27ms | 0.55s | 2.32s |
| tree | 0.074ms | 0.84ms | 11.80ms | 0.21s | 0.39s |
| $N_n/T_n$ = 0.5, $|Y_n| = 2$ | chain | 0.028ms | 0.13ms | 10.20ms | 1.19s | 5.52s |
| tree | 0.071ms | 0.95ms | 13.97ms | 0.23s | 0.46s |
| $N_n/T_n$ = 0.2, $|Y_n| = 5$ | chain | 0.046ms | 0.63ms | 0.06s | 10.32s | 45.34s |
| tree | 0.95ms | 28.11ms | 0.52s | 9.29s | 22.25s |
| $N_n/T_n$ = 0.5, $|Y_n| = 5$ | chain | 0.074ms | 1.21ms | 0.16s | 24.39s | 198.78s |
| tree | 0.90ms | 37.91ms | 0.57s | 11.15s | 27.00s |

Figure 8: Nine Gabor basis used in the experiment.
For each generated 2-D signal, we then determined the label for each of its time instance \( t \) by matching the 3 \( \times \) 3 sub-window starting at \( t \) to three pre-defined class-specific template\(^2\) shown in Figure 10(a). The label was set to 1, 2 or 3 if the sub-window matched the template of class 1, 2 or 3 respectively, and 0 otherwise. After all time instance labels were created, the signal label was set to the union of its corresponding instance labels.

2) Experimental setting: To demonstrate the performance of the proposed approach, we used 10 random splits of 100 signals and trained on each split of 80% data and tested on the rest 20% data. For each random split, we denote it as one Monte-Carlo (MC) run. We evaluated the performance on the test data with all 10 MC runs to find kernel size \( T_w \), regularization term \( \lambda_r \), and the cardinality constraints \( \bar{N}_n \). For the Gabor basis dataset, we first tuned the model parameters by evaluating the average signal label prediction accuracy with \( \lambda_r \in \{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{0}, 10^{2}\} \) and \( T_w \in \{5, 10, 20, 40, 60, 80\} \). The iteration number was set to 10,000. Using cross-validation for prediction accuracy, we found the optimal \( \lambda_r \) and \( T_w \) and used those to present the prediction performance as a function of the cardinality constraint parameter \( N \in \{5, 10, 20, 50, 100, 200\} \) (setting \( \bar{N}_1, \ldots, \bar{N}_N = N \) and the SNR\(_{dB} \in \{-10, -5, 0, 5, 10, 15, 20, 25\} \) (see Figure 9(a)).

For the binary patterns dataset, we tune the kernel size \( T_w \in \{1, 3, 5, 10\} \), regularization term \( \lambda_r \in \{10^{-6}, 10^{-4}, 10^{-2}, 10^{0}\} \) and the cardinality constraint \( \bar{N}_n \in \{3, 5, 10, 50, 100\} \).

Benchmark competing algorithm - A generative dictionary learning followed by logistic regression (GDL-LR) approach: To the best of our knowledge, we are unaware of other weak-supervision methods for convolutive dictionary learning. In order to provide a benchmark, we considered a two-step approach: a generative convolutive dictionary learning method followed by a classifier\(^3\). For the implementation of the generative dictionary learning method, we chose [49] (used previously on the HJA dataset) and constructed a generative dictionary \( D = \{d_1, d_2, \ldots, d_K\} \). We used a matched filter approach to compute a test statistic for each of the \( K \) dictionary works as \( \max_t \bar{d}_k \ast \mathbf{x}_{\text{train}}^\top |_t \), where \( \bar{d}_k \) is a time reversed version of \( d_k \) (\( \bar{d}_k(t) = d_k(-t) \)). We combined the \( K \) test statistics into one feature vector and trained \( C \) logistic regression classifiers based on the feature vectors and their corresponding binary labels indicating the presence and absence of a class \( c \in \{1, 2, \ldots, C\} \). We use the resulting \( C \) classifiers in our performance evaluation for instance level classification and for signal classification.

Using the 10 MC runs, we evaluated the proposed GDL-LR approach by trained on a fixed number of 5000 outer iterations as in [49]. We vary the dictionary window size \( T_d \in \{5, 10, 20, 40, 60, 80\} \), sparsity regularization \( \lambda_s \in \{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{0}, 10^{2}\} \) and the number of dictionary words \( K \in \{3, 5, 7, 9, 15, 18\} \) for the Gabor basis dataset. For the binary patterns dataset, we vary \( T_d \in \{1, 3, 5, 10\} \), sparsity regularization \( \lambda_s \in \{10^{-4}, 10^{-2}, 1, 5\} \) and \( K \in \{3, 7, 9, 15\} \).

Evaluation metric: In computing the instance level detection area-under-the-curve (AUC), we calculate an AUC for each class \( c \) and obtain \( AUC = 1/C \sum_c AUC_c \). For each class \( c \), we obtain the ground truth based on the presence and absence of a given class \( c \) at each time stamp \( t = 0, \ldots, T_n - 1 \) and use \( P_t = P(y(t) = c|\mathbf{x}_{\text{test}}, \mathbf{w}, \mathbf{b}, \mathbf{K}) \) as a test score [50].

The signal level detection AUC is obtained based on \( AUC_c \) for all \( c = 1, 2, \ldots, C \). For each class \( c \), the AUC\(_c \) is obtained based on the signal level ground truth and the corresponding test score defined as \( 1 - \prod_{t=-\Delta}^{T_n-1} (1 - P_t) \).

