AN ABSTRACT OF THE DISSERTATION OF

Zeyu You for the degree of Doctor of Philosophy in Electrical and Computer Engineering presented on March 12, 2019.

Title: Weak-supervision Time-series Analysis

Abstract approved: _____

Raviv Raich

Efficient time-series analysis can impact multiple application domains such as motif discovery in gene analysis or music data, extracting spectro-temporal patterns in acoustic scene analysis, or annotating and classifying electrical bio-signals (such as ECG, EEG, and EMG) for medical applications. Time-series analysis involves a variety of tasks. To predict future values of a time-series, many approaches focus on capturing the time dependence between samples, e.g., hidden Markov model or recurrent neural network. To learn a compact representation, transformation based approaches such as discrete Fourier transform (DFT), singular value decomposition (SVD) or convolutive neural network (CNN) are considered. To classify the time-series, one common approach is a distance-based K-nearest neighbors (KNN) with dynamic time warping (DTW) or edit distance. Other supervised approaches either use hand-crafted features or deep learning representation. In time-series data, as in electrical bio-signals, acoustic scene, and music theme analysis, occurrence of patterns can be modeled as stationary or time-invariant. Discovering such patterns is the key to the analysis of time-series and can be further used for reconstruction or classification. Rich time-series data may contain

multiple patterns associated with multiple labels (e.g., in acoustic scene analysis of an urban environment, vehicle sounds, bird sounds, and human speech may be observed in the time interval). To resolve the labels of individual instances, often an expensive fine-grain labeling process is required. To reduce the labor-intensive annotation efforts associated with labeling a signal at a time-instance level, coarse interval-level labeling is often considered. In this context, we propose a framework that can efficiently model and analyze large-scale multi-channel time-series data and provide fine-grain label predictions from coarse interval-labeled data.

Since our focus is on time-series data with time-invariant events, we consider a convolutive modeling. To extract time-invariant recurring patterns in time-series, we first propose a convolutive generative framework and use the resulting features for classification. To learn a time-instance label model in a weak-supervision setting efficiently, we propose novel dynamic programming approaches (using both a chain and a tree structure). Moreover, we extend the proposed weakly-supervised dictionary learning model for adapting both multiple clusters and multiple-scales. As future work, we present an application of the proposed approach to deep learning.

Weak-supervision Time-series Analysis

by

Zeyu You

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented March 12, 2019
Commencement June 2019

Doctor of Philosophy dissertation of Zeyu You presented on March 12, 2019.

APPROVED:

Major Professor, representing Electrical and Computer Engineering

---

Head of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Zeyu You, Author

ACKNOWLEDGEMENTS

I would like to acknowledge my major advisor Prof. Raviv Raich, without his patience and kindness guidance, I would not make so much progress. His motivation, enthusiasm and immerse knowledge helped me to work through this thesis.

Besides my advisor, I would like to thank my family and my friends, especially my mum, dad and my husband for always supporting me throughout my life and help me go through all these difficult periods.

Moreover, I would like to thank the rest of my committees, minor advisors, and GCR: Prof. Xiaoli Fern, Prof. Jinsub Kim, Prof. Xiao Fu, Prof. Sarah Emerson, Prof. Liang Huang and Prof. Thomas Schmidt for their support and advice.

I would also like to thank José Francisco Ruiz-Muñoz from Universidad Nacional de Colombia - Sede Manizales, we used to work together as co-authors in two of my published papers. At last but not least, I want to thank our group members, Anh Pham, Tam Nguyen Thi, Thi Kim Phung Lai and Vu Trung Viet, for their support and all the fun we had in the past years.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF APPENDICES

Chapter 1: Introduction

In recent years, digital signal processing and machine learning techniques have been widely applied to the analysis of time-series. In the analysis of time-series, classification is an important and challenging problem. Time series classification is key to many applications. For example, in species conservation, the presence or absence of endangered bird species in their habitat can be monitored based the detection of their vocalization in audio recordings. In bioinformatics, detecting subsequences of a long DNA sequence (e.g., regulatory motif) can help isolate portions of the DNA, which help to control the expression of genes.

Machine learning techniques are usually applied to ease the analysis of a large collection of data. The traditional classification approaches may not work because the attributes are ordered in time. The ordering of the attributes is usually considered as an important characteristic of the discriminative features. Time-series representation is fundamental to many of the key tasks in time-series analysis including predicting the next value, extracting recurring patterns, or classifying a short intervals of the time-series. In this context, the convolutive learning approaches are particularly suitable to analyze the time-series data which is believed to be generated in a time-invariant fashion. Finding such patterns is important for identifying a generative model for the data and in assisting in classification. In this work, we focus on convolutive models for representing and modeling time-series.

When the label information is given, the goal is extended to not only minimizing the reconstruction error but also minimizing the classification error. In most time-series

data, labeling the time-series example at a time-instance level is labor-intensive. Labeling efforts can be reduced by assigning a coarse label that indicates the presence and absence of a particular class in a given interval. This setting is referred to as weak-supervision learning, in which fairly long time-series containing multiple events are provided with only the presence or absence of classes within the interval thereby allowing a fairly inexpensive labeling process. To address the scalability issue associated with intensive labeling of large amounts of time-series data, we focus on learning under the weak-supervision setting. The goal in this setting is twofold: (i) recognizing the presence or absence of a class in a previously unseen interval and (ii) determining and localizing all occurrences of events from any class within the previously unseen time interval. Our task is to not only learn how to label long sequences but also to provide the capability of labeling them at a fine-granularity based on the coarse-granularity labels.

1.1 Unsupervised learning in time series

The amount of electronic time-series data, such as stock prices, ECG/EEG data or audio clip, is growing rapidly. Time-series data, which ranges from commercial, medical to scientific domains, may contain multiple occurrences of time-invariant patterns. Detecting such patterns in such large data streams or time-series is important for knowledge discovery, predicting the next value, or classifying them. These tasks contain many interesting challenges within the context of providing computer tools for exploring large data archives. Based on the level of data preparation, the methodology falls into two categories: (i) For those well-prepared data (involving data selection, denoising and segmented to a fixed-length vector), traditional unsupervised methods are directly applied; (ii) For those automatically collected streaming data (not involving data segmentation

or annotation), time-invariant approaches are considered (e.g., a moving window representation or convolutive modeling).

1.1.1 Time-series as fixed length vectors: distance-based pattern discovery

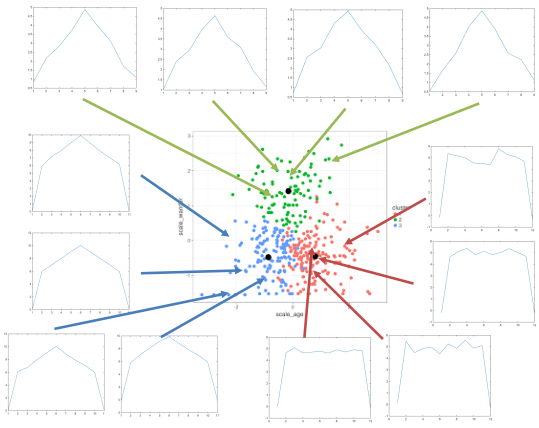


Figure 1.1: An example of clustering time-series

To discover the knowledge behind the time-series data, many traditional methods rely on the data that was pre-processed to have carefully selected fixed length time-series intervals as feature vectors. Distinct patterns within the fixed-length windows are identified using clustering methods such as k-means [117], Gaussian mixture model [47] or hidden Markov model [86]. See Figure 1.1 for an example. A common approach for identifying such patterns in the data relies on using a similarity measure between two data vectors. In many applications, the use of Euclidean distance as a dissimilarity measure may be inappropriate due to various artifacts that cannot be factored out by the Euclidean metric such as delay, multi-modality, or other distortions. Instead, [11]

proposes a dynamic time warping algorithm (DTW) to match a short sequence template in a large sequence data with a notion of "fuzziness", which provides a robust similarity measure between two temporal subsequences.

1.1.2 Time-series as fixed length vectors: transformation-based pattern discovery

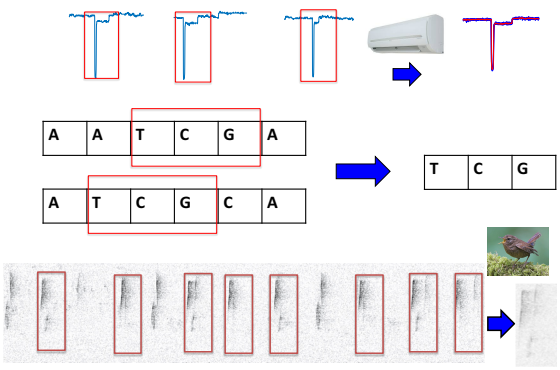


Figure 1.2: Time-series data examples with various application domain

Raw time-series data is usually high-dimensional and contains multiple temporal patterns or motifs. See Figure 1.2 for various examples. Hence, dimension reduction or data transformation is needed to improve the efficiency and accuracy of the learning methods. In the disaggregated end-use energy problem, different delays and offsets are observed in the voltage transient response associated with the activation of a home appliance. Bird syllables (recurring vocalizations) may commonly appear in large time-series audio data with some variation in time duration or frequency range. Hence, transformation based approaches are proposed such as singular value decomposition (SVD) [135], discrete Fourier transform (DFT) [97], wavelet transform [19], and piece-

wise aggregate approximation (PAA) [53] to robustly match similar time-series. As with SVD, dictionary learning methods provide a data-driven approach for identifying a compact basis for representing high-dimensional data vectors. For example, sparse coding dictionary learning can obtain an over-redundant basis and a sparse representation of data point over the basis (dictionary). In the following, we review some of the dictionary learning approaches for discovering local patterns.

1.1.2.1 Synthesis dictionary learning

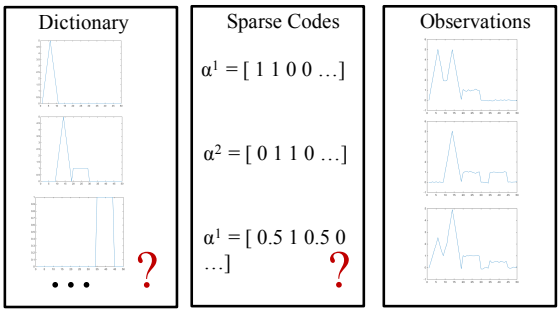


Figure 1.3: An illustration of synthesis dictionary learning

In *synthesis dictionary learning* [1, 55, 89], the goal is to simultaneously find a dictionary and corresponding coefficients to represent a set of n time-series $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^m$. A dictionary is a collection of atoms $\mathbf{D} = [\mathbf{d}_1^T, \ldots, \mathbf{d}_K^T]^T$, where $\mathbf{d}_k \in \mathbb{R}^m$ is the kth dictionary atom (or word). The ith signal can be approximated by a linear combination over the dictionary $\mathbf{D}$ by

$$\mathbf{x}_i \approx \mathbf{D}\boldsymbol{\alpha}^i = \sum_{k=1}^{K} \mathbf{d}_k \alpha_k^i, \qquad \text{for } i = 1, 2, \ldots, n,$$

where $\boldsymbol{\alpha}^i = [\alpha_1^i, \ldots, \alpha_K^i]^T$ is the coefficient vector associated with the ith signal. Synthesis dictionary learning is typically formulated as an optimization problem, where the goal is to find $\mathbf{D}$ and sparse coefficients $\{\boldsymbol{\alpha}^i\}_{i=1}^n$ that minimize the reconstruction error. Several methods have been proposed for the problem. An over-complete synthesis dictionary learning with sparse coding is introduced in [55]. Various state-of-the-art approaches have been introduced to solve the dictionary learning problem including K-SVD [1], matrix factorization [73], Lagrangian dual gradient descent and feature-sign search [59]. For analyzing audio, music or spectral image data, convolutive dictionary learning has been proposed [9, 116, 121]. Other dictionary learning based approaches have been used for searching time-varying patterns of audio signals [4, 70], e.g., to detect basic acoustic units as phonemes in speech recognition [88].

1.1.2.2 Analysis dictionary learning

In *analysis dictionary learning* [100, 102], given a signal $\mathbf{x}_i$, 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 $\mathbf{x}_i'$ ($\mathbf{x}_i \approx \mathbf{x}_i'$) such that the resulting analyzed signal

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

is sparse. For a noisy signal model, analysis dictionary learning can be formulated by minimizing a quadratic error between all $\mathbf{x}_i$'s and $\mathbf{x}_i^{'}$'s subject to the resulting analyzed

signals are sparse.

$$\begin{aligned} \underset{\mathbf{w},\mathbf{X}'}{\text{minimize}} \quad & \|\mathbf{X}' - \mathbf{X}\|_F^2 \\ \text{subject to} \quad & \|\mathbf{w}\mathbf{x}_i'\|_0 \leq p - l, \ \forall 1 \leq i \leq n, \\ & \|\mathbf{w}_j\|_2 = 1, \ \forall 1 \leq j \leq p. \end{aligned} \tag{1.1}$$

where p is the total number of analysis dictionary bases, l is the co-sparsity (number of zero elements) of $\mathbf{wx}$, and $\mathbf{w}_j$ is the jth column vector of $\mathbf{W}$.

1.1.3 Time-series as streaming data: moving window approaches

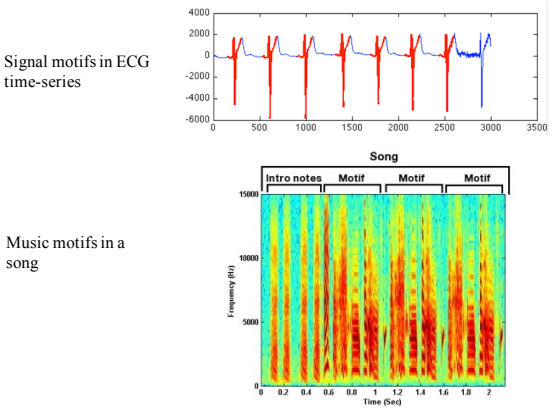


Figure 1.4: Signal motifs in multiple time-series stream data: ECG [64] and Music [30]

Due to the development of hardware and software, the amount of time-series data is growing rapidly in a wide range of fields. These measurements are generated continuously and in very high fluctuating data rates. Example domains include economic, medical, music, audio, and acoustics. Segmenting a long series into fixed-length windows that contain useful information would require domain knowledge and is time-consuming. Alternatively, automatic discovery of patterns or knowledge behind large size stream-

ing data is necessary (see Figure 1.4 for an example). Recently, a framework based on matrix profiling proposes an efficient way to extract useful and local patterns in time-series [37, 46, 143, 145, 158]. Given a collection of data objects, the matrix profile retrieves the statistics (index and distance) of each object (moving window) to its nearest neighbor [145]. The sliding window approaches, similar to convolutive modeling, provide an efficient way of discovering patterns no matter where it is in the time-series. In the following, we review several convolutive modeling approaches.

1.1.3.1 Convolutive Non-negative Matrix Factorization (CNMF)

In the analysis of audio recordings of bioacoustics, convolutive non-negative matrix factorization (CNMF) methods are applied for describing the sound-scape and analyzing the environmental impact of human activity and natural changes [12]. In CNMF, the goal is to approximate a matrix $\mathbf{V} \in \mathbb{R}^{M \times N}$ with a series of two non-negative matrices $\mathbf{W}_t \in \mathbb{R}^{M \times R}$ and $\overset{t\rightarrow}{\mathbf{H}} \in \mathbb{R}^{R \times N}$ in a convolutive manner. The CNMF is a type of structured non-negative matrix factorization (NMF) model [127], which can be applied to dictionary learning for speech or audio analysis [87, 88]. Based on a CNMF model, an observed spectrogram $\mathbf{V}$ can be written as:

$$\mathbf{V} \approx \sum_{t=0}^{T-1} \mathbf{W}_t \overset{t\rightarrow}{\mathbf{H}}, \tag{1.2}$$

such that the ik element of $\mathbf{V}$ is $v_{ik} = \sum_{t=0}^{T-1} \sum_{j=1}^{R} w_{ijt}(\overset{t\rightarrow}{h}_{jk})$.

The Kullback-Leibler (KL) divergence is used because this approach requires that the factorized matrices are both positive. The solution is achieved by repeatedly alternating

between updating $\mathbf{W}$ and $\mathbf{H}$ with sparseness constraint as:

$$\mathbf{H} = \mathbf{H} \otimes \frac{\mathbf{W}_t^T \cdot [\overset{t\rightarrow}{\frac{\mathbf{V}}{\mathbf{\Lambda}}}]}{\mathbf{W}_t^T \cdot \mathbf{1} + \breve{} \cdot \mathbf{1}}, \text{ and,} \tag{1.3}$$

$$\mathbf{W}_t = \mathbf{W}_t + \gamma_w \big[\frac{\mathbf{V}}{\mathbf{\Lambda}} \cdot \overset{t\rightarrow^T}{\mathbf{H}} - \mathbf{1} \cdot \overset{t\rightarrow^T}{\mathbf{H}}\big], \tag{1.4}$$

where $\mathbf{\Lambda} = \sum_{t=0}^{T-1} \mathbf{W}_t \overset{t\rightarrow}{\mathbf{H}}$,

Despite the utility of CNMF in analyzing time-series signals, a few challenges arise when applying the CNMF to the bioacoustic setting: (i) CNMF has a high computational requirement [120] such that it is difficult to apply to large amounts of bioacoustic signals [109]; (ii) CNMF is typically used in a single spectrogram setting, where bioacoustic signals usually contain a collection of discontinuous recordings; and (iii) CNMF often assumes that the length of the activation signal is the same as the length of the spectrogram in the time domain but it is possible that recordings register only part of a vocalization at the beginning or the end. In this case, a longer activation signal should allow for representing syllable parts in the beginning and the end of the spectrogram.

1.1.3.2 Convolutive synthesis dictionary learning

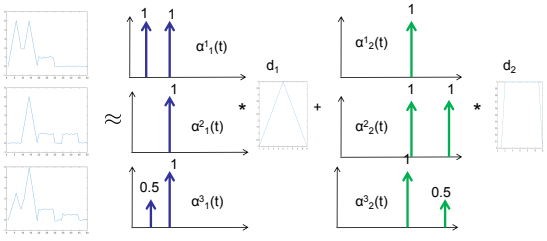


Figure 1.5: An illustration of synthesis dictionary learning

Convolutive synthesis dictionary learning is often considered [49, 160] for time invariant signals such as speech and audio. In the convolutive dictionary learning, the ith signal $\mathbf{x}_i$ 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 $\boldsymbol{\alpha}_1^i, \ldots, \boldsymbol{\alpha}_K^i$:

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

In this approach, because a signal may contain multiple time-shifted copies of the same dictionary signal, convolutive 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.

1.1.3.3 Convolutive analysis dictionary learning

In subsection 1.1.2.2, analysis dictionary learning is formulated as (1.1). Instead of the multiplicative operation between the analysis dictionary and the noiseless signals, *convolutive analysis dictionary learning* convolves the analysis dictionary $\mathbf{W}$ with estimated noiseless signals $\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_n$ so that the resulting signals $\mathbf{w}_1 * \mathbf{x}'_1, \ldots, \mathbf{w}_K * \mathbf{x}'_1, \ldots, \mathbf{w}_1 * \mathbf{x}'_n, \ldots, \mathbf{w}_K * \mathbf{x}'_n$ are sparse [92, 93].

1.2 Supervised learning in time-series

With the extreme increase of temporal data archive, discovering useful information and classifying time-series become challenging and significantly valuable. Time series classifi-

cation (TSC) tasks are applied to many real-world data such as electronic health records, economic records, human activity data and acoustic scene recordings. Due to significant differences in application domains, time series analysis in almost every task that requires expertise and human cognitive process. Therefore, labeling time-series requires excessive amount of labor and time to accurately understand the time-series, especially in the need of incorporating different domain knowledge. The time-series and its classification task is defined as follows:

According to [17], a univariate time series

$$\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T]^T$$

is an ordered set of real values, and the length of $\mathbf{x}$ is the number of real values T. An M-channeled multi-variate time series

$$\mathbf{X} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^M]$$

consists of M different univariate time series with each $\mathbf{x}^i \in \mathbb{R}^T$. For an example, a ECG signal in Figure 1.4 is considered to be a univariate time-series while a spectrogram of bird audio recording in Figure 1.2 is considered as a multi-variate (or multi-channel) time series.

A dataset

$$\mathcal{D} = \{(\mathbf{X}_1, Y_1), (\mathbf{X}_2, Y_2), \ldots, (\mathbf{X}_N, Y_N)\}$$

is a collection of pairs $(\mathbf{X}_i, Y_i)$ where $\mathbf{X}_1$ could either be a univariate or multivariate time series with Y_i as its corresponding class label. The goal is to predict its label given a new time-series $\mathbf{X}$ that is different than the training dataset. In the following, we review

different types of methods in supervised time-series analysis.

1.2.1 Distance-based approaches

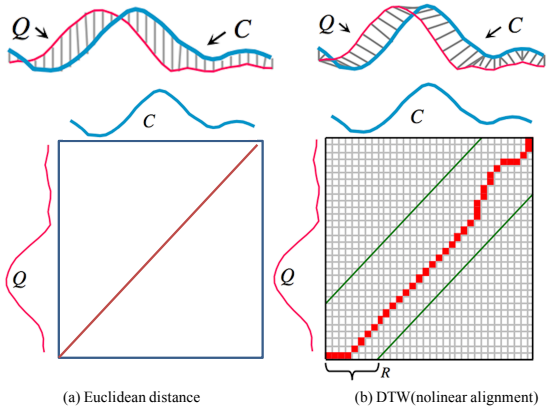


Figure 1.6: Illustration of measuring distance between two sequences for (a) Euclidean distance and (b) a non-linear mapping using DTW.

Traditional approaches of the TSC problem directly treat raw time-series as a feature vector and apply a nearest neighbor classifier [5, 66, 77]. Consider the ordering and temporal information of the time-series, the most effective and popular approach is k-nearest neighbor (NN) classifier coupled with DTW [5]. Calculating a distance between two time-series with an Euclidean distance [66] is inappropriate for time-series data due to its high-dimensionality and commonly observed phase shifts. Instead, a more robust and dynamic distance measure DTW algorithm is proposed to increase the performance of classification accuracy. For time-series with text input data, an edit distance [77] are often considered as a distance measure between two text-based time-series.

1.2.2 Transformation-based approaches

Other approaches use either an ensemble method or a data transformation phase where time series are transformed into a new feature space. One involves finding shapelets (motifs) in the data [142] and the other involves deriving features from varying size intervals of the series, such as bag-of-patterns (BoP) [63]. Similar to BoP, discriminative dictionary learning approaches [72, 74] identify a discriminative basis and jointly learn a classifier by treating the sparse codes as feature vectors. Several approaches have been proposed for dictionary learning in the presences of labels: (i) Learn one dictionary per class [99, 114, 115, 134, 141]; (ii) Prune large dictionaries [31, 133]; (iii) Jointly learn dictionary and classifier [3, 50, 74, 154]; (iv) Embed class labels into the learning of sparse coefficients [35, 57, 81, 140, 152]; and (v) Learn a histogram of dictionary elements over signal constituents [24, 34, 52, 60, 61, 91, 115, 137].

All the transform-based time-series classification approaches aim to capture the localized similarity in time, which tends to be efficient that provides an effective alternative to the traditional approach. Motivated by shapelets and BoP, COTE (Collective Of Transformation-based Ensembles) is developed to use an ensemble of 35 classifiers that does not only ensemble different classifiers over the same transformation, but also ensembles different classifiers over different time series representations [6]. [67] extended COTE with a hierarchical vote system (HIVE-COTE) which leverages a new hierarchical structure with probabilistic voting, including two new classifiers and two additional representation transformation domains. HIVE-COTE is currently considered the state-of-the-art algorithm for time series classification, but requires high computation and intractable in large data in real applications.

1.3 Deep learning approaches

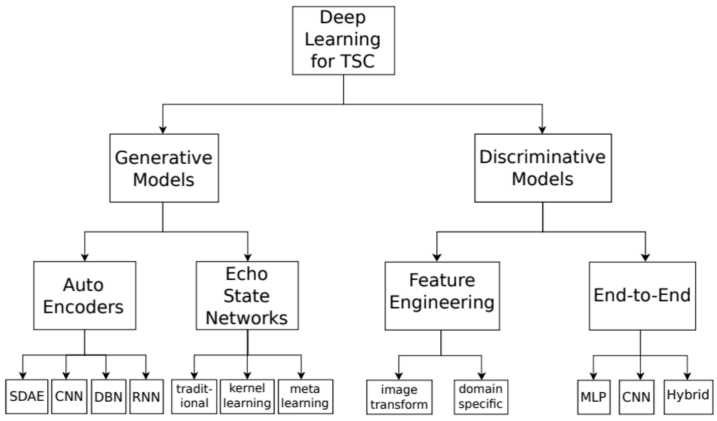


Figure 1.7: A method overview of different deep learning approaches for time series classification (reproduced from [28])

Recently, deep networks have shown significant performance improvements over traditional methods on various machine learning tasks (e.g., in areas of computer vision or speech recognition). Deep architectures can be trained efficiently to learn hidden discriminative features from the raw time series in an end-to-end manner. Deep learning approaches shown in Figure 1.7 can be divided into two main categories: generative and discriminative models.

For all generative approaches, the goal is to find a good representation of time series prior to training a classifier. For an example, to learn the wind pattern with a high-level representation, [42] used the unsupervised stacked denoising auto-encoders (SDAEs) [10] (pre-trained on a rich farm data) and transferred the model by fine tuning to a new farm data. For an effective time-series representation, a fully CNN-based model was introduced to reconstruct a multivariate time series with the same dimension by deconvolution followed by an upsampling technique [79, 131]. To model the latent

features in an unsupervised manner, deep belief networks (DBNs) were proposed and then leveraged to classify univariate and multivariate time series [7, 124]. To capture the time dependencies in the data, [75, 78, 98] introduced a recurrent neural network (RNN) auto-encoder to first generate the time series and then using the learned latent representation to train a classifier (such as SVM or random forest) for the input time series. Other approaches utilize the echo-state networks (ESNs), where they were first introduced for time series prediction in wireless communication channels [48]. ESNs were designed to mitigate the challenges of RNNs by eliminating the need to compute the gradient for the hidden layers which reduces the training time of these neural networks thus avoiding the vanishing gradient problem. A self-predict modeling was proposed in [2, 71] to classify time-series, where ESNs were used to reconstruct the time series. Other ESN-based approaches in [21, 22] defined a kernel over the learned representation followed by an SVM or an MLP classifier.

The discriminative deep learning model directly learns the mapping (a classifier or a regressor) between the raw input of a time series and the class variables in a dataset. Several deep learning approaches proceeded with hand engineered features using the most frequently encountered and computer vision inspired feature extraction method by transforming the time series into images such as Gramian fields [128, 129], recurrence plots [40, 112] and Markov transition fields [130]. For other feature extraction methods, hand-engineered features were used with some domain knowledge, then fed to a deep learning discriminative classifier. For an example, in [113], several features (such as the velocity) were extracted from sensor data placed on a surgeons hand in order to determine the skill level during surgical training. In human activity recognition tasks such as human motion detection using mobile and wearable sensor networks [45], deep learning approaches with several hand-engineered features appear to be effective. Since this type of deep learning

approach is domain agnostic, the end-to-end deep learning, which does not include any domain specific pre-processing steps is desired. The end-to-end deep learning approaches aim to incorporate the feature learning process while fine-tuning the discriminative classifier. In [39, 82], a deep multilayer perceptron (MLP) was designed to learn from scratch a discriminative time series classifier. But the temporal information is not preserved and the features learned are not time-invariant. Since convolutional neural network (CNN) preserves spatial invariance, several variants of CNN models [58, 123, 132, 155] have been considered for time series classification and have shown competitive performance on a subset of the UCR [23] and UEA [5] archive relative to their non-CNN based alternatives. Among such variants of CNN are Residual Networks (ResNets) [41], which add linear shortcut connections for the convolutional layers potentially enhancing the models accuracy. Apart from the UCR/UEA archive, deep learning has reached state-of-the-art performance in different domains such as time-series forecasting in meteorology and oceanography [159], spatio-temporal series forecasting problems [110], human activity recognition from wearable sensors [85], electronic health records [20], surgical skills identification [29], prognostics and health management (PHM) [68], and physiological signals classification [27]. Other types of hybrid architectures in which CNNs were combined with gated recurrent units (GRU) [65] or the attention mechanism [106] also showed promising results on the UCR/UEA archive datasets.

The aforementioned deep learning approaches focus on classifying a fixed size time-series to a single label. However, in many scenarios, time series data may contain events that are associated with different labels at different time instance within the time series. Hence it is important to consider a classification scheme that can provide a label at the time-instance level even when the training data only indicates presence or absence at the entire interval level.

1.4 Weakly-supervised learning in time-series

In many application areas of time-series analysis, such as audio or acoustic scene analysis, a time series is labeled by a multi-label indicating the presence or absence of each class in the time series. For example, for in-situ bio-acoustic monitoring, audio recordings obtained by unattended microphones may contain vocalizations from multiple species including simultaneous vocalizations as well as noise artifacts such as wind, rain, streams, and nearby vehicles.

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. Manually isolating an individual pattern and assigning the appropriate label as a training example in the traditional supervised learning setting is labor intensive. Alternatively, in the weak-supervision setting, each interval containing multiple time-instances, an interval-level label set describing the presence or absence of classes is provided while the individual instance-level label remains unavailable. For weakly supervised dictionary learning, several max margin based, non-convolutive, synthesis dictionary learning approaches are proposed [83, 125, 126]. Other approaches propose to learn a discriminative synthesis dictionary by fully exploiting visual attribute correlations rather than label priors [36, 136]. To the best of our knowledge, the problem of weakly-supervised convolutive analysis dictionary learning has not been studied.

To tackle a dictionary learning under the weak-supervision, especially focusing on time-series data, we use a convolutive analysis dictionary learning approach. The advantage of using analysis dictionary learning model over the synthesis dictionary learning is: (i) Analysis dictionary learning and transform learning offer an alternative to dictionary learning [100, 102]. (ii) It produces a sparsified outcome after applying the analysis dic-

tionary to the original data such that it supports for localizing the target patterns in the data. (iii) it can be applied directly at the test stage and does not require further optimization, while the synthesis dictionary approach may require recovering the activations (coefficients) during the test stage for the classification thereby increasing the computational load during testing. Both synthesis and analysis dictionary learning are used for solving problems such as reconstruction, denoising, and sparse coding. Sometimes, 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 as stated in [102]. However, without an additional supervision component, such methods have been reported to perform sub-optimally in classification tasks [32]. 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.

1.5 Research challenges

The general goal of this research work is to build an automatic system that can efficiently analyze large time-series data. To achieve this research goal, we consider several sub-tasks. One sub-task involves extracting distinct patterns from time-series. This sub-task is often considered as an unsupervised task, in which the data is generated in **the generative setting**. In the generative setting, the data examples are observed as $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$, where each x_n is a time-series signal such as a waveform or a time-frequency image. The goal is to learn distinct signal patterns/motifs or vocalization syllables (dictionary atoms) to represent and reconstruct the original data. Finding

distinct patterns is an important step for understanding the data structure, generating data examples, and for classification.

In real-world applications, efficient labeling and prediction of time-series labels given time-series examples is becoming of great importance [28]. Hence, another sub-task involves developing a model that can detect and localize the patterns of interest from a particular class under the weak supervision setting. In time-series analysis, multiple occurrences of events may be observed in a time-interval. Labeling each event requires extensive efforts. Modeling a system that can efficiently and accurately localize patterns and predict their labels is needed but non-trivial. This sub-task falls under the category of **the discriminative setting**, where the data $\mathcal{X}$ and the associated label sets $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_N\}$ are observed and the problem is not only to predict the coarse class labels but also to localize the exact time index for a particular pattern of interest.

1.6 Objectives

The aim is to provide an inference framework for the problem of learning a convolutive model for time-series data under varying degrees of supervision. To investigate the convolutive modeling of time-series data, we focus on the following tasks:

A. For the generative learning setting:

(i) Develop a robust convolutive model for finding dictionary in a collection of discontinuous time-series data.

(ii) Develop efficient algorithms that solve a large-scale optimization problem in the convolutive model.

(iii) Extend the model to fuse the weak-supervised label fit into the generative ap-

proach objective.

B. For the discriminative learning setting:

(i). Develop a probabilistic model for weakly-supervised learning.

(ii). Develop efficient algorithms that solve maximum likelihood estimation of weakly-supervised learning model.

(iii). Extend the model to allow multiple clusters per class.

(iv). Extend the model to allow basis/atoms to have different scales in an efficient manner.

C. Extend the model to include deep feature learning into the weak-supervision setting.

1.7 Structure of the thesis

The rest of this work is organized as follows. In Chapter 2, we present an efficient **generative convolutive dictionary learning** approach for time-series analysis. In Chapter 3, we focus on discriminative recurring signal detection and localization in the case that examples contain a single event from only one class. In Chapter 4, we develop a novel **weakly-supervised dictionary learning** approach that can address the case in which time series may contain multiple occurrences of events from multiple classes. An extension of our approach to multiple clusters per class is presented in Chapter 5. A multiple-scale dictionary learning model is developed and analyzed in Chapter 6. Finally, we present a summary of the work and some preliminary results on an extension of our approach to deep feature learning for weakly-supervised time series analysis in Chapter 7. A list of publications is included in Chapter 7 and Section 7.3.

Chapter 2: Generative dictionary learning for time-series [1]

Synthesis dictionary learning and analysis dictionary learning are the two approaches to target at minimizing the reconstruction error and sparse representation of the data. In time series, especially bioacoustics audio analysis, the representation of a set of spectrograms using a convolutive mixtures of sparse activations and dictionary words is particularly important. Among the reviewed methods, convolutive non-negative matrix factorization (CNMF) [127] is particularly used as a dictionary learning method. Despite the utility of CNMF in analyzing time-series signals, a few challenges arise when applied to the bioacoustic setting: (i) high computational requirement of CNMF [120] makes it difficult to be applied to large amounts of bioacoustic signals [109]; (ii) CNMF is typically used in a single spectrogram setting, where bioacoustic signals usually contain a collection of discontinuous recordings; and (iii) it is often assumed that the length of the activation signal is the same as the length of the spectrogram in the time domain but it is possible that recordings register only part of a vocalization at the beginning or the end. In this case, a longer activation signal should allow for representing syllable parts in the beginning and the end of the spectrogram. Therefore, we adapt CNMF for a collection of potentially discontinuous spectrograms in which vocalizations may occur prior to the beginning of the recording such that only part of them is observed. The proposed

[1] This chapter is a joint work with J. F. Ruiz-Muñoz. Two publications were associated with this work: Ruiz-Muñoz, José Francisco, Zeyu You, Raviv Raich, and Xiaoli Z. Fern. "Dictionary extraction from a collection of spectrograms for bioacoustics monitoring." *In 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1-6. IEEE, 2015. and Ruiz-Muñoz, José Francisco, Zeyu You, Raviv Raich, and Xiaoli Z. Fern. "Dictionary learning for bioacoustics monitoring with applications to species classification." *Journal of Signal Processing Systems* 90, no. 2 (2018): 233-247. For both of the publications, J. F. Ruiz-Muñoz was the first author.

modification is designed to better suit the convolutive dictionary learning approach to bioacoustic audio recordings which are obtained from multiple sources. To illustrate the merit in this approach, we compare our approach against a standard CNMF approach. To address challenges with computational complexity, we propose a random projected dictionary learning approach. We derive a set of iterations with a choice of step-size that guarantees monotonically decreasing objective. Furthermore, we present an application of the proposed approach for (i) denoising spectrograms, which are corrupted by rain noise and (ii) unsupervised bird syllable discovery, and (iii) supervised classification of bird song recordings.

Since we consider a random projection approach to both spectrograms and dictionary words to reduce the computational complexity, the non-negativity assumption on the dictionary words becomes invalid. Another limitation of the CNMF model is that the activation signal may occur before the time of the first observation. This requires the length of the activations to be greater than the length of the observation signals. To address these issues, we present a convolutive dictionary learning model for bioacoustics.

2.1 Problem statement

We begin by introducing the notations and symbols used in this paper in Table 2.1, and proceed with a formulation of the proposed convolutive dictionary learning model. We denote each spectrogram and each dictionary word as a function of frequency and time by $\mathbf{Y}^i(f,t)$ and $\mathbf{D}_k(f,t)$ respectively, where $\mathbf{Y}^i \in \mathbb{R}^{F\times T}$ for $i = 1, 2, \ldots, N$ and $\mathbf{D}_k \in \mathbb{R}^{F\times W}$. We denote each activation signal as a function of time by $\mathbf{a}_k^i(t)$ where $\mathbf{a}_k^i \in \mathbb{R}^{L\times 1}$. We use $(\cdot)$ in one of the coordinates of a matrix to denote the vector formed by stacking all the elements along the marked coordinate, i.e., $\mathbf{D}_k(f,\cdot) = [\mathbf{D}_k(f,1), \mathbf{D}_k(f,2), \ldots,$

Notation	Explanation
N	number of spectrograms of the dataset
F	number of frequency band
r	number of reduced frequency band
T	number of frames in time per spectrogram
K	number of dictionary words
W	length of each word
L	length of an activation signal ($L = T+W-1$)
$\mathbb{Y}$	$\{\mathbf{Y}^i \in \mathbb{R}^{F\times T} \vert 1 \leq i \leq N\}$, set of N spectrograms
$\mathbb{Y}^Q$	$\{\mathbf{Y}^{Q(i)} \in \mathbb{R}^{r\times T} \vert 1 \leq i \leq N\}$, set of N transformed spectrograms
$\mathbb{D}$	$\{\mathbf{D}_k \in \mathbb{R}^{F\times W} \vert 1 \leq k \leq K\}$, set of K dictionary words
$\mathbb{D}^Q$	$\{\mathbf{D}_k^Q \in \mathbb{R}^{r\times W} \vert 1 \leq k \leq K\}$, set of K transformed dictionary words
$\mathbb{A}$	$\{\mathbf{a}_k^i \in \mathbb{R}^{L\times 1} \vert 1 \leq k \leq K, 1 \leq i \leq N\}$, set of $N \times K$ activation signals

Table 2.1: Overview of the notations used in this paper

$\mathbf{D}_k(f, W)]^T$ for $k = 1, 2, \ldots, K$, and $f = 1, 2, \ldots, F$.