C. Results on synthetic datasets

1) Gabor basis dataset: Based on the highest prediction accuracy, the hyper-parameters of our WSCADL approach are set to be \( \lambda_r = 10^{-4} \) and \( T_w = 40 \) via the aforementioned cross-validation. The hyper-parameters on the GDL-LR approach are set to be \( \lambda_r = 1, K = 9 \) and \( T_d = 40 \). The optimal window size is learned to be 40, which is close to the ground truth Gabor basis length. We believe the kernel size should at least cover the length of the signal patterns to obtain a good performance. If the kernel size is set to be too large, over-fitting may occur. For the proposed WSCADL approach and the competing GDL-LR framework, we observe that the prediction performance increases when SNR\(_{dB} \) increases in Figure 9(a). While the two methods perform similarly at low SNR\(_{dB} \) values, but for medium and high values the proposed WSCADL approach outperforms the competing GDL-LR approach.

Figure 9: Gabor basis dataset performance metrics for the WSCADL approach (solid \( o \)) and the GDL-LR approach (dashed \( o \)) as a function of SNR\(_{dB} \) in (a), and for the WSCADL approach as a function of \( \bar{N}_n \) in (b).

To show the importance of the cardinality constraints, we present the signal label accuracy of the proposed approach, average instance-level and signal-level detection AUC as a function of the cardinality parameter \( \bar{N}_n \) in Figure 9(b). As Figure 9(b) shows, when \( \bar{N}_n < 20 \), the signal label accuracy and signal-level AUC drops significantly. We suspect that this
setting forces some of the non-zero instance labels to be predicted as zero both in the training and test. When \( \bar{N}_n > 20 \), the performance drops gracefully. In this setting, we allow some of the zero instance labels to be predicted as non-zero. From Figure 9(b), the optimal \( \bar{N}_n \) is 20 in terms of signal label accuracy. We also observe the average instance-level AUC reaches the peak when \( \bar{N}_n = 10 \), which is the ground truth maximum cardinality in the data. However, the average signal-level AUC and signal label accuracy reaches peak when \( \bar{N}_n = 20 \), which is slightly higher than the ground truth.

To evaluate the performance in terms of AUC, we fixed SNR\(_{dB}\) to be 20. We present the detection AUC performance for both methods in Table III. Comparing the instance level and the signal detection performance from class 1 to 7, we observe that our proposed WSCADL approach outperforms the GDL-LR approach. For class 8 and 9, the GDL-LR approach detection AUCs is comparable to our WSCADL and sometimes, the AUCs for the GDL-LR approach is slightly higher than our approach. The variance of the detection AUCs for the GDL-LR approach is mostly higher than our WSCADL approach. We suspect that since the GDL-LR approach performs an unsupervised dictionary learning followed by a classifier training in a separate fashion, the resulting words may have large variability. We believe that this can be fixed by combining the two steps into one. However, due to the weak-supervision setting, the combined approach is a non-trivial extension, which to the best of our knowledge is unavailable.

### Table II: Gabor basis dataset: Detection AUCs (%) for the WSCADL and the GDL-LR approaches with optimal tuning parameters

| Class | WSCADL-ins. | WSCADL-sig. | GDL-LR-ins. | GDL-LR-sig. |
|-------|-------------|-------------|-------------|-------------|
| c=1   | 99.09 ± 1.94 | 99.89 ± 0.36 | 92.68 ± 4.33 | 91.77 ± 5.18 |
| c=2   | 99.95 ± 0.02 | 96.74 ± 2.40 | 90.74 ± 12.78 | 81.92 ± 7.83 |
| c=3   | 99.26 ± 1.97 | 99.67 ± 0.72 | 95.45 ± 10.33 | 90.00 ± 5.84 |
| c=4   | 96.80 ± 7.34 | 97.65 ± 1.72 | 90.40 ± 9.67 | 87.85 ± 5.46 |
| c=5   | 99.75 ± 0.10 | 92.84 ± 2.15 | 97.27 ± 2.65 | 86.56 ± 7.60 |
| c=6   | 97.96 ± 2.92 | 95.63 ± 5.30 | 93.24 ± 6.77 | 89.38 ± 4.85 |
| c=7   | 87.83 ± 11.19 | 94.32 ± 9.89 | 84.26 ± 15.93 | 83.73 ± 15.15 |
| c=8   | 94.40 ± 2.08 | 94.84 ± 4.12 | 93.93 ± 4.86 | 96.54 ± 3.49 |
| c=9   | 94.96 ± 5.18 | 85.59 ± 5.29 | 96.22 ± 5.16 | 97.95 ± 1.21 |

Table II: Gabor basis dataset: Detection AUCs (%) for the WSCADL and the GDL-LR approaches with optimal tuning parameters

#### 2) Binary patterns dataset: the hyper-parameters are set to be \( \lambda_r = 10^{-2}, \bar{N}_n = 3 \) and \( T_{dd} = 5 \) via the aforementioned cross validation. The optimal kernel size \( 3 \times 5 \) is slightly higher than the ground truth window size \( 3 \times 3 \). For the GDL-LR approach, the optimal dictionary window size \( T_{dd} \) is 5, sparsity constraint \( \lambda_n \) is 1 and the number of dictionary words \( K \) is 15.

The learned WSCADL words in Figure 10(b) and the learned GDL-LR words in 10(c) show that WSCADL is able to recover the true patterns while the GDL-LR approach fails. Figure 10(d) and (e) also shows that WSCADL can localize the corresponding class patterns ideally while the GDL-LR approach is failing in this task. The resulting detection AUCs in both WSCADL and GDL-LR approaches are shown in Table III

Due to the discriminative nature of the data, our proposed WSCADL model outperforms the GDL-LR approaches significantly. Since the data was not constructed as a linear combination of dictionary words, the GDL-LR approach was not able to recover a dictionary that could reconstruct the data accurately. Under the discriminative data generation setting, the GDL-LR approach can reproduce the original data only when the sparsity constraint is relaxed. However, regardless of sparsity the GDL-LR approach seems to perform poorly on classification. We suspect that this is due to the lack of discriminative power in the GDL-LR dictionary words obtained.