We assume that spectrograms are composed of a sequences of successive spectro-temporal units called dictionary words that are activated at certain time instants. The convolutive dictionary learning approach is targeting at jointly finding a set of K dictionary words $\mathbb{D} = \{\mathbf{D}_1(f,t), \mathbf{D}_2(f,t), \ldots, \mathbf{D}_K(f,t)\}$ and a set of K sparse activation signals $\mathbb{A} = \{\mathbf{a}_1^1(t), \ldots, \mathbf{a}_K^1(t), \mathbf{a}_1^2(t), \ldots, \mathbf{a}_K^2(t),\ \mathbf{a}_1^N(t), \ldots, \mathbf{a}_K^N(t)\}$ such that $\mathbf{Y}^i(f,t) \approx \sum_{k=1}^K \mathbf{a}_k^i(t) * \mathbf{D}_k(f,t)$ for $1 \leq f \leq F, 1 \leq t \leq T, 1 \leq i \leq N$ (e.g., see Fig. 2.1). The goal is to minimize the distance between the original and the reconstructed spectrograms. Learning a convolutive dictionary model can be formulated as the following optimiza-

tion problem:

$$\begin{aligned} \underset{\mathbb{D},\mathbb{A}}{\text{minimize}} \quad & \sum_{i=1}^{N}\Big[\sum_{f=1}^{F}\sum_{t=1}^{T}(\mathbf{Y}^i(t,f)-\sum_{k=1}^{K}\mathbf{a}_k^i(t)*\mathbf{D}_k(t,f))^2 \\ & \qquad +\lambda\sum_{k=1}^{K}\sum_{t=1}^{L}|\mathbf{a}_k^i(t)|\Big] \qquad (2.1) \\ \text{subject to} \quad & \sum_{f=1}^{F}\sum_{t=1}^{W}(\mathbf{D}_k(f,t))^2 \le 1,\ \forall\ 1\le k\le K. \end{aligned}$$

Under this convolutive model, since each frequency band or each spectrogram can be

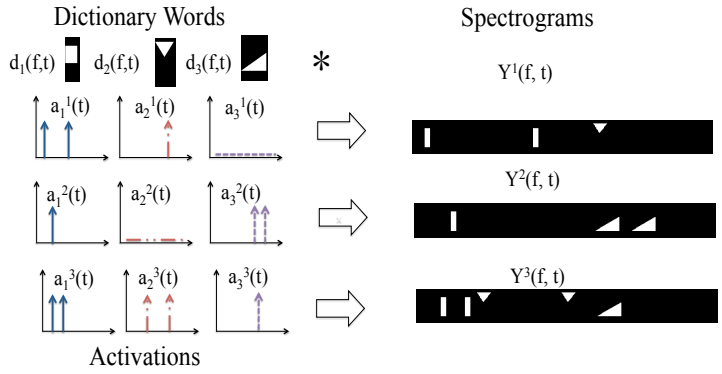


Figure 2.1: A convolutive model for dictionary learning (reproduction of [103])

applied with 1-D convolution separately, the objective in (2.1) can be reformulated using Toeplitz matrices. Given a vector $\mathbf{x} = [x(1), x(2), \cdots, x(m)]$, the Toeplitz matrix $\mathbf{T}(\mathbf{x}, c, p, w) \in \mathbb{R}^{(p-c+1)\times w}$ is:

$$\mathbf{T} = \begin{bmatrix} x(c) & x(c-1) & \cdots & x(c-w+1) \\ x(c+1) & x(c) & \cdots & x(c-w+2) \\ \vdots & \cdots & \cdots & \vdots \\ x(p) & x(p-1) & \cdots & x(p-w+1) \end{bmatrix}.$$

Using a coordinate descent approach, the minimization in (2.1) can be facilitated by alternating between the dictionary learning problem

$$(DL) \quad \underset{\mathbf{d}^1,\mathbf{d}^2,\ldots,\mathbf{d}^F}{\text{minimize}} \quad \sum_{f=1}^{F} \|\mathbf{y}_f - \mathbf{T}_A \mathbf{d}^f\|^2 \qquad \text{subject to} \quad \|\mathbf{D}_k\|^2 \leq 1, \ \forall 1 \leq k \leq K, \tag{2.2}$$

where $\mathbf{y}_f = [\mathbf{Y}^1(f,\cdot), \mathbf{Y}^2(f,\cdot), \ldots, \mathbf{Y}^N(f,\cdot)]^T \in \mathbb{R}^{NT\times 1}$ for $f = 1,2,\ldots,F$, $\mathbf{d}^f = [\mathbf{D}_1(f,\cdot), \mathbf{D}_2(f,\cdot), \ldots, \mathbf{D}_K(f,\cdot)]^T \in \mathbb{R}^{KW\times 1}$ for $f = 1,2,\ldots,F$ and $\mathbf{T}_A \in \mathbb{R}^{NT\times KW}$ given by

$$\mathbf{T}_A = \begin{bmatrix} \mathbf{T}(\mathbf{a}_1^1, W, L, W) & \cdots & \mathbf{T}(\mathbf{a}_K^1, W, L, W) \\ \mathbf{T}(\mathbf{a}_1^2, W, L, W) & \cdots & \mathbf{T}(\mathbf{a}_K^2, W, L, W) \\ \vdots & \cdots & \vdots \\ \mathbf{T}(\mathbf{a}_1^N, W, L, W) & \cdots & \mathbf{T}(\mathbf{a}_K^N, W, L, W) \end{bmatrix}$$

and activation extraction problem

$$(AE) \quad \underset{\mathbf{a}^1,\mathbf{a}^2,\ldots,\mathbf{a}^N}{\text{minimize}} \quad \sum_{i=1}^{N} (\|\mathbf{y}^i - T_D \mathbf{a}^i\|^2 + \lambda \|\mathbf{a}^i\|_1), \tag{2.3}$$

where $\mathbf{y}^i = [\mathbf{Y}^i(1,\cdot)^T, \mathbf{Y}^i(2,\cdot)^T, \ldots, \mathbf{Y}^i(F,\cdot)^T]^T \in \mathbb{R}^{FT\times 1}$ for $i = 1,2,\ldots,N$, $\mathbf{a}^i = [{\mathbf{a}_1^i}^T, {\mathbf{a}_2^i}^T, \ldots, {\mathbf{a}_K^i}^T]^T \in \mathbb{R}^{KL\times 1}$ for $i = 1,2,\ldots,N$, and $\mathbf{T}_D \in \mathbb{R}^{FT\times KL}$ given by

$$\mathbf{T}_D = \begin{bmatrix} \mathbf{T}(\mathbf{D}_1(1,\cdot), W, L, L) & \cdots & \mathbf{T}(\mathbf{D}_K(1,\cdot), W, L, L) \\ \mathbf{T}(\mathbf{D}_1(2,\cdot), W, L, L) & \cdots & \mathbf{T}(\mathbf{D}_K(2,\cdot), W, L, L) \\ \vdots & \cdots & \vdots \\ \mathbf{T}(\mathbf{D}_1(F,\cdot), W, L, L) & \cdots & \mathbf{T}(\mathbf{D}_K(F,\cdot), W, L, L) \end{bmatrix}.$$

Constructing $\mathbf{T}_D$ and $\mathbf{T}_A$ Toeplitz matrices is memory inefficient and solving the above alternating quadratic programming problem with matrix inversion is time consuming. To reduce the computational complexity and the memory issue of the convolutive model, we

propose a random projected convolutive model with modified gradient descent algorithm that utilizes the convolution operator.

2.2 Solution approach for dictionary learning and activation extraction

Consider the (DL) problem in (2.2), least square solution with normalization or projected Newton descent method are both simple to derive. Consider the L_1 regularized (AE) problem in (2.3), Least-angle-regression (LARS) algorithm [26] or feature-sign sparse coding algorithm are also applicable [59]. However, these algorithms require a large matrix inversion to obtain an efficient and exact solution. In our problem, computing $\mathbf{T}_D^T\mathbf{T}_D$ and $\mathbf{T}_A^T\mathbf{T}_A$ requires a computational complexity of the order $\mathcal{O}(FKT\log T)$ and $\mathcal{O}(NKT\log T)$ respectively and computing their inverse requires $\mathcal{O}((KL)^3)$ and $\mathcal{O}((KW)^3)$ respectively, which limits the practical applicability of the approach. Hence, we propose an optimization transfer algorithm to minimize (2.1). In optimization transfer, a surrogate function $g(x, x')$ is considered as a replacement to the original objective $f(x)$ such that (i) $f(x) \leq g(x, x'),\ \forall x, x'$ and (ii) $f(x') = g(x', x'),\ \forall x'$. The update iteration $x^{(j+1)} = \arg\min_x g(x, x')$ guarantees $f(x^{(j+1)}) \leq f(x^{(j)})$. The update rules are derived by minimizing a surrogate such that the computation is reduced by utilizing the fast Fourier transform (FFT) implementation of the discrete Fourier transform (DFT) and its inverse.

2.2.1 Random projected dictionary learning

The convolutive model provides a natural representation for spectrograms of bird vocalizations. To reduce the computational complexity, we propose to make use of the fact that bird vocalizations are concentrated in a small range of frequencies. Consequently, spectrograms of bird vocalization tend to have sparse columns. We consider a compressive sampling approach to facilitated the reduction in computational complexity.

To reduce the computational complexity, we apply the same transformation to both the spectrogram side and dictionary word side. In such way, the computational complexity of computing both dictionary words and activations is decreased by reducing the number of unknowns. The new formulation of the dictionary learning is

$$\begin{aligned} \underset{\mathbb{D}^Q,\mathbb{A}}{\text{minimize}} \quad & \sum_{i=1}^{N}(\sum_{c=1}^{r}\sum_{t=1}^{T}(\mathbf{Y}^{Q(i)}(c,t) - \sum_{k=1}^{K}\mathbf{a}_k^i(t) * \mathbf{D}_k^Q(c,t))^2 \\ & \qquad\qquad +\lambda\sum_{k=1}^{K}\sum_{t=1}^{L}|\mathbf{a}_k^i(t)|) \\ \text{subject to} \quad & \sum_{c=1}^{r}\sum_{t=1}^{W}\mathbf{D}_k^Q(c,t)^2 \leq 1,\ \forall\ 1 \leq k \leq K \end{aligned} \tag{2.4}$$

with a transformation matrix $\mathbf{Q} = [\mathbf{Q}(1), \mathbf{Q}(2), \ldots, \mathbf{Q}(F)] \in \mathbb{R}^{r\times F}$, where $\mathbf{Q}(f) = [q_1(f), q_2(f), \ldots, q_r(f)]^T \in \mathbb{R}^r$ such that $r < F$. Note that $\mathbf{Y}^{Q(i)} = \mathbf{Q}\mathbf{Y}^i$ and $\mathbf{D}_k^Q = \mathbf{Q}\mathbf{D}_k$.

Many dimension reduction techniques can be considered when generating the transformation matrix $\mathbf{Q}$, e.g., principal component coefficients (PCC) and Mel-frequency cepstral coefficients (MFCCs). But the problem of signal or spectrogram distortion and the difficulty of recovering the original signal or spectrogram may arise. For example, if the intensities at several frequency bins are compressed into a single coefficient using MFCC, it is difficult to recover the their value from the single coefficient. To prevent

a potential distortion problem, we apply a compressive transformation with a random matrix [122]. We rely on the sparsity of the signal and the compressive approach to improve recovery. The recovery of the spectrograms or dictionary words can be implemented using a linear programming approach [8, 101].

2.2.2 Dictionary learning

Since we consider an optimization transfer approach (i.e., majorization-minimization [44]) to facilitate an iterative minimization of the objective in (2.4), our goal is to identify an efficient surrogate for our DL objective. The following inequality

$$\begin{aligned}\frac{1}{2}\|\mathbf{y}^f - T_A\mathbf{d}^f\|^2 &= \frac{1}{2}\|T_A(\mathbf{d}^f - \mathbf{d}^{f'}) - (\mathbf{y}^f - T_A\mathbf{d}^{f'})\|^2 \\ &\leq \frac{\gamma_f}{2}\|\mathbf{d}^f - \mathbf{d}^{f'}\|^2 - T_A^T(\mathbf{y}_f - T_A\mathbf{d}^{f'})\mathbf{d}^f + \text{const.} \\ &= \frac{\gamma_f}{2}\|\mathbf{d}^f - (\mathbf{d}^{f'} + \frac{1}{\gamma_f}T_A^T(\mathbf{y}^f - T_A\mathbf{d}^{f'}))\|^2 + \text{const.}, \end{aligned} \tag{2.5}$$

provides a surrogate to $\sum_{f=1}^F \frac{1}{2}\|\mathbf{y}^f - \mathbf{T}_A\mathbf{d}^f\|^2$. To satisfy the inequality, we choose $\gamma_d = \max_f \gamma_f \geq \max_f \|T_A(\mathbf{d}^f - \mathbf{d}^{f'})\|^2/\|\mathbf{d}^f - \mathbf{d}^{f'}\|^2$. Replacing the objective using the surrogate in (2.5) and minimizing with respect to the dictionary yields

$$\begin{aligned}\underset{\mathbf{d}^1,\mathbf{d}^2,\ldots,\mathbf{d}^F}{\text{minimize}} \quad & \sum_{f=1}^F \frac{\gamma_d}{2}\|\mathbf{d}^f - \mathbf{g}^f\|^2 \\ \text{subject to} \quad & \|D_k\|^2 \leq 1,\ \forall 1 \leq k \leq K,\end{aligned} \tag{2.6}$$

where $\mathbf{g}^f = \mathbf{d}^{f'} + \frac{1}{\gamma_d}V_D$ and $V_D = T_A^T(\mathbf{y}^f - T_A\mathbf{d}^{f'})$. We denote $\mathbf{d}^f = [\mathbf{d}_1^{f\,T}, \ldots, \mathbf{d}_K^{f\,T}]^T$ and $\mathbf{g}^f = [\mathbf{g}_1^{f\,T}, \ldots, \mathbf{g}_K^{f\,T}]^T$, where $\mathbf{g}_k^f = \mathbf{d}_k^{f'} + \frac{1}{\gamma_d}V_D^k$ and $V_D^k = T_A^{k\,T}(\mathbf{y}^f - T_A^k\mathbf{d}_k^{f'})$. To solve (2.6), we form the Lagrangian $L(\mathbb{D}, \mathbf{fi}) = \sum_{f=1}^F \sum_{k=1}^K \frac{\gamma_d}{2}\|\mathbf{d}_k^f - \mathbf{g}_k^f\|^2 + \sum_{k=1}^K \beta_k(\sum_{f=1}^F \|\mathbf{d}_k^f\|^2 - 1)$.

Minimizing the Lagrangian with respect to $\mathbf{d}_k^f$ results in $\mathbf{d}_k^f = \frac{\gamma_d}{\gamma_d+2\beta_k}\mathbf{g}_k^f$. Substituting $\mathbf{d}_k^f$ back into the Lagrangian yields the dual function $\sum_{k=1}^{K}\big[\beta_k(\frac{\gamma_d}{\gamma_d+2\beta_k}\sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2-1)\big]$. Maximizing the dual objective with respect to β_k subject to $\beta_k \geq 0$ yields

$$\begin{cases} \beta_k^* = 0, & \sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2 \leq 1; \\ \frac{\gamma_d}{\gamma_d+2\beta_k^*} = \frac{1}{\sqrt{\sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2}}, & \text{Otherwise.} \end{cases}$$

To obtain the optimal $\mathbf{d}_k^f$ we replacing $\beta_k = \beta_k^*$ back into $\mathbf{d}_k^f = \frac{\gamma_d}{\gamma_d+2\beta_k}\mathbf{g}_k^f$ and obtain

$$\mathbf{d}_k^f = \begin{cases} \mathbf{g}_k^f & \sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2 \leq 1; \\ \frac{\mathbf{g}_k^f}{\sqrt{\sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2}} & \text{Otherwise.} \end{cases} \tag{2.7}$$

Finally, replacing $\mathbf{g}_k^f = \mathbf{d}_k^{f'} + \frac{1}{\gamma_d}V_D^k$ back into (2.7) yields

$$\mathbf{d}_k^{f(j+1)} = \begin{cases} \mathbf{d}_k^{f(j)} + \frac{1}{\gamma_d}\mathbf{v}_D^k, & \sum_f \|\mathbf{d}_k^{f(j)} + \frac{1}{\gamma_d}\mathbf{v}_\mathbf{D}^k\|^2 \leq 1; \\ \frac{\mathbf{d}_k^{f(j)}+\frac{1}{\gamma_d}\mathbf{v}_D^k}{\sqrt{\sum_f \|\mathbf{d}_k^{f(j)}+\frac{1}{\gamma_d}\mathbf{v}_D^k\|^2}}, & \text{otherwise,} \end{cases} \tag{2.8}$$

where $\mathbf{v}_D^k = {\mathbf{T}_A^k}^T(\mathbf{y}_f - \mathbf{T}_A^k\mathbf{D}_k^{(j)}(f,\cdot))$ and $\mathbf{T}_A^k = [\mathbf{T}(\mathbf{a}_k^1, W, L, W)^T, \ldots, \mathbf{T}(\mathbf{a}_k^N, W, L, W)^T]^T$.

Step-size selection for the DL update: To determine the step size γ_d, we consider two cases. When the updated dictionary words satisfies the constraint that $\sum_{f=1}^{F}\|\mathbf{g}_k^f\|^2 \leq 1$, the optimal step-size $\gamma_d^* = \frac{\|T_A V_D\|^2}{\|V_D\|^2}$. When the constraints are not satisfied, the optimal step-size has no closed-form solution. Setting $\gamma_d = \max_{\mathbf{v}}\frac{\|\mathbf{T}_A\mathbf{v}\|^2}{\|\mathbf{v}\|^2} = \lambda_{\max}(\mathbf{T}_A^T\mathbf{T}_A)$ ensures that $\|\mathbf{T}_A\mathbf{v}\|^2 \leq \gamma_d\|\mathbf{v}\|^2$ for any $\mathbf{v}$. This conservative approach results in a small step size $1/\gamma_d$, which leads to a slow convergence rate. To improve this, we consider the

following tighter bound on γ_d. We rely on maximizing first individual for each f and then take the maximum over all fs.