### D. Real-world datasets and results

#### 1) Dataset description: Below we describe the two real-world datasets.

**AASP challenge - office live scene dataset:** This dataset consists of audio recordings of sounds taken in an office environment [51]. The training dataset consists of 20 to 22 individual events (such as door slam, phone ringing, and pen drop) recording with varying time from 0.05 to 20s for 16

#### Table III: Binary patterns dataset: Detection AUCs (%) for the WSCADL and the GDL-LR approaches with optimal tuning parameters

| Class | WSCADL-ins. | WSCADL-sig. | GDL-LR-ins. | GDL-LR-sig. |
|-------|-------------|-------------|-------------|-------------|
| c=1   | 100.00 ± 0.00 | 99.59 ± 1.08 | 57.92 ± 17.20 | 48.11 ± 5.18 |
| c=2   | 100.00 ± 0.00 | 99.34 ± 0.62 | 60.16 ± 18.93 | 56.64 ± 8.63 |
| c=3   | 100.00 ± 0.00 | 97.78 ± 4.45 | 56.30 ± 8.77 | 51.89 ± 10.97 |
various class. The test dataset contains seven roughly three minutes long office live sound recordings, where each single recording is multiple labeled. The task is to detect the presence and absence of events on the test set.

**HJA bioacoustic dataset:** The HJA dataset contains 548 labeled 10-second recordings of 13 different bird species. The audio recordings of bird song are collected at the H. J. Andrews (HJA) Experimental Forest, using unattended microphones \([52]\). Each recording may contain multiple species.

2) **Data preprocessing:** For AASP challenge office live training dataset, we compared our proposed approach with the supervised dictionary learning approaches. Since the competing supervised dictionary learning algorithms use a fixed size feature vector, we created a fixed duration training signals from the various duration training data. For a fair comparison, we used this modified short duration training data for all algorithms. The fixed short duration training data is selected to be 1 sec duration because (i) most single occurrence of a sound event lasts less than 1 second and (ii) over 80% of the recordings are around 1sec duration. Recordings longer than 1s were chunked into 1s duration signals. Recordings shorter than 1s were extended to 1s using the last sample value. Note that our proposed WSCADL algorithm does not require the aforementioned preprocessing as it can handle varying signal length. To perform a detection task on the test audio with 3 minutes long, we chunk the test recordings into 10s and apply the following procedures.

For both datasets, each audio recording was applied with (i). Spectrogram generation: FFT is applied to each windowed signal with 16ms window size of 0.9 overlap ratio and the number of FFT bins is twice of the window samples; (ii). Noise whitening: each column on the spectrogram was divided by the noise spectrum \([52]\). (iii). Spectrogram down-sampling: a Matlab built-in imresize function is applied (For office live dataset on experiment 1, training spectrogram is down-sampled from \(R_{707 \times 612}^{256 \times 200}\) to \(R_{256 \times 200}^{256 \times 200}\) and test spectrograms are from \(R_{707 \times 612}^{256 \times 200}\) to \(R_{256 \times 200}^{256 \times 200}\). On experiment 2, spectrograms are down-sampled from \(R_{707 \times 612}^{256 \times 200}\) to \(R_{256 \times 200}^{256 \times 200}\). For HJA dataset, spectrograms are from \(R_{256 \times 1249}^{256 \times 200}\) to \(R_{256 \times 200}^{256 \times 200}\).

3) **Experimental setup:** Below we present two real-world experimental setting.

**Office live experimental setting:** we considered two experiments. In the first experiment, we trained on the training dataset, which consists of the 1s duration training examples, and tested on the 10sec-long recording test set. In the second experiment, we use the 10sec-long recordings for both training and test.

**Experiment 1:** For cross-validated parameter tuning, we trained on 80% of the original labeled data and validated on the independent 20% of the data. Parameter tuning was performed for all dictionary learning approaches and the parameters that yielded the highest prediction accuracy were selected. For tuning our approach, we set the dictionary window size \(T_w \in \{10, 20, 30, 40\}\), the cardinality constraint \(N_n \in \{5, 10, 60, 100, 200\}\) along with a regularization term \(\lambda_r \in \{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 1, 10\}\). Using the learned WSCADL words for the optimal tuning parameter value, we evaluated both signal and instance level detection performance on the test set. For the other supervised dictionary learning approaches, it is not easy to perform the detection task since their approaches are non-convolutive.

**Experiment 2:** We trained on 80% of the sub-sampled test set along with the signal labels generated by the union of event ground truth labels. For choosing the optimal model parameters, we considered the same range as in experiment 1. We evaluated the detection performance on the remaining 20%.

**HJA bioacoustic experimental setting:** For cross-validated parameter tuning, we trained on 80% of the training data and evaluated the performance on the independent 20% of the data. The tuning parameters considered were: the window size \(T_w \in \{10, 20, 50, 100\}\), the cardinality constraint \(N_n \in \{10, 20, 40, 60, 100, 160, 200\}\) and the regularization term \(\lambda_r \in \{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 10^0, 10^1\}\). After we learned the analysis words for the optimal tuning parameter value, we used the dictionary to predict the signal label on the test set.

4) **Results:** Below we present the results on two real-world datasets.

| Experiment-ins/sig | minimum class | mean over class | maximum class |
|--------------------|---------------|----------------|--------------|
| 1-instance         | 40.90 ± 1.99  | 53.29 ± 1.00   | 65.66 ± 0.14 |
| 2-instance         | 36.57 ± 1.60  | 54.75 ± 3.32   | 77.65 ± 1.64 |
| 1-signal           | 45.57 ± 4.62  | 64.23 ± 0.39   | 95.12 ± 4.62 |
| 2-signal           | 45.58 ± 9.18  | 70.19 ± 3.54   | 99.88 ± 9.18 |

Table IV: Instance level and signal detection AUCs (%) for both experiments across five MC runs.

**Office live event detection:** We compared our WSCADL approach with discriminative dictionary learning approaches: sparse representation-based classification (SRC) \([23]\); label consistent K-SVD (LCKSVD1,LCKSVD2) \([53]\); dictionary learning with structured incoherence and shared features (DLSI) \([26]\). Fisher discrimination dictionary learning (FDDL) \([36]\); dictionary learning for separating the particularity and the commonality (COPAR) \([54]\); fast low-rank shared dictionary learning for object classification (LRSDL) \([55]\).

![Figure 11](image-url)  
Figure 11: Prediction accuracy as a function of the cardinality parameter \(N_n\) on the AASP dataset.

For our proposed approach, the optimal tuning parameters found are window size \(T_w = 10\), the regularization term \(\lambda_r = 10^{-4}\) and the cardinality parameter \(N_n = 10\) as shown in Figure [II] which presents the performance of our proposed
WSCADL approach on varying cardinality parameter $\bar{N}_n$ for the AASP dataset. Setting the cardinality parameter less than or larger than the optimal value reduces the accuracy. For all other discriminative dictionary learning algorithms, the parameters values are tuned with cross-validation. The SRC algorithm uses all training examples as dictionary. In LCKSVD1 and LCKSVD2, DLSI and FDDL, 10 dictionary words per each class are used so that the total number of dictionary atoms is 160. In COPAR and LRSDL algorithms, 10 dictionary words per class are used with 5 shared dictionary atoms. However, in the proposed WSCADL algorithm, we assign total of 16 dictionary words therefore only 1 dictionary word is learned to predict each class. The proposed model is limited to 16 words in total since the model uses a single word per class. Potential extensions to allow more words per class may be considered as future work.

Figure 12(a) shows that our proposed method outperforms other discriminative dictionary learning approaches except SRC. Additionally, our approach outperforms all others on predicting whether the true class is among the ranked three classes as shown in Figure 12(b). The instance and signal label detection receiver operating curves (ROCs) for the proposed method are shown in Figure 12(c) and (d), and the resulting AUCs are shown in Table IV. The average and maximum detection AUCs across 16 classes for instance and signal are slightly higher in experiment 2 than experiment 1, while the minimum AUCs are lower. The detection AUC for the best performing class in experiment 2 is close to 100%, which indicates that WSCADL is able to discover that class perfectly for each test recording. Moreover, the potential of the proposed approach is demonstrated using experiment 2, in which only weak-supervision is provided. Despite this limiting setting, the average AUCs in experiment 2 are comparable or higher than the average AUCs reported in experiment 1 in which a single label per example is provided. This illustrates the potential in the label-economic weak-supervision setting and the potential of the proposed approach under this setting.

HJA bioacoustic classification: We compared our proposed WSCADL approach with the GDL-LR approach that both are dictionary learning based approaches, and with methods that perform segmentation and multi-instance multi-label (MIML) leaning approaches: MLR [48], SIM [56], MIMLfast (short for Mfast) [57], and LSB-CMM (short for LSB) [58], MIML SVM (short for M-SVM) [59] and MIMLNN (short for M-NN) [60], [61].

We evaluated all of the approaches using multi-label evaluation metrics from [60]. The results indicate that the proposed WSCADL approach outperforms GDL-LR for all metrics considered. Additionally, the proposed approach shows a slight advantage in terms of rank loss and average precision over the other MIML algorithms. For one error and Hamming loss the proposed approach is comparable in performance to the other MIML approaches. Our approach is outperformed by some of the alternative MIML approaches in terms of coverage. The results from column MLR to M-NN in Table V are directly extracted from Table 4 in [48]. Note that the alternative MIML approaches from MLR to M-NN involve a process of converting spectrograms into a bag-of-words using segmentation and feature extraction for each segment while the proposed WSCADL approach and GDL-LR are directly applied to the raw spectrograms. We suspect that the disadvantage observed with the alternative MIML approaches (based on the bag-of-words representation) is due to error propagation from the segmentation and feature extraction steps, which are not jointly optimized for MIML classification.

VI. CONCLUSION

In this work, we developed a novel probabilistic model that aims at learning a convolutive analysis dictionary under
the weak-supervision setting. We incorporated cardinality constraints as observations to enforce sparsity of the signal label to determine the location of the patterns-of-interest from a given class. For the model parameter estimation, we developed the EM update rules and introduced novel chain and tree reformulations of the proposed graphical model to facilitate efficient probability calculations during the inference. In particular, under cardinality constraints that are expressed as a fraction of the signal length, we showed that the computational complexity for the chain reformulation is quadratic in the signal length and nearly-linear for the tree reformulation, which is verified in a numerical runtime comparison. As a sanity check, we demonstrated that the proposed discriminative approach performs comparably to a generative alternative on data that follows the generative paradigm. However, when the data follows a discriminative model, our approach outperforms the generative approach. Additionally, we showed that the proposed approach yields competitive and sometimes superior performance in terms of accuracy or AUC on real-world datasets when compared to either state-of-the-art approaches for dictionary learning or to alternative (i.e., non dictionary-based) solutions in the weak-supervision setting.