From (2.8), we have $\mathbf{d}^f - \mathbf{d}^{f'} = c(\gamma_d)(\mathbf{d}^{f'} + \frac{1}{\gamma_d}\mathbf{v}_D) - \mathbf{d}^{f'} = \alpha_1 \mathbf{d}^{f'} + \alpha_2 \mathbf{v}_D$. Since $\mathbf{d}^f - \mathbf{d}^{f'} \in \text{span}\{\mathbf{d}^{f'}, \mathbf{v}_D\}$, we can further restrict γ_d without violating the bound on γ_d. Using Gram–Schmidt orthogonalization, we obtain the orthogonal basis for $[\mathbf{v}_d, \mathbf{d}^{f'}]$ as $\mathbf{u}_1 = \mathbf{v}_d/\|\mathbf{v}_d\|$ and $\mathbf{u}_2 = \tilde{\mathbf{d}}^{f'}/\|\tilde{\mathbf{d}}^{f'}\|$, where $\tilde{\mathbf{d}}^{f'} = \mathbf{d}^{f'} - (\mathbf{d}^{f'\,T}\mathbf{u}_1)\mathbf{u}_1$. For every value of (α_1, α_2) in the representation of $\mathbf{d}^f - \mathbf{d}^{f'} = \alpha_1 \mathbf{d}^{f'} + \alpha_2 \mathbf{v}_D$ there exists a (β_1, β_2) in the equivalent representation of $\mathbf{d}^f - \mathbf{d}^{f'} = \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2$. Hence, we can find γ_f by maximizing the following with respect to (β_1, β_2):

$$\frac{\|T_A(\mathbf{d}^f - \mathbf{d}^{f'})\|^2}{\|\mathbf{d}^f - \mathbf{d}^{f'}\|^2} = \frac{\|\mathbf{T}_A[\mathbf{u}_1, \mathbf{u}_2][\beta_1, \beta_2]^T\|^2}{\|[\beta_1, \beta_2]^T\|^2}.$$

Consequently, we can bound $\frac{\|T_A(\mathbf{d}^f - \mathbf{d}^{f'})\|^2}{\|\mathbf{d}^f - \mathbf{d}^{f'}\|^2}$ by

$$\gamma_f = \lambda_{\max}([\mathbf{u}_1, \mathbf{u}_2]^T \mathbf{T}_A^T \mathbf{T}_A [\mathbf{u}_1, \mathbf{u}_2]).$$

Note that although $\mathbf{T}_A^T \mathbf{T}_A$ is independent of frequency f, it is fairly large and its associated eigen-decomposition may be computationally intensive. Instead, we replace it with the eigen-decomposition of F 2×2 f-dependent matrices $[\mathbf{u}_1, \mathbf{u}_2]^T \mathbf{T}_A^T \mathbf{T}_A [\mathbf{u}_1, \mathbf{u}_2]$. To ensure that the bound holds for every f, we select the step size $\gamma_d^* = \max_f \gamma_f$.

2.2.3 Activation Extraction

Similarly to DL, we consider an optimization transfer approach to facilitate an iterative approach to the minimization of the objective in (2.4) with respect to the activations.

Similar bounding technique yields

$$\frac{1}{2}\|\mathbf{y}^i - T_D\mathbf{a}^i\|^2 \leq \frac{\gamma_a}{2}\|\mathbf{a}^i - (\mathbf{a}^{i'} + \frac{1}{\gamma_a}T_D^T(\mathbf{a}^i - T_D\mathbf{a}^{i'}))\|^2 + \text{const.},$$

where $\gamma_a \geq \frac{\|T_D(\mathbf{a}^i - \mathbf{a}^{i'})\|^2}{\|\mathbf{a}^i - \mathbf{a}^{i'}\|^2}$. Consequently, the surrogate problem of (AE) is defined as:

$$\underset{\mathbf{a}^1,\mathbf{a}^2,\ldots,\mathbf{a}^N}{\text{minimize}} \quad \sum_{i=1}^{N} \frac{\gamma_a}{2}(\|\mathbf{a}^i - \mathbf{h}^i\|^2 + \lambda\|\mathbf{a}^i\|_1), \tag{2.9}$$

where $\mathbf{h}^i = \mathbf{a}^{i'} + \frac{1}{\gamma_a}V_A = [h_1^i(1), \ldots, h_1^i(L), h_K^i(1), \ldots, h_K^i(L)]^T$ and $V_A = T_D^T(\mathbf{y}^i - T_D\mathbf{a}^{i'})$. Note that the objective in (2.9) is separable:

$$\sum_{i=1}^{N} \frac{\gamma_a}{2}(\|\mathbf{a}^i - \mathbf{h}^i\|^2 + \lambda\|\mathbf{a}^i\|_1) = \frac{\gamma_a}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{t=1}^{L}(a_k^i(t) - h_k^i(t))^2 + \lambda|a_k^i(t)|).$$

Consequently, the solution to (2.9) can be obtained by solving element-wise for every $a_k^i(t)$. The optimal solution to each element $a_k^i(t)^*$ is obtained by a thresholding function as $a_k^i(t)^* = S_{\frac{\lambda}{\gamma_a}}(h_k^i(t))$ where $S_{\frac{\lambda}{\gamma_a}}(h_k^i(t))$ is a soft-thresholding function defined as

$$S_{\frac{\lambda}{\gamma_a}}(h_k^i(t)) = \begin{cases} h_k^i(t) + \frac{\lambda}{\gamma_a}, & h_k^i(t) < -\frac{\lambda}{\gamma_a}; \\ h_k^i(t) - \frac{\lambda}{\gamma_a}, & h_k^i(t) > \frac{\lambda}{\gamma_a}; \\ 0, & Otherwise. \end{cases}$$

The resulting update rule for extracting the activation signal $\mathbf{a}_k^{i(j)}(t)$ at iteration j follows the iterative soft-thresholding approach as

$$\mathbf{a}_k^{i(j+1)}(t) = \begin{cases} \mathbf{a}_k^{i(j)}(t) + \frac{1}{\gamma_a}(\mathbf{v}_A^k(t) - \lambda), & \mathbf{a}_k^{i(j)}(t) + \frac{1}{\gamma_a}(\mathbf{v}_A^k(t) - \lambda) > 0 \\ \mathbf{a}_k^{i(j)}(t) + \frac{1}{\gamma_a}(\mathbf{v}_A^k(t) + \lambda), & \mathbf{a}_k^{i(j)}(t) + \frac{1}{\gamma_a}(\mathbf{v}_A^k(t) + \lambda) < 0 \\ 0, & \text{otherwise,} \end{cases} \tag{2.10}$$

where $\mathbf{v}_A^k = (\mathbf{T}_D^k)^T(\mathbf{y}^i - \mathbf{T}_D\mathbf{a}^{i(j)})$ and $\mathbf{T}_D^k = [\mathbf{T}(\mathbf{D}_k(1,\cdot), W, L, L)^T, \ldots, \mathbf{T}(\mathbf{D}_k(F,\cdot), W, L, L)^T]^T$.

Step-size selection for the AE update: Since the optimal step-size for activation updates must satisfy

$$\gamma_a \geq \|\mathbf{T}_D(\mathbf{a}^i - \mathbf{a}^{i'})\|^2 / \|\mathbf{a}^i - \mathbf{a}^{i'}\|^2,$$

we can bound γ_a by $\lambda_{\max}(\mathbf{T}_D^T\mathbf{T}_D)$, which is the largest eigenvalue of the matrix $\mathbf{T}_D^T\mathbf{T}_D$. Computing the largest eigenvalue of a $KL \times KL$ matrix is costly. Instead, we apply the DFT operator and further bound the maximum eigenvalue of $\mathbf{T}_D^T\mathbf{T}_D$. Using Parseval Theorem and Cauchy-Schwartz inequality, we derive the following bound

$$\begin{aligned} \frac{\|\mathbf{T}_D(\mathbf{a}^i - \mathbf{a}^{i'})\|^2}{\|\mathbf{a}^i - \mathbf{a}^{i'}\|^2} &= \sum_{f=1}^{F} \frac{\|\sum_{k=1}^K D_k(f,\cdot) * (\mathbf{a}_k^i(\cdot) - \mathbf{a}_k^{i'}(\cdot))\|^2}{\sum_{k=1}^K \|\mathbf{a}_k^i - \mathbf{a}_k^{i'}\|^2} \\ &\leq \sum_{f=1}^{F} \frac{\frac{1}{2\pi}\int_{-\pi}^{\pi} |\sum_{k=1}^K \hat{\mathbf{D}}_k(f,\omega)(\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega))|^2 d\omega}{\sum_{k=1}^K \frac{1}{2\pi}\int_{-\pi}^{\pi} |\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega)|^2 d\omega} \\ &\leq \sum_{f=1}^{F} \frac{\int_{-\pi}^{\pi} \sum_{k=1}^K |\hat{\mathbf{D}}_k(f,\omega)|^2 \sum_k |\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega)|^2 d\omega}{\int_{-\pi}^{\pi} \sum_{k=1}^K |\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega)|^2 d\omega} \\ &\leq \sum_{f=1}^{F} \frac{\max_\omega \sum_{k=1}^K |\hat{\mathbf{D}}_k(f,\omega)|^2 \int_{-\pi}^{\pi} \sum_k |\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega)|^2 d\omega}{\int_{-\pi}^{\pi} \sum_{k=1}^K |\hat{\mathbf{a}}_k^i(\omega) - \hat{\mathbf{a}}_k^{i'}(\omega)|^2 d\omega} \\ &\leq \sum_{f=1}^{F} \max_\omega \sum_{k=1}^K |\hat{\mathbf{D}}_k(f,\omega)|^2. \end{aligned}$$

Hence, $\frac{\|\mathbf{T}_D(\mathbf{a}^i-\mathbf{a}^{i'})\|^2}{\|\mathbf{a}^i-\mathbf{a}^{i'}\|^2}$ can be upper bounded by $\gamma_a^* = \sum_{f=1}^{F} \max_\omega \sum_{k=1}^{K} |\hat{\mathbf{D}}_k(f,\omega)|^2$, where $\hat{\mathbf{D}}_k(f,\omega)$ is the ωth coefficient in the frequency domain on the Discrete Fourier Transform of $\mathbf{D}_k(f,\cdot)$ at frequency band f for dictionary word k.

2.2.4 Solution approach for random projection model

For the random projection approach, we simply replace $\mathbb{Y}$ with $\mathbb{Y}^Q$ and $\mathbb{D}$ with $\mathbb{D}^Q$. We propose an efficient algorithm for practical dictionary extraction. The algorithm consists of three main parts: (i) Transforming the original input spectrograms using a random projection matrix, (ii) Alternatively solving the (DL) and (AE) until a convergence criterion is met, and (iii) Recovering the uncompressed domain dictionary words by solving the (DL) problem with the extracted activations and the original data $\mathbb{Y}$.

2.2.5 Computational complexity

Since the convolution operator with length L can be computed more efficiently by using FFT and inverse fast Fourier transform (IFFT), the computational complexity for each convolution block with size L is $\mathcal{O}(L \log L)$. For the iterative procedure in (2.8), calculating $\mathbf{T}_A\mathbf{d}^f$, $\mathbf{V}_D$ and γ_d^* all require $\mathcal{O}(NKL \log L)$, therefore the overall computational complexity for (DL) is $\mathcal{O}(FNKL \log L)$. Updating the activations produces the same computational complexity as $\mathcal{O}(NFKL \log L)$. The total computational complexity for the algorithm without random projection is $\mathcal{O}(FNKL \log L)$. With random projection, the computational complexity is proportional to the original computation complexity of $\mathcal{O}(FNKL \log L)$. If the reduced frequency band r is 20% of the original frequency band F, the running time will be five times faster than the uncompressed dictionary learning

algorithm, which makes the convolutive dictionary learning method more efficient and practical.

2.3 Extension to classification framework

Dictionary learning is not limited to spectrogram reconstruction or denoising. It can be considered as a preprocessing step for classification [33]. We present a dictionary-based classification step that aims to use the learned sparse representation for classifying bioacoustic recordings. The proposed scheme is inspired by the framework used in music analysis [144]. In Fig. 2.2, we present the classification framework in two parts: i) training and ii) test. In the first part, the dictionary words and activation signals are estimated from the training set, a set of features is extracted from the activations signals which is used for training an SVM classifier. In the second part, the activations signals corresponding to the test set are estimated using the dictionary previously learned. Then features are extracted based on the activations. Finally, the features are provided as an input tor the SVM classifier. The supervised dictionary learning adaptation, feature extraction and SVM for training and classification are explained below.

2.3.1 Supervised dictionary learning

For classification, we consider the case in which each recording may contain dictionary words from multiple classes. In our application, vocalizations in the same recording may come from multiple bird species. This classification framework has been considered for species recognition of in-situ recordings [15]. This setting is often referred to as the multiple label setting. In the multiple label setting, recording i is associated with a

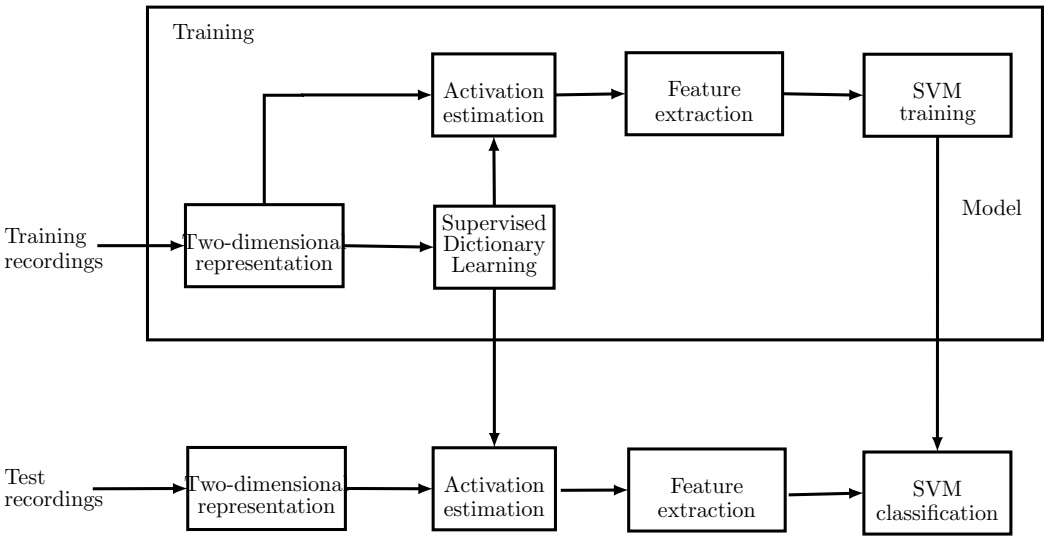


Figure 2.2: Diagram of dictionary-based classification (reproduction of [103].

label vector $[h_{i1}, h_{i2}, \ldots, h_{iM}]^T$ in which $h_{ij} \in \{0, 1\}$. The j entry of the label vector is binary and indicates the presence (by 1) or the absence (by 0) of the j species in the ith recording. The label information can therefore be summarize using the matrix $\mathbf{H} \in \mathbb{R}^{N \times M}$ where $\mathbf{H}_{ij} = h_{ij}$. To adopt the dictionary learning approach to this setting, we assume that the dictionary consists of M sub-dictionaries (one for each class). The jth sub-dictionary consists of K_j words and the total number of dictionary words is K such that $\sum_{j=1}^{M} K_j = K$. Moreover, each dictionary word is affiliated with one class only. We use the mapping $S(\cdot) : [1, 2, \ldots, K] \rightarrow [1, 2, \ldots, M]$ to indicate this affiliation. Hence the set of all dictionary words associated with class j is $\mathcal{S}_j = \{k |\ k \in [1, 2, \ldots, K],\ S(k) = j\}$ and its cardinality is $K_j = |\mathcal{S}_j|$. Moreover, we assume that dictionary words of a class can only be used to construct a given spectrogram if that class is present in the spectrogram. Alternatively, if the class is absent from spectrogram i, the activations associate with its dictionary words $a_k^i(t) = 0$ for $t = 1, 2, \ldots, T$. Consequently, we extend our formulation

to

$$\underset{\mathbb{D},\mathbb{A}}{\text{minimize}} \sum_{i=1}^{N}\Big[\sum_{f=1}^{F}\sum_{t=1}^{T}(\mathbf{Y}^i(t,f) - \sum_{k=1}^{K}\mathbf{a}_k^i(t) * \mathbf{D}_k(t,f))^2 + \lambda \sum_{k=1}^{K}\sum_{t=1}^{L}|\mathbf{a}_k^i(t)|\Big]$$
$$\text{subject to} \sum_{f=1}^{F}\sum_{t=1}^{W}(\mathbf{D}_k(f,t))^2 \leq 1,\ \forall\ 1 \leq k \leq K. \tag{2.11}$$
$$\text{and } a_k^i(t) = 0 \text{ for } t = 1, 2, \ldots, T \text{ if } h_{ij} = 0,\ k \in \mathcal{S}_j, \quad j \in [1, \ldots, M]$$

2.3.2 Feature extraction

We consider using a summarization of the activations as a feature vector that will provide information about the presence or absence of a given class in a recording. To this end, we map the set of activations of the i-th spectrogram $\{\mathbf{a}_k^i = [a_{k1}^i, \ldots, a_{kL}^i] \in \mathbb{R}^L \times 1 | 1 \leq k \leq K\}$ to a vector where its dimension is the number of estimated features. Therefore, we compute a vector $\mathbf{g}_i = [g_{i1} \ldots g_{iK}]$ where

$$g_{ik} = \frac{\sum_{t=1}^{L}|a_{kt}^i|}{\sum_{k=1}^{K}\sum_{t=1}^{L}|a_{kt}^i|}.$$

In the summarization process, for each activation, the entire activation time series is first replaced with its l_1 norm. Then, the l_1 norms are scaled by the sum of l_1 norms to make $g_{i\cdot}$ sum to one. We use the set of feature extracted from activation signals as input of a support vector machine (SVM) classifier.

2.3.3 SVM classifier

For training the SVM classifier, we use two types of kernels:

- Histogram intersection kernel (HIK): which is computed as follows

$$K_{HI}(\mathbf{g}_i, \mathbf{g}_j) = \sum_{k=1}^{K} \min(g_{ik}, g_{jk}).$$

  Notice that the range of this kernel is $[0, 1]$ due to the applied normalization ($\sum_{k=1}^{K} g_{ik} = 1$).

- Exponential kernel (K_{E_P}):

$$K_{E_P}(\mathbf{g}_i, \mathbf{g}_j) = e^{-\frac{||\mathbf{g}_i - \mathbf{g}_j||_p^p}{\delta}}$$

  where the parameter P is usually 1 or 2, and δ requires being tuned.

2.4 Results and Analysis

In this section, we empirically evaluate the proposed random projected dictionary learning approach on both synthetic and real data. First, we compare how the boundary effect is addressed by our approach and CNMF. Additionally, we evaluate the proposed approach for the problems of denoising, dictionary discovery and classification of birdsong recordings.

2.4.1 Analysis on synthetic data

In this case, we use three spectrograms synthetically generated with three dictionary words and their corresponding sparse activation signals. The dimensions of each spectrogram are fixed to $F = 50 \times T = 500$, and the dimensions of each dictionary word are $F = 50 \times W = 50$.

The learned dictionary words for these three spectrograms and activations using our approach and CNMF [107] are shown in Fig. 2.3. The number of iterations is $10,000$ in both cases. We observe the proposed approach accurately recovers the dictionary words (see Fig. 2.3(e)) and the spectrograms (see Fig. 2.3(c)) despite the boundary effect in the first spectrogram in Fig. 2.3(b). However, CNMF learns each dictionary word as a mixture of the original dictionary words (see Fig. 2.3(g)) including the part of the dictionary word appearing in the beginning of the first spectrogram, and it fails to recover the spectrograms(see Fig. 2.3(d)). As it can be seen, our model is more robust to boundary effects than CNMF.

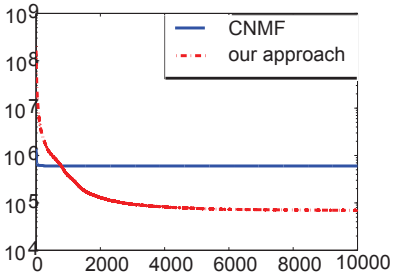
2.4.2 Analysis on real-world data

In order to apply our random projected convolutive dictionary learning approach for birdsong analysis tasks, we use two real-world data sets:

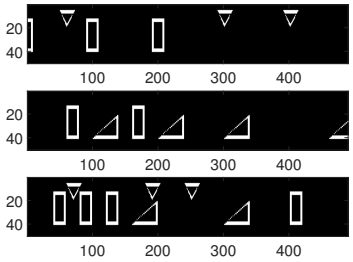
- **MLSP 2013[2] dataset:** it contains 645 recordings of 19 different bird species.
- **H. J. Andrews (HJA) dataset [15]:** it contains a total of 548 recordings with six different locations $PC1$, $PC4$, $PC7$, $PC8$, $PC13$, and $PC15$.

We convert each recording into a two-dimensional spectrogram with $F = 247$ and $T =$

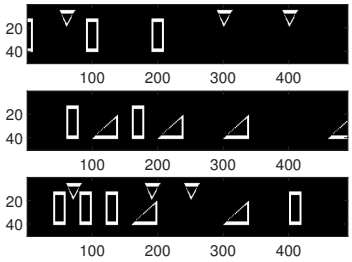
[2] `https://www.kaggle.com/c/mlsp-2013-birds`



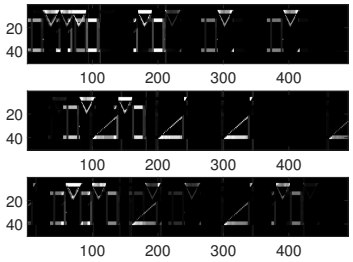
(a) Objective vs. Iteration.



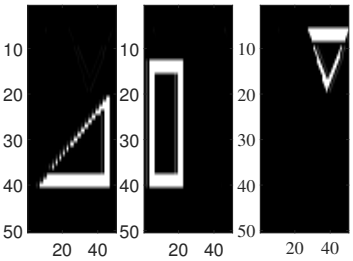
(b) Original spectrograms.



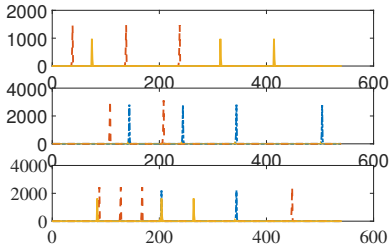
(c) Reconstruction by our approach.



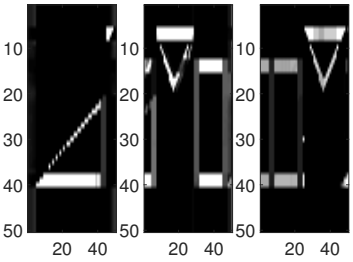
(d) Reconstruction by CNMF [107].



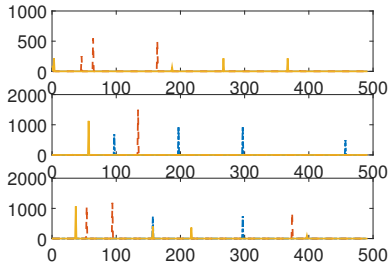
(e) Learned dictionary words by our approach.



(f) Learned activations by our approach.



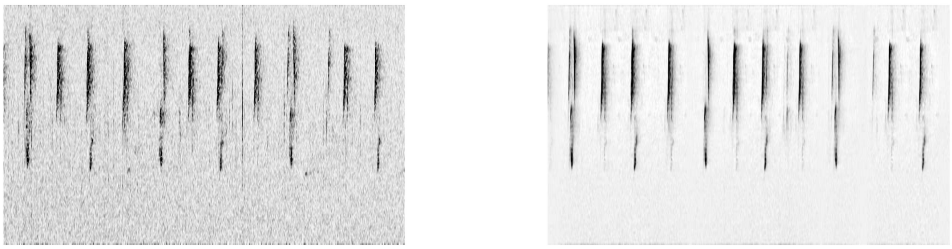
(g) Learned dictionary words by CNMF [107].



(h) Learned activations by CNMF [107].

Figure 2.3: Comparison between our approach and CNMF [107] (reproduction of [103]).

2497 and examine four aspects of the proposed approach: (i) spectrogram denoising (ii) optimal parameter selection, (iii) dictionary learning, and (iv) species classification.

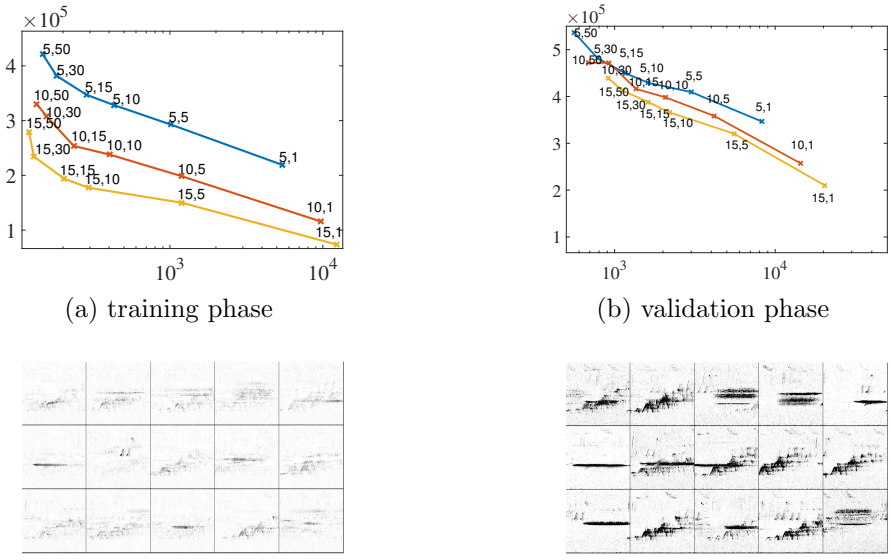


(a) Test spectrogram on $PC1$ with rain noise (b) Reconstructed test spectrogram on $PC1$

Figure 2.4: Examples of rain denoising on test spectrogram (reproduction of [103]).

Spectrogram denoising: We use the proposed dictionary learning approach for spectrogram denoising. To this end, we learn a dictionary from a clean set of recordings and use it for recovering a rain corrupted dataset. In this test, we use the HJA data set. The result in Fig. 2.4 shows that after running the dictionary learning algorithm, the rain artifact that appears as a long vertical line has been significantly reduced in the reconstructed spectrogram.

Parameter selection for dictionary learning The model parameters that affect the performance of dictionary learning are the number of dictionary words K and sparsity of the activations λ. To show the relationship between the model parameters and the dictionary learning performance, we present the reconstruction error $\sum_{i=1}^{N}\sum_{f=1}^{F}\sum_{t=1}^{T}(\mathbf{Y}^i(t,f)-\sum_{k=1}^{K}\mathbf{a}_k^i(t)*\mathbf{D}_k(t,f))^2$ against a practical approximation of the L_0 norm of the activations (number of the elements in $\mathbb{A}$ that are greater than $\epsilon = 10^{-2}$). During the training phase, we select 8 spectrograms from location $PC15$ of the HJA dataset and run the proposed algorithm to extract the dictionary words for each of the following parameter values $K = \{5, 10, 15\}$ and $\lambda = \{1, 5, 10, 15, 30, 50\}$. We apply the learned dictionary



(a) training phase

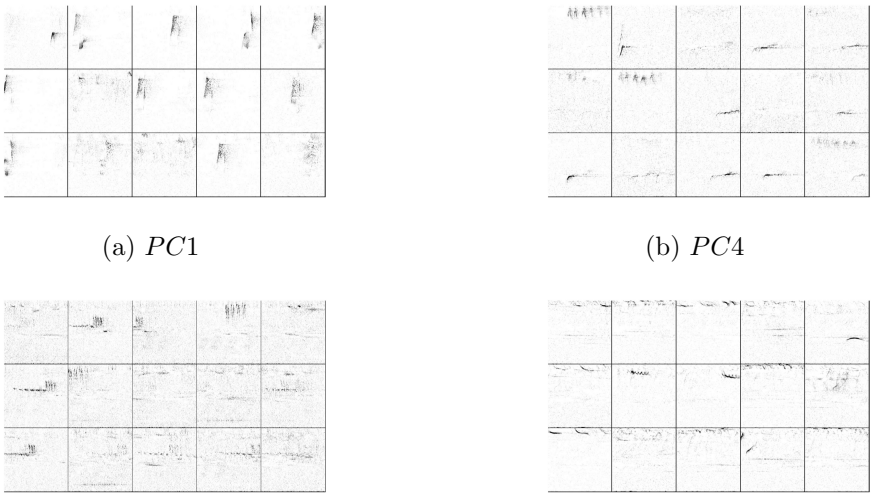
(b) validation phase

(c) learned dictionary

(d) learned dictionary

Figure 2.5: Parameter selection (reproduction of [103]): (a) training phase reconstruction error vs. L_0 norm of activations for $PC15$ (the first number for each point represents K and the second number for each point represents λ); (b) validation phase reconstruction error vs. L_0 norm of activations for $PC15$; (c) learned dictionary with $K = 15$ and $\lambda = 10$ for $PC15$; (d) learned dictionary with $K = 15$ and $\lambda = 50$ for $PC15$

words in the validation phase to independent three test spectrograms, the performance curves are shown in Fig. 2.5a (a) and (b). Results show that the reconstruction error decreases with decreasing value of λ and/or increasing the value of K. The L_0 norm of the activations increases with decreasing the value of λ. For a large λ, the dictionary concentrates on high energy words and low energy words are not discovered. For a small λ, the L_0 norm of activations increases significantly even though the reconstruction error decreases.



(a) $PC1$ (b) $PC4$

(c) $PC7$ (d) $PC13$

Figure 2.6: Learned dictionary words for HJA dataset (reproduction of [103]).

We select the optimal set of parameters ($\lambda = 10, K = 15$) to balance the reconstruction error and the sparseness of the activation in the validation set. We show the extracted dictionary words in the Fig. 2.6.

Extracted dictionary words on MLSP2013 dataset: We select four or five rich-of-syllable spectrograms from each species to learn the bird song dictionary and show the discovered dictionary words of all 19 species in Fig. 2.7 by using randomly projected

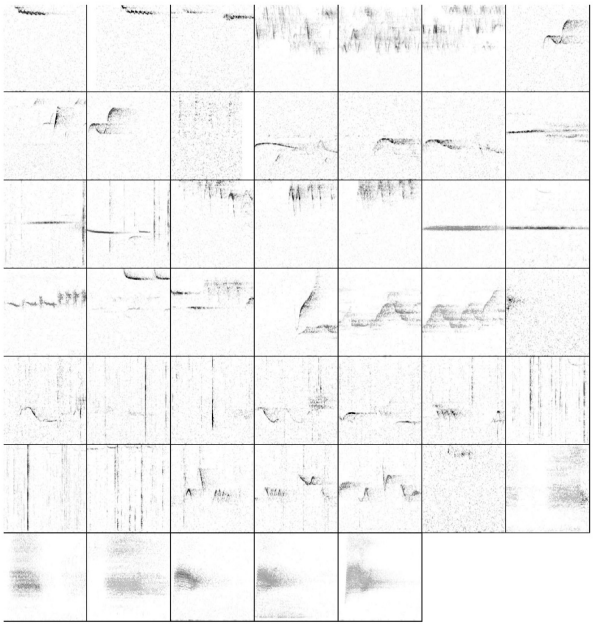


Figure 2.7: Learned bird dictionary words (reproduction of [103]).

dictionary learning with $r = 10\%F$ and setting $W = 200$ for all species.

2.4.3 Classification experiments

In order to test the discriminative information provided by the learned dictionary, we formulate the problem of bird species recognition in recordings of the HJA dataset. We randomly choose 36 recordings for training and 58 for testing (choosing at least ten per class in each case) and perform five two-class (one-against-all) experiments. Table 2.2 shows the classes (species) considered and the number of training and test recordings for each class. Five binary classification problems are considered where each single class is

Table 2.2: Number of training and test recordings selected of the HJA data set.

Index	Abbreviation (Class label)	Class name	# training recordings	# test recordings
1	BRCR	Brown Creeper	25	26
2	WIWR	Winter Wren	10	10
3	PSFL	Pacific-slope Flycatcher	16	19
4	CBCH	Chestnut-backed Chickadee	10	12
5	HAFL	Hammond's Flycatcher	10	10
-	-	Others	10	10

selected as target. Performance is evaluated by using the multi-label measures $\mathrm{F1}_{macro}$ and $\mathrm{F1}_{micro}$, as follows:

$$\mathrm{F1}_{macro} = \frac{1}{M}\sum_{j=1}^{M}\mathrm{F1}(TP_j, FP_j, TN_j, FN_j)$$

and

$$\mathrm{F1}_{micro} = \mathrm{F1}(\sum_{j=1}^{M}TP_j, \sum_{j=1}^{M}FP_j, \sum_{j=1}^{M}TN_j, \sum_{j=1}^{M}FN_j)$$

where $\mathrm{F1}(\cdot)$ stands for F1-score value, and TP_j, FP_j, TN_j and FN_j are the true positive, false positive, true negative and false negative values estimated when the target is class j.

Random projection is applied with $r = 12$. The following are the used parameters: $\lambda = 0.1$ (heuristically fixed), $K = 20$ (for supervised dictionary learning: $K_j = 3$ where $j \in \{1, 2, 3, 4, 5\}$ and, additionally, all spectrograms are labeled with a new class 6, for which $K_6 = 5$, in order to find common patterns), and, 10000 iterations for dictionary words and activations estimation. The parameter C, which controls the trade off between errors of the SVM and margin maximization, and δ are selected among $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$. The reported performance is the best

$F1_{micro}$ score obtained in each case.

Table 2.3 shows the results. The baseline is the classification framework where the features are directly extracted from spectrograms instead of activation signals, which implies that supervised dictionary learning is not applied. The kernels used are K_{HI} and K_{E_P} ($P = 1$ and 2; K_{E_1} and K_{E_2}, respectively).

Table 2.3: Classification results obtained with the proposed approach where features are extracted from activation signals and the baseline where features are directly extracted from spectrograms.

Features from\Kernel	K_{HI}		K_{E_1}		K_{E_2}	
	$F1_{\text{macro}}$	$F1_{\text{micro}}$	$F1_{\text{macro}}$	$F1_{\text{micro}}$	$F1_{\text{macro}}$	$F1_{\text{micro}}$
Activations	**0.7835**	0.7472	**0.7855**	0.**7746**	**0.8127**	**0.7800**
Spectrograms	0.7805	**0.7895**	0.71062	0.7724	0.7060	0.7639

According to our results, in the baseline exhibits higher $F1_{\text{micro}}$ scores than $F1_{\text{macro}}$ scores, which means that performances is not similar for all classes. On the other hand, since $F1_{\text{macro}}$ measure equally weighs all classes, we can say that our dictionary learning approach can find more complex relationships and increase the performance even when frequency band of vocalizations is not highly discriminant.

Chapter 3: Simple case study: supervised recurring signal pattern recognition and localization [1]

Due to rapid data growth we are facing nowadays, the capability to recognize recurring patterns in data becomes increasingly important because it helps to find regularities in data and can be used for downstream data analysis tasks such as feature extraction and classification. **The weakly-supervised dictionary learning problem can be simplified to recognizing and localizing a recurring signal pattern problem.** A common goal in this context is to discover recurrent patterns from data without any prior knowledge of what the patterns might look like. Toward this goal, several approaches have been proposed recently, most of which focused on finding the fundamental characteristics of the signal pattern [18, 62, 90, 111] and are generative in nature. By contrast, limited work has considered a discriminative approach for this task. One important issue with generative approaches of discovering recurring pattern is that the detection performance significantly degrades with increased noise and variations of the recurring signal. To address this issue, we focus on the problem of finding a discriminative convolutional kernel of the unknown recurring pattern, such that the resulting signal will directly indicate the location of the pattern. The problem of discovering convolutional kernel of recurring unknown pattern has been less studied.

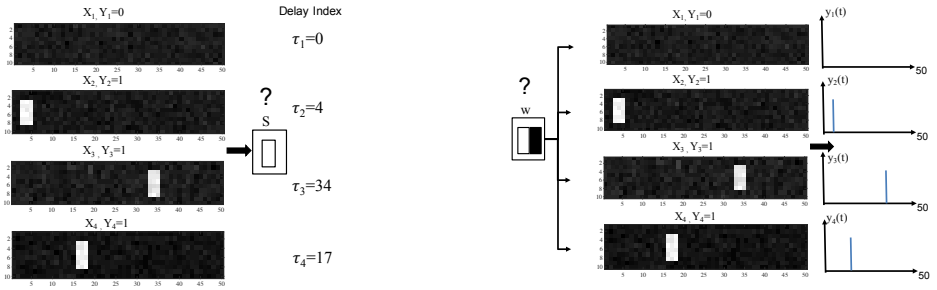
[1] This chapter is a joint work with Raviv Raich, Xiaoli Fern, and Jinsub Kim. This work was published as: Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Discriminative recurring signal detection and localization." In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2377-2381.

3.1 Single class case

We are given a collection of signals and their labels $\{(\mathbf{x}_1, Y_1), (\mathbf{x}_2, Y_2) \ldots, (\mathbf{x}_M, Y_M)\}$, where $\mathbf{x}_m$ denotes the mth signal $\mathbf{x}_m(t)$ for $1 \leq t \leq T_m$ and $Y_m \in \{0, 1\}$ denotes the presence or absence of an arbitrarily time-delayed pattern within signal $\mathbf{x}_m$. Our goal is to develop a discriminative framework for training a detector based on the given training data that can detect presence or absence of an unknown recurring pattern in a test signal. In contrast to the classical time delay estimation, we do not assume that the patterns within different signals are identical or identical up to a scaling factor. A generative model for detecting a recurring pattern [150] aims at finding the pattern and its corresponding delay as shown in Fig. 3.1(a). A discriminative approach [147] uses a convolution kernel to predict the presence and absence of that pattern as shown in Fig. 3.1(b). Unlike the generative approach, the discriminative kernel does not resemble the original shape of that recurring pattern, but transforms the original signal data into a new signal that matches up with the signal label. Here, we focus on the latter.