**APPENDIX A**

**DERIVATION OF AUXILIARY FUNCTION**

In the EM algorithm, the auxiliary function is given by the expectation of the complete data log-likelihood over the hidden variables and conditioned on the observed data. Therefore, the auxiliary function is formulated as:

\[ Q(\theta, \theta') = E_{H|D, \theta', \phi} [\log P(D, H; \theta, \phi)] \]

Applying the natural logarithmic operation on the complete data likelihood, we have

\[ \log P(D, H; \theta, \phi) = \log P(X) + \sum_{n=1}^{N} \sum_{I=1}^{T_n} \log I_{y_n(t)\in I}(\theta_n(t)) + \sum_{I=1}^{T_n} \log I_{y_n(t)\in \overline{I}}(\theta_n(t)) \]

Since the hidden data is only associated with each time instance label signal \( y_1, \ldots, y_N \), the expectation of \( P(I_n = \{y_n\}, \tilde{N}_n) \) and \( P(Y_n(y_n)) \) are constant. Therefore the auxiliary function is computed as:

\[ Q(\theta, \theta') = \sum_{n=1}^{N} \sum_{I=1}^{T_n} \log I_{y_n(t)\in I}(\theta_n(t)) + \sum_{n=1}^{C} \sum_{c=0}^{C} [\sum_{t=1}^{T_n} P(y_n(t) = c|D; \tilde{N}_n, \theta')] \cdot \log(\sum_{u=0}^{C} e^{w_x^c x_{nt} + b_c}) + \text{const.} \]

**APPENDIX B**

**COMPUTATIONAL COMPLEXITY ANALYSIS**

**A. E-step chain inference**

**Time complexity** is \( \mathcal{O}(|\sum_{n=1}^{N} |Y_n|2^{\tilde{N}_n} N_n T_n|) \): In the chain inference with both forward and backward message passing, each update of forward and backward message requires running over all possible values of \( y_n(t) \) and \( (\ell, l) \), therefore, the computational complexity is \( \mathcal{O}(|Y_n| + 1)|2^{Y_n}| \min(t, \tilde{N}_n) \). Since each time step is only depend on the previous time step, the overall computational complexity is

\[ T_n^c(t) = T_n^c(t-1) + \mathcal{O}(|Y_n| + 1)|2^{Y_n}| \min(t, \tilde{N}_n). \]

After solving this recursive formula, we have \( T_n^c = \mathcal{O}(|Y_n|^{2\tilde{N}_n} |N_n T_n|) \). Therefore, the overall chain inference needs a computational complexity of \( \sum_{n=1}^{N} T_n^c = \mathcal{O}(\sum_{n=1}^{N} |Y_n|^{2\tilde{N}_n} |N_n T_n|) \). The space complexity is \( \mathcal{O}(|2^{Y_n}| T_n|N_n|) \).

**B. E-step tree inference**

**Time complexity** is \( \mathcal{O}(\sum_{n=1}^{N} |Y_n| (\log_2 \tilde{N}_n)^2 T_n) \): In the tree inference on both forward and backward message passing, each update of forward and backward message requires running over all possible values of \( Y_{n(2t-1)}^{j} N_{n(2t-1)}^{j} \) and \( Y_{n(2t)}^{j} N_{n(2t)}^{j} \), therefore, the computational complexity is \( \mathcal{O}(|4^{Y_n}| (\min(\tilde{N}_n, 2^{L-j} + 1)) \).

However, the updates of the forward and backward messages on the tree for controlling cardinality parameter \( \tilde{N}_n \) are operating in a convolutive nature. When \( \tilde{N}_n \) is large, we rely on FFT and Inverse of FFT to speedup the convolution such that the convolution complexity will become \( \mathcal{O}((\min(\tilde{N}_n, 2^{L-j} + 1) \log(\min(\tilde{N}_n + 1, 2^{L-j} + 1))). \)

Since current instance on tree level \( j \) only depend on previous two parents at \( j + 1 \), the recursive formula of the overall computational complexity is

\[ T_n^{(j)}(t) = T_n^{(j+1)}(2t) + T_n^{(j+1)}(2t-1) + \mathcal{O}(|4^{Y_n}| X \log X), \]

where \( X = \min(\tilde{N}_n, 2^{L-j} + 1) \). Since \( 1 \leq t \leq 2^j, 1 \leq j \leq L \). After solving this recursive formula, we have \( T_n^u = \mathcal{O}(|4^{Y_n}| (\log_2 \tilde{N}_n)^2 T_n) \). The overall computational complexity of tree approach is \( \sum_{n=1}^{N} T_n^{(j)} = \sum_{n=1}^{N} \mathcal{O}(|Y_n| (\log_2 \tilde{N}_n)^2 T_n) \). The space complexity is \( \mathcal{O}(|2^{Y_n}| T_n \log_2 \tilde{N}_n|) \).

**REFERENCES**

[1] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, “Dictionary learning algorithms for sparse representation,” Neural computation, vol. 15, no. 2, pp. 349–396, 2003.

[2] M. Aharon, M. Elad, and A. Bruckstein, “K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation,” IEEE Transactions on signal processing, vol. 54, no. 11, pp. 4311–4322, 2006.

[3] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, “Online learning for matrix factorization and sparse coding,” The Journal of Machine Learning Research, vol. 11, pp. 19–60, 2010.

[4] H. Lee, A. Battle, R. Raina, and A. Y. Ng, “Efficient sparse coding,” in Advances in neural information processing systems, 2006, pp. 801–808.

[5] R. Vipperla, S. Bozonnnet, D. Wang, and N. Evans, “Robust speech recognition in multi-source noise environments using convolutive non-negative matrix factorization,” Proc. CHiME, pp. 74–79, 2011.
Zeyu You received her B.S. in Electronics and Information Technology from Huazhong University of Science and Tech (2008) and her M.S. in Electrical and Computer Engineering from Oregon State University (2014). She is currently a Ph.D. Candidate in Electrical and Computer Engineering at Oregon State University, Corvallis, OR. Her current research interests include motif discovering, dictionary learning and multi-instance multi-label learning, and applications to bioacoustics.

Raviv Raich (S’98 A’04 A’17) received the B.Sc. and M.Sc. degrees from Tel Aviv University, Tel-Aviv, Israel, in 1994 and 1998, respectively, and the Ph.D. degree from the Georgia Institute of Technology, Atlanta, in 2004, all in electrical engineering. Between 1999 and 2000, he was a Researcher with the Communications Team, Industrial Research, Ltd., Wellington, New Zealand. From 2004 to 2007, he was a Postdoctoral Fellow with the University of Michigan, Ann Arbor. Dr. Raich has been with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, as an Assistant Professor (2007-2013) and is currently an Associate Professor (2013-present). His research interests are in statistical signal processing and machine learning. Dr. Raich served as an Associate Editor for the IEEE Transactions On Signal Processing in 2011-2014. He currently serves as the chair of the Machine Learning for Signal Processing (MLSP) Technical Committee of the IEEE Signal Processing Society.

Xiaoli Fern received her Ph.D. degree in Computer Engineering from Purdue University, West Lafayette, IN, in 2005 and her M.S. degree from Shanghai Jiao Tong University (SJTU), Shanghai China in 2000. Since 2005, she has been with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR where she is currently an associate professor. She received an NSF Career Award in 2011. Dr. Fern is currently serving as action editor for the Machine Learning Journal and regularly serves on the program committee for top-tier international conferences on machine learning and data mining including NIPS, ICML, ECML, AAAI, KDD, ICDM, SIAM SDM. Her general research interest is in the areas of machine learning and data mining.

Jinsub Kim received his Ph.D. in electrical and computer engineering from Cornell University. From September 2013 to June 2014, Jinsub Kim was a postdoctoral associate at the School of ECE at Cornell University. He joined the School of EECS in Oregon State University in August 2014. His research interest spans statistical signal processing, statistical learning, optimization methods, and their applications to security and smart energy systems.

Z. Zhou and M.-L. Zhang, “Multi-instance multi-label learning,” in Advances in neural information processing systems, 2012, pp. 557–565.

Z. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, “Multi-instance multi-label learning,” Artificial Intelligence, vol. 176, no. 1, pp. 2291–2320, 2012.

M.-L. Zhang and Z.-H. Zhou, “Multi-label learning by instance differentiation,” in AAAI, vol. 7, 2007, pp. 669–674.
Supplementary Material of Weakly-supervised Dictionary Learning

Zeyu You, Student member, IEEE, Raviv Raich, Senior Member, IEEE,
Xiaoli Z. Fern, Member, IEEE, and Jinsub Kim, Member, IEEE,

APPENDIX C
DERIVATION OF COMPLETE DATA LIKELIHOOD

Given the observed data and the hidden data, we perform the complete data likelihood as:

\[ P(D, \mathcal{H}; \theta, \phi) \]

\[ = P(\mathcal{X}, \mathcal{Y}, I_1 = 1, \ldots, I_N = 1, y_1, \ldots, y_N; w, b, \tilde{N}_1, \ldots, \tilde{N}_N). \]

Using the probability rule of \( P(A, B) = P(A|B)P(B) \) and the independence assumption of each observed data point \((x_n, Y_n, I_n = 1)\), the complete data likelihood can be further computed as:

\[ P(D, \mathcal{H}; \theta, \phi) = P(\mathcal{X}) \prod_{n=1}^{N} P(Y_n, y_n, I_n = 1|x_n; w, b, \tilde{N}_n). \]

Apply the probabilistic graphic structure in Fig. 4, we have

\[ P(D, \mathcal{H}; \theta, \phi) = P(\mathcal{X}) \prod_{n=1}^{N} P(I_n = 1|y_n; \tilde{N}_n)P(Y_n|y_n) \cdot P(y_n|x_n; w, b). \]

Plug in the model formulation in (2) and (3) and due to conditional independence assumption of the each time instance label, we arrive the final formulation of the complete data likelihood as:

\[ P(D, \mathcal{H}; \theta, \phi) = P(\mathcal{X}) \prod_{n=1}^{N} \mathbb{E}_{y_n} \left[ \mathbb{I}_{\sum_{l=0}^{T_n-1-\Delta} y_n(t) = \sum_{l=\Delta}^{T_n-1+\Delta} y_n(t)} \right] \]

\[ \mathbb{I}_{(\sum_{l=\Delta}^{T_n-1+\Delta} y_n(t)) \leq \tilde{N}_n} \prod_{t=-\Delta}^{T_n-1+\Delta} P(y_n(t)|x_n; w, b). \]

APPENDIX D
DERIVATION OF FORWARD MESSAGE PASSING ON CHAIN

The derivation of the chain forward message passing is based on the definition of the forward message on the chain model \( \alpha_t(\mathbb{L}, l) = P(Y^n = l, N^n = l|x_n; \theta^i) \) and the marginal probability

\[ P(Y^n = l, N^n = l|x_n; \theta^i) = \sum_{Y^{-1} = 1} \sum_{N^{-1} = 1} \sum_{y_0(t)} P(Y^n = l, N^n = l, Y^{-1}, N^{-1}, y_0(t)|x_n; \theta^i). \]