To predict the presence or absence of the common pattern, we consider a sliding window of size T_0 and treat the signal segment within each window as an instance. Specifically, we associate $\mathbf{x}_m(t)$, the mth signal at location t, with a corresponding sequence $y_{mt} \in \{0, 1\}$. The instance label y_{mt} being equal to 1 indicates the presence of a pattern at location t in $\mathbf{x}_m$. The sequence of instance labels for $\mathbf{x}_m$, which we denote by $\mathbf{y}_m \triangleq [y_{m1}, \ldots, y_{mT_m}]$, directly determines the bag-level label Y_m. Specifically, if $\mathbf{y}_m$ contains any entry with value 1, then Y_m is 1; otherwise, Y_m is zero.

The Probabilistic Model: In developing our model, we focus on a special case of the problem in which a single observance of the pattern of interest is made in each signal. Consequently, we assume that although the signal the instance label sequence



(a) Generative (b) Discriminative

Figure 3.1: Problem formulation of generative and discriminative recurring pattern recognition (reproduction of [147])

$\mathbf{y}_m$ are not observed, we have the information that $\mathbf{y}_m$ is either a vector with all zero entries or a vector with all zero entries except a single nonzero entry taking value 1. For completeness, we express the mth signal label Y_m in terms of the corresponding instance label sequence $\mathbf{y}_m$ as

$$Y_m = \begin{cases} 0, \ \mathbf{y}_m = \mathbf{0} \\ 1, \ \mathbf{y}_m \in \{\mathbf{e}_{m1}, \ldots, \mathbf{e}_{mT_m}\} \\ 2, \ \text{otherwise}, \end{cases}$$

where T_m is the number of sliding window segments in $\mathbf{x}_m$, and $\mathbf{e}_{ml} = [0, \ldots, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^{T_m}$, $\forall l = 1, \ldots, T_m$ is the lth standard basis vector of R^{T_m}, *i.e.*, its lth entry is one, and all other entries are zeros. Note that $Y_m = 2$ is only used for ensuring a complete probabilistic characterization of the model. However, in our setting, it is never observed.

We employ a probabilistic model (shown in Fig. 3.2) with a logistic function to model the conditional distribution of an instance label y_{mt} given a realization of the corresponding sliding window segment $\mathbf{x}_{mt} = [\mathbf{x}_m(t), \mathbf{x}_m(t-1), \ldots, \mathbf{x}_m(t-T_0+1)]^T \in \mathbb{R}^{T_0}$ as the tth windowed instance and $\mathbf{w} = [\mathbf{w}(1), \ldots, \mathbf{w}(T_0)]^T \in \mathbb{R}^{T_0}$ as the kernel signal. Therefore, the probabilistic model for y_{mt} is given by:

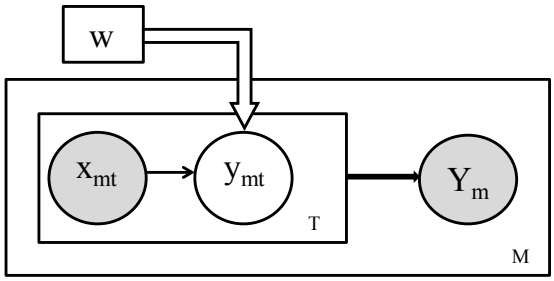


Figure 3.2: The probabilistic graphical model (reproduction of [147]).

$$P(y_{mt}|\mathbf{x}_{mt};\mathbf{w}) = \frac{e^{\mathbf{w}^T\mathbf{x}_{mt}y_{mt}}}{1+e^{\mathbf{w}^T\mathbf{x}_{mt}}}. \tag{3.1}$$

Note that $\mathbf{w}^T\mathbf{x}_{mt}$ for all $t = 1,\ldots,T_m$ and $m = 1,\ldots,M$ are implemented as a convolution such that $\mathbf{w}^T\mathbf{x}_{mt} = \sum_{\tau=0}^{T_0}\mathbf{x}_m(t-\tau)\mathbf{w}(\tau)$.

To model the mth signal label Y_m given the instance labels $\mathbf{y}_m$, we consider two cases. When the signal label is positive $Y_m = 1$, only one out of T_m instance label can be one and the others are zeros. When the signal label is negative $Y_m = 0$, all of the T_m instances must be zeros. Therefore, the probabilistic model for the signal label Y_m given the instance labels $\mathbf{y}_m$ is:

$$P(Y_m|\mathbf{y}_m) = [\sum_{l=1}^{T_m}\mathbb{I}(\mathbf{y}_m = \mathbf{e}_{ml})]^{Y_m}[\mathbb{I}(\mathbf{y}_m = \mathbf{0})]^{1-Y_m}, \tag{3.2}$$

The probabilistic graphical model in Fig. 3.2 describes the conditional dependence structure of our model.

Extension to 2-D signals: When the data signal is 2-D such as spectrogram *i.e.*, $\mathbf{x}_m \in \mathbb{R}^{F\times T}$ for some frequency F, the probabilistic model in (3.1) can be smoothly adopted by setting the convolutive kernel to be 2-D as well, *i.e.*, $\mathbf{w} \in \mathbb{R}^{F\times T_0}$. In this case, $\mathbf{w}^T\mathbf{x}_{mt}$ is replaced with $\text{trace}(\mathbf{w}^T\mathbf{x}_{mt}) = \sum_{f=1}^{F}\sum_{\tau=0}^{T_0}\mathbf{x}_m(f,t-\tau)\mathbf{w}(f,\tau)$.

Maximum Likelihood Estimation: Given our proposed model, we consider estimating the model parameter $\mathbf{w}$ using maximum likelihood estimation (MLE).

Data Likelihood: Denote $\mathbf{D} = \{(\mathbf{x}_1, Y_1), (\mathbf{x}_2, Y_2) \ldots, (\mathbf{x}_M, Y_M)\}$ as the observed data and assume that $Y_m \in \{0, 1\}$, the data likelihood $L(\mathbf{w}) = P(\mathbf{D}; \mathbf{w})$, is obtained as

$$L(\mathbf{w}) = \prod_{m=1}^{M} \frac{(\sum_{l=1}^{T_m} e^{\mathbf{w}^T \mathbf{x}_{ml}})^{Y_m}}{\prod_{t=1}^{T_m}(1 + e^{\mathbf{w}^T \mathbf{x}_{mt}})} P(\mathbf{x}_m). \tag{3.3}$$

Therefore, the negative log-likelihood function is:

$$f(\mathbf{w}) = \sum_{m=1}^{M} [\sum_{t=1}^{T_m} \log(1 + e^{\mathbf{w}^T \mathbf{x}_{mt}}) - Y_m \log(\sum_{t=1}^{T_m} e^{\mathbf{w}^T \mathbf{x}_{mt}})] + C,$$

where $C = \sum_{m=1}^{M} \log(P(\mathbf{x}_m))$ is a constant. The challenge is this function is a combination of convex and concave function such that the non-convexity of the problem makes it harder to minimize.

Solution with CCCP: Since the objective is a convex-concave function, we apply the convex-concave procedure (CCCP) [151] to update $\mathbf{w}$. The general idea is to construct a majorizing function $g(\mathbf{w}, \mathbf{w}^i)$ such that (i) $g(\mathbf{w}, \mathbf{w}^i) \geq f(\mathbf{w})$ for any $\mathbf{w}, \mathbf{w}^i$; and (ii) $g(\mathbf{w}, \mathbf{w}^i) = f(\mathbf{w})$ for $\mathbf{w} = \mathbf{w}^i$. Minimizing $g(\mathbf{w}, \mathbf{w}^i)$ function instead of $f(\mathbf{w})$ results in the following update rule $\mathbf{w}^{(i+1)} = \arg\min_{\mathbf{w}} g(\mathbf{w}, \mathbf{w}^i)$, which yields non increasing sequence of the objective, i.e., $f(\mathbf{w}^{(i+1)}) \leq f(\mathbf{w}^i)$.

A simple upper bound function $g(\mathbf{w}, \mathbf{w}^i)$ can be obtained by linearizing the convex function $v(\mathbf{w}) = \log(\sum_{t=1}^{T} e^{\mathbf{w}^T \mathbf{x}_{mt}})$. Since $v(\mathbf{w}) \geq v(\mathbf{w}^i) + (\mathbf{w} - \mathbf{w}^i)^T \Delta v(\mathbf{w}^i)$, then

$f(\mathbf{w}) \leq g(\mathbf{w}, \mathbf{w}^i)$ [56]. Therefore, the upper bound $g(\mathbf{w}, \mathbf{w}^i)$ is:

$$g(\mathbf{w}, \mathbf{w}^i) = \sum_{m=1}^{M} [\sum_{t=1}^{T_m} \log(1 + e^{\mathbf{w}^T \mathbf{x}_{mt}}) - Y_m [\log(\sum_{t=1}^{T_m} e^{\mathbf{w}^{iT} \mathbf{x}_{mt}}) + \Big(\frac{\sum_{t=1}^{T_m} e^{\mathbf{w}^{iT} \mathbf{x}_{mt}} \mathbf{x}_{mt}}{\sum_{t=1}^{T_m} e^{\mathbf{w}^{iT} \mathbf{x}_{mt}}} \Big)^T (\mathbf{w} - \mathbf{w}^i)].$$

Using the gradient descent method, we obtain the update rule as follows:

$$\mathbf{w}^{i+1} = \mathbf{w}^i + \gamma \frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}} \mid_{\mathbf{w}=\mathbf{w}^i}, \text{where,} \tag{3.4}$$
$$\frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}} \mid_{\mathbf{w}=\mathbf{w}^i} = \sum_{m=1}^{M} \sum_{t=1}^{T_m} [P(y_{mt}) - Y_m P(y_{mt}|Y_m)] \mathbf{x}_{mt},$$

and γ is a learning rate. We refer to $P(y_{mt}) = P(y_{mt} = 1|\mathbf{x}_{mt}; \mathbf{w}^i)$ in (3.1) as a prior probability and $P(y_{mt}|Y) = P(y_{mt} = 1|Y, \mathbf{x}; \mathbf{w}^i) = \frac{e^{\mathbf{w}^{iT} \mathbf{x}_{mt}}}{\sum_{t=1}^{T_m} e^{\mathbf{w}^{iT} \mathbf{x}_{mt}}}$ as a posterior probability, which can also be directly computed using Bayes rule.

Prediction: Given a test signal $\mathbf{x}^{test}$, the localization signal or instance label signal $\hat{y}_t^{test}$ is obtained by

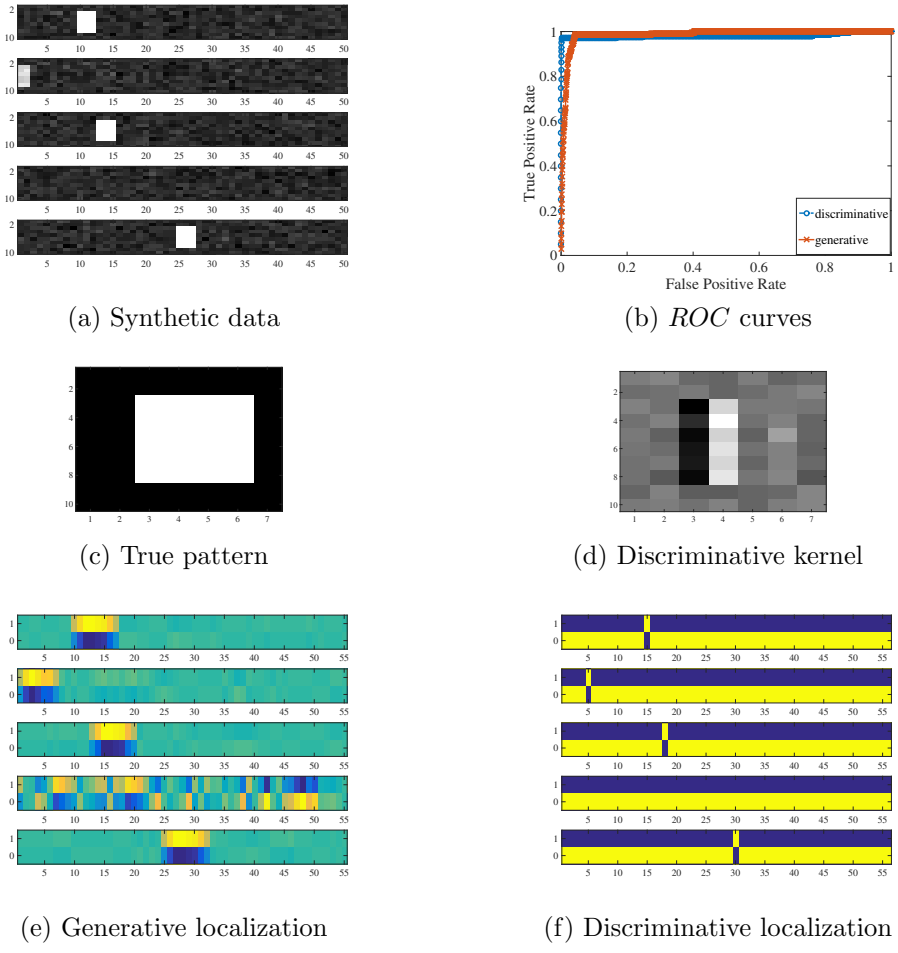
$$\hat{y}_t^{test} = \arg \max_{a \in \{0,1\}} P(y_t = a|\mathbf{x}^{test}, \mathbf{w}) \; \forall \; t = 1, \ldots T.$$

A signal level label is obtained by

$$\hat{Y}^{test} = \cup_{t=1}^{T} \hat{y}_t^{test}.$$

Computational complexity: To simplify the computational complexity analysis, assume that the number of instance per signal T_m are all the same and equal to T. The overall computational complexity is $\mathcal{O}(NMTT_0)$, where N is the total number of itera-

tion needed for updating the kernel $\mathbf{w}$. If T_0 is set to be large ($T_0 \approx T$), we can apply Fast Fourier Transform (FFT) and Inverse of FFT to speed up the convolution [84] such that the computational complexity will become $\mathcal{O}(NMT \log T)$.



(a) Synthetic data (b) *ROC* curves

(c) True pattern (d) Discriminative kernel

(e) Generative localization (f) Discriminative localization

Figure 3.3: Synthetic data results (reproduction of [147]).

Numerical Evaluation: In order to evaluate our discriminative pattern recognition approach, we perform a numerical synthetic experiment.

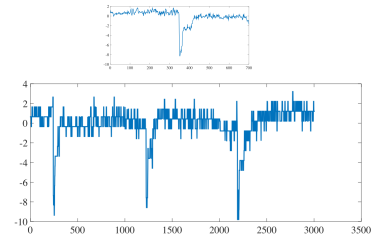
Synthetic data generation: The synthetic 2-D signals are generated with height

(number of frequency bins) $F = 10$ and width (time frames) $T = 50$ by randomly placing a rectangular shape into one of the $T - 6$ maximally overlapped 10×7 windows of the signals. Each window is referred as an instance and is labeled as 1 if the rectangular shape is within that window, otherwise, it will be labeled as 0, the negative class. See Fig.3.3 (a) for an example.

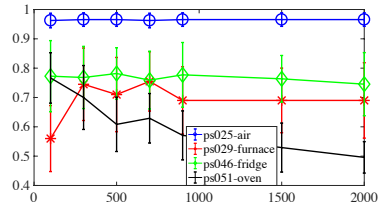
Numerical results: To verify our proposed approach, we use 10 independent random shuffles of 200 signals with balanced label that are split into 160 training signals and 40 test signals. The convolution kernel dimensions are set to $F = 10$ and $T_0 = 7$. Fig. 3.3(c) shows the original rectangular shape, while Fig. 3.3(d) shows the learned kernel, which appears to approximate the gradient of the rectangular shape. Fig. 3.3(f) verifies that the position where the rectangular signal lies is correctly predicted, however, using the original signal pattern in the generative framework yields some ambiguity about the shape location (see Fig. 3.3(e)). To show the resulting detection performance, we plot the receiver operating characteristic (ROC) curve [13]. The averaged test ROC curves based on the 10 different training sets are shown in Fig. 3.3(b). We can see that using a discriminative kernel produces higher true positive rate when the threshold is low.

Real-world Experiment: In this experiment, our goal is to learn a discriminative activation signature for each appliance using a set of training data and to test the detection performance on a separate test data set.

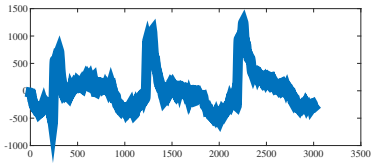
Data Set and preprocessing: We use the Pecan Street dataset (*Source: Pecan Street Research Institute*), which contains four homes of disaggregated, time-sampled electricity usage data. The data set includes both voltage and apparent power readings in a period of 25 days. For the experimental setup, we split the four home data into training data with extracted activation signature with 1000 samples of a period of 11/17/2012-



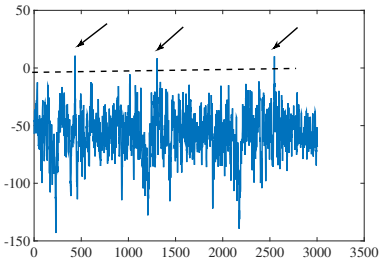
(a) fridge activation and signal



(b) Tuning T_0



(c) Generative detector



(d) Discriminative detector

Figure 3.4: Detection comparison between generative and discriminative fridge activation patterns (reproduction of [147]).

11/25/2012 and test data is one hour readings, which contains around 500, 000 samples, with a period of 11/26/2012-12/11/2012. For each home and each appliance, the training activation event of short sequence voltage responses are generated based on the ground truth of a power increase from 0 to 80 watt or more on the independent measurement from a commercial power meter. The negative labeled data of short sequence voltage responses are randomly extracted based on non-increase of the power meter.

Due to a time varying DC offset on voltage peak to peak (V_{pp}) value, we consider a moving window (each window contains 1000 samples) approach to calculate the average DC offset signal. For both training and test data, we remove that DC offset. We also apply a five-tap median filter to despike the voltage waveforms, since the voltage peak to peak (V_{pp}) waveform is corrupted by spike noise. For home ps-029, we use a fifteen-tap median filter.

Results and Analysis: In the training phase, we tune the window size T_0 using ten random shuffles of the data. On each of the ten, we first shuffle and then pick the first 80% of the data for training and the remaining 20% for validation. For each random shuffle, we compute the signal label accuracy $1/M \sum_{m=1}^{M} I(\hat{Y}_m^{val} = Y_m^{val})$ for $T_0 \in \{100, 300, 500, 700, 900, 1500, 2000\}$ and present mean and standard deviation (over the ten shuffles) as in Fig. 3.4(b). An iterative gradient descent method is used to find the discriminative activation signature. To compare with the results obtained from a generative approach [150], we show a detection example from the generative approach and the discriminative approach in the Fig. 3.4. Since [150] uses T_0=700, we show the resulting AUCs comparison with our approach by using both T_0=700 and best window size in the Table 3.1.

Fig. 3.4(c) shows an example for the detector output (before applying the threshold) for the generative detector. We observe that the peak level of the discriminative

detector output in Fig. 3.4(d) appears more consistent than that of the generative detector. In general, we observe that the discriminative approach presents higher detection performance than the generative approach, especially for some of the appliances which contains more variation in their template. For example, for oven in home ps-025 and Fridge in ps-046, the detection AUCs of the discriminative approach are 0.86 and 0.87 respectively, which are significantly higher than the generative approach AUCs, 0.52 and 0.49 respectively.

House ID	App. Name	Gen.(T_0=700)	Disc.(T_0=700)	Disc.(best T_0)
PS-025	Air-Cond.	0.95	**0.97**	**0.97** T_0=700
PS-025	Oven	0.52	0.80	**0.86** T_0=100
PS-029	Air-Cond.	0.92	**0.99**	**0.99** T_0=700
PS-029	Dryer	**0.99**	0.93	0.96 T_0=100
PS-029	Fridge	0.72	0.85	**0.86** T_0=900
PS-029	Furnace	0.86	**0.89**	**0.89** T_0=700
PS-029	Microwave	0.88	**0.94**	**0.94** T_0=700
PS-029	Oven	**0.91**	0.78	0.88 T_0=100
PS-046	Air-Cond.	0.85	0.93	**0.97** T_0=500
PS-046	Fridge	0.49	0.85	**0.87** T_0=900
PS-046	Furnace	0.54	**0.56**	**0.56** T_0=700
PS-046	Oven	**0.92**	0.76	0.88 T_0=100
PS-051	Air-Cond.	0.91	**0.97**	**0.97** T_0=700
PS-051	Oven	**0.78**	0.61	0.72 T_0=100

Table 3.1: AUC for the generative method [150] and for our discriminative method.

Discussion on computational complexity: In a generative approach, [150] proposes an algorithm of computational complexity of $\mathcal{O}(T_0(MT)^2)$. In our discriminative approach, the computational complexity is $\mathcal{O}(NMTT_0)$. If the total number of iteration N is set to be less than MT, our discriminative approach is more efficient.

3.2 Multiple class case as an extension

Given the observed signals and the associated labels $\{(\mathbf{x}_1, Y_1), (\mathbf{x}_2, Y_2) \ldots, (\mathbf{x}_M, Y_M)\}$, where $\mathbf{x}_m$ denotes the mth signal $\mathbf{x}_m(t)$ for $0 \leq t \leq T_m$ and $Y_m \in \{0, 1, 2, \ldots, C\}$ denotes a particular class c type of time-delayed pattern is shown within signal $\mathbf{x}_m$, the goal is to develop a discriminative framework for training a detector based on the given training data that can detect an unknown recurring pattern belongs to which class in a test signal.

To predict the multiple recurring patterns, we consider a sliding window of size T_w and treat the signal segment within each window as an instance. We denote $\Delta = (T_w - 1)/2$ by assuming window size is odd for simplicity. Specifically, we associate $\mathbf{x}_m(t)$, the mth signal at location t, with a corresponding sequence $\mathbf{y}_m(t) \in \{0, c\}$ for $c = 1, 2, \ldots C$. The instance label y_{mt} being equal to c indicates the presence of a pattern c at location t in $\mathbf{x}_m$. The sequence of instance-level labels for $\mathbf{x}_m$, which we denote by $\mathbf{y}_m \triangleq [y_{m1}, \ldots, y_{mT_m}]$, directly determines the bag-level label Y_m. Specifically, if $\mathbf{y}_m$ contains any entry with value c, then Y_m is c; otherwise, Y_m is zero.

The Probabilistic Model: In developing our model, we focus on a special case of the problem in which a single observance of the pattern of interest is made in each signal. Consequently, we assume that although the signal time-instance label sequence $\mathbf{y}_m$ are not observed, we have the information that $\mathbf{y}_m$ is either a vector with all zero entries or a vector with all zero entries except a single nonzero entry taking value 1. For completeness, we express the mth signal label Y_m in terms of the corresponding time-instance label sequence $\mathbf{y}_m$ as

$$Y_m = \begin{cases} 0, \ \mathbf{y}_m = \mathbf{0} \\ c, \ \mathbf{y}_m \in \{\mathbf{e}^c_{m(-\Delta+1)}, \ldots, \mathbf{e}^c_{m(T_m+\Delta)}\}; c = 1, 2, \ldots, C \\ C+1, \ \text{otherwise}, \end{cases}$$

where T_m is the number of samples in the mth signal and there is total of $T_m + T_w + 1$ sliding window segments in $\mathbf{x}_m$, and $\mathbf{e}^c_{ml} = [0, \ldots, 0, c, 0, \ldots, 0]^T \in \mathbb{R}^{T_m+T_w-1}$, $\forall l = -\Delta+1, \ldots, T_m+\Delta$ is the lth standard basis vector of $R^{T_m+T_w-1}$, *i.e.*, its lth entry is class c, and all other entries are zeros. Note that $Y_m = C+1$ is only used for ensuring a complete probabilistic characterization of the model. However, in our setting, it is never observed.

To model the mth signal label Y_m given the instance labels $\mathbf{y}_m$, we consider two cases. When the signal label is positive $Y_m = 1$, only one out of T_m time instance label can be one and the others are zeros. When the signal label is negative $Y_m = 0$, all of the T_m time instances must be zeros. Therefore, the probabilistic model for the signal label Y_m given the instance labels $\mathbf{y}_m$ is:

$$P(Y_m|\mathbf{y}_m) = \prod_{c=1}^{C} [\sum_{l=-\Delta+1}^{T_m+\Delta} \mathbb{I}(\mathbf{y}_m = e^c_{ml})]^{\mathbb{I}(Y_m=c)} [\mathbb{I}(\mathbf{y}_m = \mathbf{0})]^{\mathbb{I}(Y_m=\mathbf{0})}, \tag{3.5}$$

Maximum Likelihood Estimation: Given our proposed model, we consider estimating the model parameter $\mathbf{w}$ using maximum likelihood estimation (MLE).

Data Likelihood: Denote $\mathbf{D} = \{(\mathbf{x}_1, Y_1), (\mathbf{x}_2, Y_2) \ldots, (\mathbf{x}_M, Y_M)\}$ as the observed

data and assume that $Y_m \in \{0, 1\}$, the data likelihood $L(\mathbf{w}) = P(\mathbf{D}; \mathbf{w})$, is obtained as

$$L(\mathbf{w}) = \prod_{m=1}^{M} \frac{\prod_{c=1}^{C} (\sum_{l=-\Delta+1}^{T_m+\Delta} e^{\mathbf{w}_c^T \mathbf{x}_{ml} - \mathbf{w}_0^T \mathbf{x}_{ml}})^{\mathbb{I}(Y_m=c)}}{\prod_{t=-\Delta+1}^{T_m+\Delta} (1 + \sum_{u=1}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{mt} - \mathbf{w}_0^T \mathbf{x}_{mt}})} P(\mathbf{x}_m). \tag{3.6}$$

Therefore, the negative log-likelihood function is:

$$\begin{aligned} f(\mathbf{w}) = & \sum_{m=1}^{M} [\sum_{t=-\Delta+1}^{T_m+\Delta} \log(1 + \sum_{u=1}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{mt} - \mathbf{w}_0^T \mathbf{x}_{mt}}) - \\ & \sum_{c=1}^{C} \mathbb{I}(Y_m = c) \log(\sum_{t=-\Delta+1}^{T_m+\Delta} e^{\mathbf{w}_c^T \mathbf{x}_{mt} - \mathbf{w}_0^T \mathbf{x}_{mt}})]. \end{aligned} \tag{3.7}$$

The challenge is this function is a combination of convex and concave function such that the non-convexity of the problem makes it harder to minimize.

Solution with CCCP: Since the objective is a convex-concave function, we apply the convex-concave procedure (CCCP) [151] to update $\mathbf{w}$. A simple upper bound function $g(\mathbf{w}, \mathbf{w}^i)$ can be obtained by linearizing the convex function $v(\mathbf{w}) = \sum_{m=1}^{M} \sum_{c=1}^{C} \mathbb{I}(Y_m = c) \log(\sum_{t=1}^{T_m} e^{\mathbf{w}_c^T \mathbf{x}_{mt} - \mathbf{w}_0^T \mathbf{x}_{mt}})$. Since $v(\mathbf{w}) \geq v(\mathbf{w}^i) + (\mathbf{w} - \mathbf{w}^i)^T \Delta v(\mathbf{w}^i)$, then $f(\mathbf{w}) \leq$

$g(\mathbf{w}, \mathbf{w}^i)$ [56]. Therefore, the upper bound $g(\mathbf{w}, \mathbf{w}^i)$ is:

$$
\begin{aligned}
g(\mathbf{w}, \mathbf{w}^i) = & \sum_{m=1}^{M} [\sum_{t=-\Delta+1}^{T_m+\Delta} \log(1 + \sum_{u=1}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{mt} - \mathbf{w}_0^T \mathbf{x}_{mt}}) - \\
& \sum_{c=1}^{C} \mathbb{I}(Y_m = c) \Big(\log(\sum_{t=-\Delta+1}^{T_m+\Delta} e^{\mathbf{w}_c^{iT} \mathbf{x}_{mt} - \mathbf{w}_0^{iT} \mathbf{x}_{mt}}) + \\
& (\mathbf{w}_c - \mathbf{w}_c^i - \mathbf{w}_0 + \mathbf{w}_0^i)^T \big(\frac{\sum_{t=-\Delta+1}^{T_m+\Delta} e^{\mathbf{w}_c^{iT} \mathbf{x}_{mt} - \mathbf{w}_0^{iT} \mathbf{x}_{mt}} \mathbf{x}_{mt}}{\sum_{t=-\Delta+1}^{T_m+\Delta} e^{\mathbf{w}_c^{iT} \mathbf{x}_{mt} - \mathbf{w}_0^{iT} \mathbf{x}_{mt}}} \big) \Big)].
\end{aligned}
$$

Using the gradient descent method, we obtain the update rule as follows:

$$
\begin{aligned}
\mathbf{w}_0^{i+1} &= \mathbf{w}_0^i + \gamma \frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}_0} \mid_{\mathbf{w}_0 = \mathbf{w}_0^i}, \text{where,} \qquad (3.8) \\
\frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}_0} \mid_{\mathbf{w}_0 = \mathbf{w}_0^i} &= \sum_{m=1}^{M} \sum_{t=-\Delta+1}^{T_m+\Delta} [P(y_{mt} = 0) \\
& - \quad (1 - \sum_{c=1}^{C} \mathbb{I}(Y_m = c) P(y_{mt} | Y_m))] \mathbf{x}_{mt},
\end{aligned}
$$

and

$$
\begin{aligned}
\mathbf{w}_c^{i+1} &= \mathbf{w}_c^i + \gamma \frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}_c} \mid_{\mathbf{w}_c = \mathbf{w}_c^i}, \text{where,} \qquad (3.9) \\
\frac{\partial g(\mathbf{w}, \mathbf{w}^i)}{\partial \mathbf{w}_c} \mid_{\mathbf{w}_c = \mathbf{w}_c^i} &= \sum_{m=1}^{M} \sum_{t=-\Delta+1}^{T_m+\Delta} [P(y_{mt}) - \mathbb{I}(Y_m = c) P(y_{mt} | Y_m)] \mathbf{x}_{mt}.
\end{aligned}
$$

for $c = 1, 2, \ldots, C$, and γ is a learning rate. We refer $P(y_{mt}) = P(y_{mt} = c | \mathbf{x}_{mt}; \mathbf{w}^i)$ as a prior probability and refer $P(y_{mt}|Y) = P(y_{mt} = c | Y, \mathbf{x}; \mathbf{w}^i) = \frac{e^{\mathbf{w}_c^{iT} \mathbf{x}_{mt} - \mathbf{w}_0^{iT} \mathbf{x}_{mt}}}{\sum_{t=-\Delta+1}^{T_m+\Delta+1} e^{\mathbf{w}_c^{iT} \mathbf{x}_{mt} - \mathbf{w}_0^{iT} \mathbf{x}_{mt}}}$ as a posterior probability.

Chapter 4: Weakly-supervised dictionary learning for time-series [1]

When both data and global labels are observed, the conventional dictionary learning approaches are not suited. Instead, we present a convolutive analysis dictionary learning under weak supervision,where we learn a dictionary given a set of signals andtheir label sets. In analysis dictionary learning, we are given a collection of data vectors $\{\mathbf{x}_1, \ldots, \mathbf{x}_B\}$ and look for an analysis dictionary $[\mathbf{w}_1, \ldots, \mathbf{w}_C]$ such that for each n the analysis of $\mathbf{x}_n$ given by $[\mathbf{w}_0^T\mathbf{x}_n, \ldots, \mathbf{w}_C^T\mathbf{x}_n]$ is sparse [102]. In the convolutive setting, we are given a set of signals $\{x_1(t), \ldots, x_N(t)\}$ and look for an analysis dictionary of the form $[w_1(t), \ldots, w_C(t)]$ such that for each n the analysis of signal $x_n(t)$ given by $[w_0(t) * x_n(t), \ldots, w_C(t) * x_n(t)]$ is a sparse signal, where $*$ is a convolution operator [92, 93]. Based on the convolutive version of analysis dictionary learning, we develop a discriminative version of the convolutive analysis dictionary learning is. In particular, we consider the weak supervision setting in which every signal $x_n(t)$ is accompanied with a single label set Y_b. Focusing on time-series analysis, we proceed with the proposed formulation of this problem using a probabilistic model approach.

4.1 Problem statement

In the following of the whole chapter, 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,

[1] This chapter is a joint work with Dr. Raviv Raich, Xiaoli Fern and Jinsub Kim. This work was published as: Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly supervised dictionary learning." *IEEE Transactions on Signal Processing* 66, no. 10 (2018): 2527-2541.

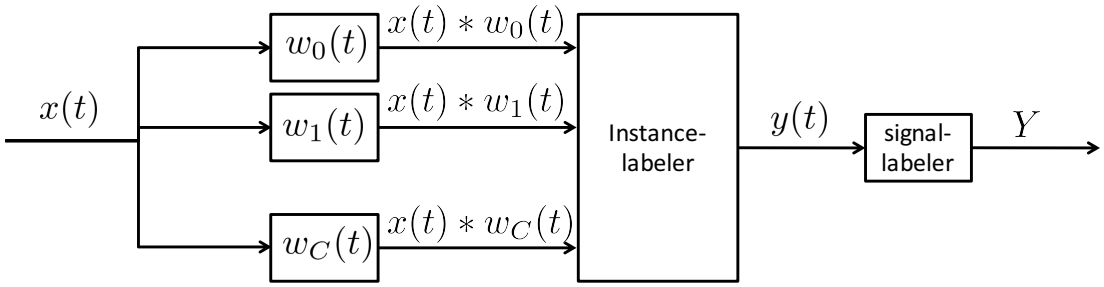


Figure 4.1: An illustration of the setting of weakly supervised analysis dictionary learning (reproduction of [148]).

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 $|_t$ 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 $w * x|_t = \sum_{u=-\infty}^{\infty} x(t-u)w(u)$. Common notations used in this paper are given in Table 4.1.

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 [92, 93]. 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.

Table 4.1: List of notations

Notation	Explanation
x_n	the nth signal, e.g., $x_n(t)$ for 1-D signal indexed by time t and $x_n(f,t)$ for 2-D signal indexed by frequency f and time t
F	the number of frequency bins, $F = 1$ in 1-D signal case
T_n	number of samples in nth signal
w_c	the cth dictionary signal, e.g., $w_c(t)$ for 1-D signal and $w_c(f,t)$ for 2-D signal
$\mathbf{w}_c$	is the vector format of cth analysis dictionary word
b_c	is the bias term for the cth analysis dictionary word
$\mathbf{w}$	$[\mathbf{w}_0^T, \mathbf{w}_1^T, \ldots, \mathbf{w}_C^T]^T$, C dictionary words
$\mathbf{b}$	$[b_0, b_1, \ldots, b_C]^T$, a vector of C bias terms
C	number of class or dictionary words
T_w	number of samples for dictionary
$\mathbf{x}_{nt}$	is a tth time instance in nth signal
$\mathcal{X}$	$\{x_n\}_{n=1}^N$, set of N signals
$\mathcal{Y}$	$\{Y_n\}_{n=1}^N$, N sets of label set
$y_n(t)$	is the instance label at time index t in the nth signal
$\bar{N}_n$	maximum number of non-novel instances in nth signal

In the *weak supervision setting* (see Fig. 4.1), y_n is unknown and we are interested in learning a discriminative version of the convolutive 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 $\mathcal{D}$; 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 $\mathcal{D}$. We proceed with the proposed formulation of this problem using a

probabilistic model approach.

4.2 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 $w_c * x_n \mid_t$ with $\mathbf{w}_c^T \mathbf{x}_{nt}$ for $t = -\Delta, -\Delta + 1, \ldots, T_n - 1 + \Delta$ such that

$$x_n * w_c \mid_t = \sum_{\tau=-\Delta}^{\Delta} x_n(t-\tau) w_c(\tau) = \mathbf{x}_{nt}^T \mathbf{w}_c,$$

where $\mathbf{x}_{nt} \in \mathbb{R}^{T_w}$ is defined as

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

and $\mathbf{w}_c \in \mathbb{R}^{T_w}$ is given by

$$\mathbf{w}_c = [w_c(-\Delta), w_c(-\Delta+1), \ldots, w_c(\Delta)]^T.$$

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

$$x_n * w_c \mid_t = \sum_{f=1}^{F} \sum_{\tau=-\Delta}^{\Delta} x_n(f, t-\tau) w_c(f,\tau) = \mathbf{x}_{nt}^T \mathbf{w}_c,$$

where each windowed portion of the signal is

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

and

$$\mathbf{w}_c = [w_c(1, -\Delta), w_c(1, -\Delta+1), \ldots, w_c(F, \Delta)]^T.$$

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.