The forward message passing update rule can be formulated by marginalizing the previous state variables \((Y^{-1}, N^{-1})\) and the current time instance \(y_0(t)\). Rely on the v-structure on the update step of the chain structure in Fig. 4(a) and the chain rule of the joint probability \( P(A, B, C) = P(A|B, C)P(B|C) \) such that

\[ P(Y^n = l, N^n = l, Y^{-1}, N^{-1}, y_0(t)|x_n; \theta^i) \]

\[ = P(Y^n = l, N^n = l|Y^{-1}, N^{-1}, y_0(t)) \cdot P(y_0(t)|x_n; \theta^i). \]

We have

\[ \alpha_t(\mathbb{L}, l) = \sum_{a, c} \sum_{a=0}^{t-1} \sum_{\mathbb{L} = \mathbb{A} \cup \{c\}} \mathbb{I}(l = a + \mathbb{I}(c \neq 0)) \cdot P(y_0(t)|x_n; \theta^i) \cdot \alpha_{t-1}(\mathbb{L}, a). \]

Due to the constraints that \( L = \mathbb{A} \cup \{c\} \) and \( l = a + \mathbb{I}(c \neq 0) \), for a fixed value of \( L \) and \( a \), \( a \) can only have one value for a particular class \( c \). Thus the update rule of the forward message can be further simplified as:

\[ \alpha_t(\mathbb{L}, l) = P(y_0(t)|x_n; \theta^i)\alpha_{t-1}(\mathbb{L}, l) \]

\[ + \sum_{c=1}^{C} P(y_0(t)|x_n; \theta^i)\mathbb{I}(l \neq 0) \cdot [\alpha_{t-1}(\mathbb{L}, l - 1) + \mathbb{I}(c \in \mathbb{L})\alpha_{t-1}(\mathbb{L}_{\mathbb{A}, l})]. \]

APPENDIX E
DERIVATION OF BACKWARD MESSAGE PASSING ON CHAIN

The derivation of the chain backward message passing is based on the definition of the backward message on the chain model \( \beta_{t-1}(\mathbb{L}, l) = P(Y_n, I_n = 1|Y^{-1} = l, \mathbb{L}, N^{-1} = l, x_n; \theta^i, \tilde{N}_n) \) and the marginal probability

\[ P(Y_n, I_n = 1|Y^{-1} = l, N^{-1} = l, x_n; \theta^i, \tilde{N}_n) = \sum_{Y^{+1} = 1} \sum_{N^{+1} = 1} \sum_{y_n(t)} P(Y_n, I_n = 1, Y^{+1}, N^{+1}, y_n(t)|x_n; \theta^i). \]

\[ P(Y_n, I_n = 1, Y^{+1}, N^{+1}, y_n(t)|x_n; \theta^i, \tilde{N}_n) \]
Rely on the v-structure on the update step of the chain structure in Fig. 4(b) and the chain rule of the conditional probability \( P(\mathcal{A}, \mathcal{B}\mid \mathcal{C}) = P(\mathcal{A}, \mathcal{B}\mid \mathcal{C}) P(\mathcal{B}\mid \mathcal{C}) \), we have
\[
P(Y_n, I_n = 1|Y_{n_1}^n, Y_{n_1}^{n_1}, y_n(t)) P(Y_{n_1}^{n_1}, N_{n_1}^{n_1}, x_n; \theta^i, \bar{N}_n)
= P(Y_n, I_n = 1|Y_{n_1}^n, Y_{n_1}^{n_1}, y_n(t), x_n; \theta^i, \bar{N}_n).
\]
\[
P(Y_n, y_n(t)|y_n(t)) P(y_n(t)|x_n; \mathbf{w}^i, \mathbf{b}^i).
\]
Given the current time joint state node \((Y_t^t, N_t^t)\), the observed node \((Y_n, I_n)\) is independent of the previous joint state node \((Y_{n_1}^{n_1}, N_{n_1}^{n_1})\) and the current time instance \(y_n(t)\), so \( P(Y_n, I_n = 1|Y_{n_1}^n, Y_{n_1}^{n_1}, y_n(t), x_n; \theta^i, \bar{N}_n) = P(Y_n, I_n = 1|Y_{n_1}^n, N_{n_1}^{n_1}, x_n; \theta^i, \bar{N}_n) \). Combining the above two equations, applying the deterministic rule \( P(Y_n^t = \mathbb{L}, N_t^t = l | Y_{n_1}^{n_1} = \mathbb{A}, N_{n_1}^{n_1} = a, y_n(t) = c) = \mathbb{I}(l = a + \|c\| = 0) \) and applying the definition of the forward message \( P(Y_{n_1}^{n_1}, N_{n_1}^{n_1}|x_n; \theta^i) = \alpha_{t-1}(Y_{n_1}^{n_1}, N_{n_1}^{n_1}) \), the joint probability is performed as:
\[
P(y_n(t) = c, Y_n, I_n = 1|x_n; \theta^i, \bar{N}_n)
= \sum_{\mathbb{L} \subseteq 2^n, a = 0}^{t-1} \sum_{l = 0}^{t-1} \mathbb{I}(l = a \cup \{c\}) \mathbb{I}(a = l + \|c\| = 0) \alpha_{t-1}(\mathbb{L}, l) \beta_l(a, c)
\]
\[
p(y_n(t) = c|x_n; \mathbf{w}^i, \mathbf{b}^i),
\]
where \( \bar{N}_n = \min(\bar{N}_n - \|c\|, 0), \).

APPENDIX G
DERIVATION OF FORWARD MESSAGE PASSING ON TREE

The forward message passing update on tree can be first applied with the definition of the forward message on tree \( \alpha_{t-1}^l(\mathbb{L}, l) = P(Y_{n_1}^{n_1} = \mathbb{L}, N_{n_1}^{n_1} = l | x_n; \theta^i) \) and the marginal probability
\[
P(Y_{n_1}^{n_1}, N_{n_1}^{n_1}|x_n; \theta^i) = \sum_{Y_{n_1}^{n_1}} \sum_{N_{n_1}^{n_1}} Y_{n_1}^{n_1} Y_{n_1}^{n_1} N_{n_1}^{n_1} N_{n_1}^{n_1} P(Y_{n_1}^{n_1}, N_{n_1}^{n_1} | x_n; \theta^i)
\]
According to the v-structure of the update step in Fig. 6(a), the joint probability can be decomposed as:
\[
P(Y_{n_1}^{n_1}, N_{n_1}^{n_1}|x_n; \theta^i) = P(Y_{n_1}^{n_1}, N_{n_1}^{n_1} | x_n; \theta^i) P(Y_{n_1}^{n_1} | x_n; \theta^i)
\]
Due to the deterministic rule between \((Y_{n_1}^{n_1}, N_{n_1}^{n_1})\) and \((Y_{n_1}^{n_1}, N_{n_1}^{n_1})\), the \((Y_{n_1}^{n_1}, N_{n_1}^{n_1})\) as proposed in (15) and (16), \( P(Y_{n_1}^{n_1}, N_{n_1}^{n_1}|x_n; \theta^i) = \mathbb{I}(Y_{n_1}^{n_1} = \mathbb{L} \cup \{c\}) \mathbb{I}(\|c\| = 0) \).
Combining the above equations, we obtain the update rule of the forward message passing on tree as:
\[
\alpha_{t-1}^l(\mathbb{L}, l) = \sum_{A \subseteq 2^n, a = 0}^{t-1} \sum_{l = 0}^{t-1} \mathbb{I}(\mathbb{L} = A \cup \{c\}) \mathbb{I}(a = l + \|c\| = 0)
\]
\[
\alpha_{t-1}^l(a, c) \alpha_{t-1}^l(\mathbb{L} \setminus A, l - a),
\]
where \( \bar{N}_n^{*} = \min(\bar{N}_n, 2L-j) + 1 \).
APPENDIX H
DERIVATION OF BACKWARD MESSAGE PASSING ON TREE

Given the definition of the backward message on tree $\beta_{2t-1}(A, a) = P(Y_n, I_n = 1 | Y^j_{n(2t-1)} = A, x_n; \theta^j, \tilde{N}_n)$ and $\beta_{2t}^j(E, e) = P(Y_n, I_n = 1 | Y^j_{n(2t)} = E, N^j_{n(2t)} = e, x_n; \theta^j, \tilde{N}_n)$, the backward message passing update on tree can be derived based on marginal probabilities:

\[
P(Y_n, I_n = 1 | Y^j_{n(2t-1)}, N^j_{n(2t-1)}, x_n; \theta^j, \tilde{N}_n) = \sum_{Y^j_{n(2t)}} \sum_{N^j_{n(2t)}} \sum_{Y^{j-1}_{n(t)}} \sum_{N^{j-1}_{nt}} P(Y_n, I_n = 1, Y^j_{n(2t)}, N^j_{n(2t)}, Y^{j-1}_{nt}, N^{j-1}_{nt}, x_n; \theta^j, \tilde{N}_n)
\]

and

\[
P(Y_n, I_n = 1 | Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \theta^j, \tilde{N}_n) = \sum_{Y^j_{n(2t-1)}} \sum_{N^j_{n(2t-1)}} \sum_{Y^{j-1}_{nt}} \sum_{N^{j-1}_{nt}} P(Y_n, I_n = 1, Y^j_{n(2t-1)}, N^j_{n(2t-1)}, Y^{j-1}_{nt}, N^{j-1}_{nt}, x_n; \theta^j, \tilde{N}_n).
\]

According to the v-structure of the update step in Fig. 6(b), the joint probabilities can be decomposed as:

\[
P(Y_n, I_n = 1, Y^j_{n(2t)}, N^j_{n(2t)}, Y^{j-1}_{nt}, N^{j-1}_{nt})
\]

\[
| Y^j_{n(2t-1)}, N^j_{n(2t-1)}, x_n; \theta^j, \tilde{N}_n)
\]

\[
= P(Y^j_{nt}, N^j_{nt}) P(Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \theta^j, \tilde{N}_n)
\]

and

\[
P(Y^j_{n(2t-1)}, N^j_{n(2t-1)}, Y^{j-1}_{nt}, N^{j-1}_{nt})
\]

\[
| Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \theta^j, \tilde{N}_n)
\]

Due to the deterministic rule that

\[
P(Y^j_{nt-1}, N^j_{nt-1}) P(Y^j_{n(2t-1)}, N^j_{n(2t-1)}, Y^{j-1}_{nt}, N^{j-1}_{nt})
\]

\[
= \|Y^j_{nt} = Y^j_{n(2t-1)} \cup Y^j_{n(2t)}\| (N^j_{nt-1} = N^j_{n(2t-1)} + N^j_{n(2t)})
\]

we derive the update of the backward message passing update rule by combining the above equations as:

\[
\beta_{2t-1}^j(A, a) = \sum_{L \in 2^Y_n} \sum_{l=0}^{L} \sum_{E \in 2^Y_n} \sum_{e=0}^{E} \|L = A \cup E\| (l = a + e)
\]

\[
\beta_{2t}^j(l, 0) \alpha_{2t}^j(E, e)
\]

\[
= \sum_{E \in 2^Y_n} \sum_{e=0}^{E} \beta^{-1}_t(A \cup E, a + e) \alpha_{2t}^j(E, e).
\]

APPENDIX I
AASP SPECTROGRAMS IN THE TRAINING DATA

Figure 13: Overview of Office live scene sound transformed spectrograms in AASP challenge