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 Convolutive 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 $[w_0 * x_n \mid_t, w_1 * x_n \mid_t, \ldots, w_C * x_n \mid_t]$, the analysis result evaluated at time t:

$$P(y_n(t) = c|x_n) = f_c(w_0 * x_n \mid_t, \ldots, w_C * x_n \mid_t)$$

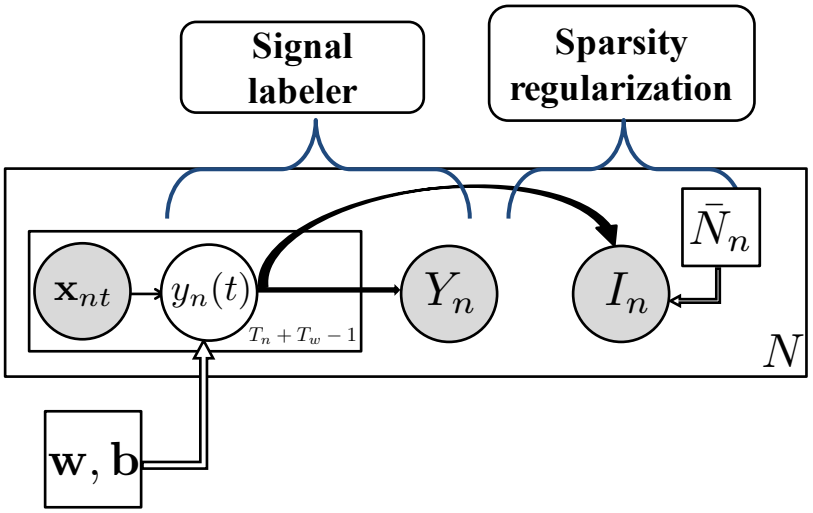


Figure 4.2: The proposed graphical model for WSCADL (reproduction of [148]).

for $c = 0, 1, \ldots, C$, where f_c is an arbitrary multivariate function such that $f_c : \mathbb{R}^{C+1} \to [0, 1]$ 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 $\bar{N}_n$:

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

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

$$Y_n = \bigcup_{\substack{t=-\Delta, \\ y_n(t)\neq 0}}^{T_n-1+\Delta} \{y_n(t)\}.$$

Note that this assumption makes it is 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.

Model description: The probabilistic graphical model for the weakly-supervised convolutive analysis dictionary learning (WSCADL) is shown in Figure 4.2, in which, our target is to estimate the model parameters $\mathbf{w} = [\mathbf{w}_0^T, \mathbf{w}_1^T, \ldots, \mathbf{w}_C^T]^T$ and $\mathbf{b} = [b_0, b_1, \ldots, b_C]^T$. 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 $w_c * x_n$ for $c = 0, 1, \ldots, C$ evaluated at time t and hence through signal window $\mathbf{x}_{nt}$. The probabilistic model for $y_n(t)$ follows the multinomial logistic regression model given by:

$$P(y_n(t) = c|x_n; \mathbf{w}, \mathbf{b}) = \frac{e^{\mathbf{w}_c^T \mathbf{x}_{nt} + b_c}}{\sum_{u=0}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{nt} + b_u}}, \tag{4.1}$$

for $c = 0, 1, \ldots, C$, where $\mathbf{w}_c$ is the cth 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 [95]. To integrate a constraint on the number of non-zero instances in the nth signal (i.e., the sparsity of $y_n(t)$) into our probability model, we introduce the latent random variable I_n, an indicator that takes the value 1 if the number of nonzero $y_n(t)$ is less than or equal a sparsity upper bound $\bar{N}_n$ and zero otherwise. We treat $\bar{N}_n$ as a tuning (or hyper-) parameter of the graphical model. The smaller the value of $\bar{N}_n$, the sparser the label signal $y_n(t)$ is. The probability model for sparsity indicator I_n of label signal y_n is given by

$$P(I_n = 1|y_n; \bar{N}_n) = \mathbb{I}_{(\sum_{t=-\Delta}^{T_n-1+\Delta} \mathbb{I}(y_n(t)\neq 0) \leq \bar{N}_n)}. \tag{4.2}$$

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 $Y_n = \cup_t \{y_n(t)\}$. Alternatively, if the label signal $y_n(t)$ contains zeros then we expect $Y_n \cup \{0\} = \cup_t \{y_n(t)\}$. Consequently, the corresponding probabilistic model for the signal label Y_n is given by:

$$P(Y_n|y_n) = \mathbb{I}_{(Y_n = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})} + \mathbb{I}_{(Y_n \cup \{0\} = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})}. \tag{4.3}$$

4.3 Solution approach

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.

4.3.1 Complete and incomplete data likelihood

Define the observed data as $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}, I_1 = 1, \ldots, I_N = 1\}$, the hidden data as $\mathcal{H} = \{y_1, \ldots, y_N\}$, the unknown parameters as $\boldsymbol{\theta} = [\mathbf{w}^T, \mathbf{b}^T]^T$, and the tuning parameters as $\boldsymbol{\phi} = [\bar{N}_1, \ldots, \bar{N}_N]^T$. According to the graphical model shown in Figure 4.2, the complete

data likelihood $P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi})$ is computed as

$$P(\mathcal{X}) \prod_{n=1}^{N} \overbrace{\left[\mathbb{I}_{(Y_n = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})} + \mathbb{I}_{(Y_n \cup \{0\} = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})}\right]}^{P(Y_n|y_n)} \overbrace{\mathbb{I}_{(\sum_{t=-\Delta}^{T_n-1+\Delta} \mathbb{I}(y_n(t)\neq 0) \leq \bar{N}_n)}}^{P(I_n=1|y_n;\bar{N}_n)} \overbrace{\prod_{t=-\Delta}^{T_n-1+\Delta} P(y_n(t)|x_n; \mathbf{w}, \mathbf{b})}^{P(y_n|x_n;\mathbf{w},\mathbf{b})}. \tag{4.4}$$

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

$$\begin{aligned} L(\boldsymbol{\theta}) = & \sum_{y_1(-\Delta)=0}^{C} \dots \sum_{y_1(T_1+\Delta)=0}^{C} \sum_{y_2(-\Delta)=0}^{C} \dots \sum_{y_2(T_2+\Delta)=0}^{C} \\ & \dots \sum_{y_N(-\Delta)=0}^{C} \dots \sum_{y_N(T_N+\Delta)=0}^{C} P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi}) \end{aligned} \tag{4.5}$$

Taking the natural logarithm of (4.5) and plugging in the probability with (4.4) produces the incomplete log-likelihood:

$$\begin{aligned} l(\boldsymbol{\theta}) = & \log P(\mathcal{X}) + \sum_{n=1}^{N} \log\Big(\sum_{y_n(-\Delta)=0}^{C} \dots \sum_{y_n(T_n-1+\Delta)=0}^{C} \\ & \left[\mathbb{I}_{(Y_n = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})} + \mathbb{I}_{(Y_n \cup \{0\} = \cup_{t=-\Delta}^{T_n-1+\Delta}\{y_n(t)\})}\right] \\ & \mathbb{I}_{(\sum_{t=-\Delta}^{T_n-1+\Delta} \mathbb{I}(y_n(t)\neq 0) \leq \bar{N}_n)} \prod_{t=-\Delta}^{T_n-1+\Delta} P(y_n(t)|x_n; \mathbf{w}, \mathbf{b})\Big). \end{aligned} \tag{4.6}$$

Calculating the incomplete data likelihood in (4.5) involves enumerating all possible instance labels, which is computationally intractable especially when the number of instance per signal is large.

4.3.2 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 [80], where the proposed approach is very similar to the one in [95]. 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 $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = E_{\mathcal{H}|\mathcal{D};\boldsymbol{\theta}^i}[\log P(\mathcal{H}, \mathcal{D}; \boldsymbol{\theta})]$.
- **M-step**: $\boldsymbol{\theta}^{i+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)$

The auxiliary function for the proposed model is given by

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n-1+\Delta} [\sum_{c=0}^{C} P(y_n(t) = c|\mathcal{D}; \bar{N}_n, \boldsymbol{\theta}^i) \cdot (\mathbf{w}_c^T \mathbf{x}_{nt} + b_c) - \log(\sum_{u=0}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{nt} + b_u})] + const.$$

The derivation of the auxiliary function for our model is explained in Appendix B. The maximization of the auxiliary function $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)$ provides an update rule for both dictionary words $\mathbf{w}$ and bias terms $\mathbf{b}$ with a learning rate γ:

$$\begin{aligned} \mathbf{w}_c^{i+1} &= \mathbf{w}_c^i + \gamma \frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_c} \mid_{\boldsymbol{\theta}=\boldsymbol{\theta}^i}, \\ b_c^{i+1} &= b_c^i + \gamma \frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_c} \mid_{\boldsymbol{\theta}=\boldsymbol{\theta}^i}, \end{aligned}$$

for $c = 0, 1, \ldots, C$, where

$$\frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_c} = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n - 1 + \Delta} [P(y_n(t) = c | Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) - P(y_n(t) = c | x_n; \mathbf{w}, \mathbf{b})] \mathbf{x}_{nt}, \tag{4.7}$$

$$\text{and } \frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_c} = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n - 1 + \Delta} [P(y_n(t) = c | Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) - P(y_n(t) = c | x_n; \mathbf{w}, \mathbf{b})]. \tag{4.8}$$

The term $P(y_n(t) = c|x_n; \mathbf{w}, \mathbf{b})$ in (4.7) and (4.8), which is calculated using (4.1), is regarded as a prior probability of the instance label, i.e., without any information about the signal label or a sparsity constraint. The term $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i)$ can be viewed as a posterior instance label probability that takes into account the signal label and the sparsity constraint. Denote the difference between the posterior probability of $y_n(t)$ and its prior by $a_{nc}(t) = P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) - P(y_n(t) = c|x_n; \mathbf{w}, \mathbf{b})$.

The gradient calculation w.r.t. $\mathbf{w}_c$ in (4.7) is performed as a convolution between $a_{nc}(t)$ and the time-reversed signal $x_n(-t)$ such that

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial w_c(t)} &= \sum_{n=1}^{N} \sum_{\tau=0}^{T_n - 1} a_{nc}(t + \tau) x_n(\tau) \\ &= \sum_{n=1}^{N} a_{nc}(t) * x_n(-t) \end{aligned}$$

for $t = -\Delta, -\Delta + 1, \ldots, \Delta$. When both signal $\mathbf{x}_n$ and kernel w_c are 2-D, the gradient

step in (4.7) is

$$\begin{aligned}\frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial w_c(f,t)} &= \sum_{n=1}^{N} \sum_{\tau=0}^{T_n-1} a_{nc}(t+\tau) x_n(f,\tau) \\ &= \sum_{n=1}^{N} a_{nc}(t) * x_n(f,-t)\end{aligned}$$

for $f = 1, 2, \ldots, F$ and $t = -\Delta, -\Delta+1, \ldots, \Delta$.

Regularization: To guarantee the boundedness of the solution, we add an L_2-regularization term $-\lambda_r/2 \sum_{c=0}^{C} \|\mathbf{w}_c\|^2$ in the M-step.

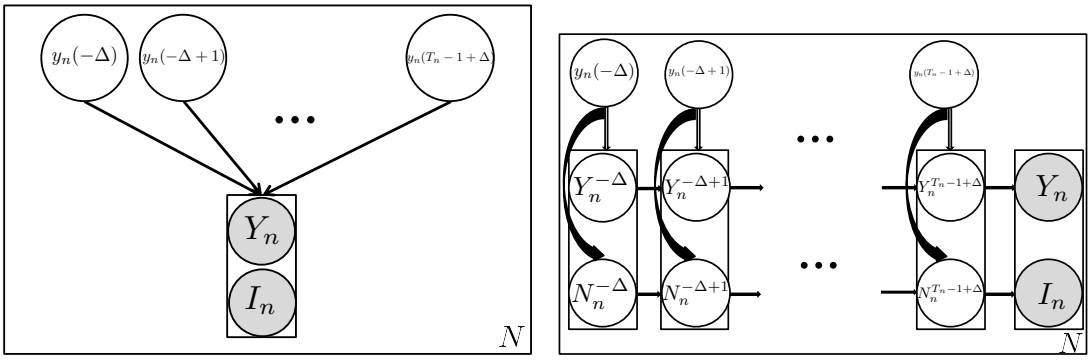
4.3.3 Key challenge

The computation of the gradient in (4.7) and (4.8) involves the computation of the posterior probability $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i)$ 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.

4.4 Graphical model reformulation for the E-step

The goal of the E-step is to obtain the posterior probability $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i)$, which first requires the calculation of the prior probability. To efficiently compute the prior probability $P(y_n(t)|x_n; \mathbf{w}, \mathbf{b})$ as a function of t for each signal x_n, $C+1$ convo-

lutions of the form $w_c * x_n$ are performed to obtain the values of $\mathbf{w}_c^T \mathbf{x}_{nt}$ in (4.1) 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) Original model (b) Graphical chain model

Figure 4.3: The label portion of the proposed graphical model (a) and its reformulation as a chain (b) (reproduction of [148]).

4.4.1 Chain model reformulation

Consider the label portion of the original graphical model in Figure 4.3(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 4.3(a) does not offer an immediate efficient approach for computing the posterior. Hence, we propose a reformulation of the model as follows. We denote

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

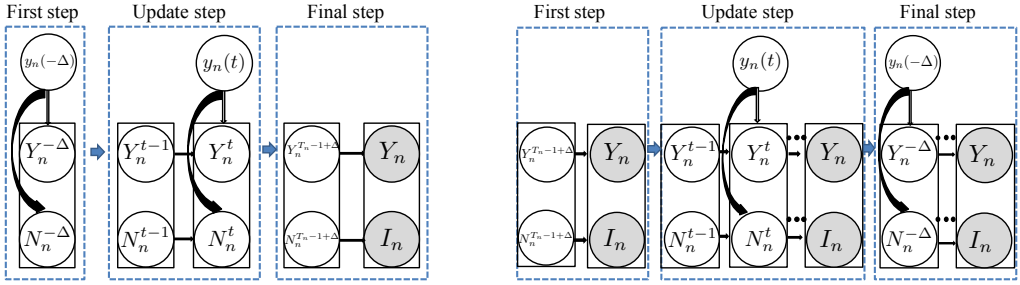
as the label set and the number of non-zero class instances of the first t instances in nth signal. Both of the aforementioned newly introduced variables follow a recursive rule

$$Y_n^{t+1} = Y_n^t \cup \{y_n(t+1) \mid y_n(t+1) \neq 0\} \tag{4.9}$$

$$N_n^{t+1} = N_n^t + \mathbb{I}(y_n(t+1) \neq 0) \tag{4.10}$$

for $t = -\Delta, \ldots, T_n - 1 + \Delta - 1$. This reformulation gives rise to the chain model in Figure 4.3(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



(a) Forward algorithm (b) Backward algorithm

Figure 4.4: Graphical illustration of the chain forward and backward message passing routines (reproduction of [148]).

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 $P_{ntc} = P(y_n(t) = c, Y_n, I_n | x_n; \bar{N}_n, \boldsymbol{\theta}^i)$ and then applying Bayes rule as

$$P(y_n(t) = c | y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) = P_{ntc} / \textstyle\sum_{c=0}^{C} P_{ntc}. \tag{4.11}$$

We compute the joint probabilities P_{ntc} 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 $P(Y_n^t, N_n^t|x_n;\boldsymbol{\theta}^i)$ for $t=-\Delta,\ldots,T_n-1+\Delta$. Denote an element in the power set of all class labels in the nth signal by $\mathbb{L}\in 2^{Y_n}$. A forward message is defined as

$$\alpha_t(\mathbb{L},l)\triangleq P(Y_n^t=\mathbb{L},N_n^t=l|x_n;\boldsymbol{\theta}^i).$$

The first message is initialized as Figure 4.4(a) first step shows

$$\alpha_1(\mathbb{L},l)=\begin{cases}P(y_n(-\Delta)=0|x_n;\mathbf{w}^i,\mathbf{b}^i), & l=0,\mathbb{L}=\{0\};\\ P(y_n(-\Delta)=c|x_n;\mathbf{w}^i,\mathbf{b}^i), & l=1,\mathbb{L}=\{c\},c\in Y_n;\\ 0, & Otherwise,\end{cases}$$

The update equation for the forward message of the tth instance is calculated by marginalizing over the $(t-1)$th state and the tth instance label as Figure 4.4(a) update step shows:

$$\begin{aligned}\alpha_t(\mathbb{L},l)=&\alpha_{t-1}(\mathbb{L},l)P(y_n(t)=0|x_n;\mathbf{w}^i,\mathbf{b}^i)\\&+\mathbb{I}_{(l\neq 0)}\sum_{c=1}^{C}P(y_n(t)=c|x_n;\mathbf{w}^i,\mathbf{b}^i)\\&\cdot[\alpha_{t-1}(\mathbb{L},l-1)+\mathbb{I}_{(c\in\mathbb{L})}\alpha_{t-1}(\mathbb{L}_{\setminus c},l)].\end{aligned}\tag{4.12}$$

In the final step, $P(Y_n=\mathbb{L},I_n=1|x_n;\boldsymbol{\theta}^i)=\sum_{l=1}^{\bar{N}_n}\alpha_{T_n-1+\Delta}(\mathbb{L},l)$.

Step 2: Backward message passing. In this step, the goal is to compute the conditional joint state probability defined as $P(Y_n,I_n=1|Y_n^t,N_n^t,x_n;\boldsymbol{\theta}^i,\bar{N}_n)$. We denote a

backward message as

$$\beta_t(\mathbb{L}, l) \triangleq P(Y_n, I_n = 1 | Y_n^t = \mathbb{L}, N_n^t = l, x_n; \boldsymbol{\theta}^i, \bar{N}_n).$$

According to the graphical model in Figure 4.4(b) such that Y_n, I_n is only dependent on $Y_n^{T_n-1+\Delta}, N_n^{T_n-1+\Delta}$, the first backward message is initialized as

$$\beta_{T_n-1+\Delta}(\mathbb{L}, l) = \mathbb{I}_{(l \leq \bar{N}_n)} \mathbb{I}_{(\mathbb{L}=Y_n)}.$$

The update equation for the $t-1$th backward message is calculated by marginalizing over the tth backward message as Figure 4.4(b) update step shows $\beta_{t-1}(\mathbb{L}, l) =$

$$\sum_{c=0}^{C} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{I}_{(c \neq 0)}) P(y_n(t) = c | x_n, \mathbf{w}^i, \mathbf{b}^i). \tag{4.13}$$

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 $l \geq t + 1$, (ii) $\beta_t(\mathbb{L}, l) = 0$ for $l > \bar{N}_n$, $\mathbb{L} \notin 2^{Y_n}$. Where (i) is from the definition of N_n^t in (4.10) and (ii) is from the sparsity constraint in (4.2) and the definition of Y_n^t in (4.9) such that each $Y_n^{t-1} \subseteq Y_n^t$, and $Y_n^{T_n-1+\Delta} = \mathbb{L}$.

Step 3: Joint probability. Finally, the numerator of (4.11) for $t = -\Delta, \ldots, T_n - 1 + \Delta$ is computed using all of the forward messages and the backward messages as

$$\begin{aligned} P(y_n(t) = c, Y_n, I_n = 1 | x_n; \boldsymbol{\theta}^i, \bar{N}_n) &= p(y_n(t) = c | x_n; \mathbf{w}^i, \mathbf{b}^i) \\ &\cdot \sum_{\mathbb{L} \in 2^{Y_n}} \sum_{l=0}^{\bar{N}_n^*} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{I}_{(c \neq 0)}) \alpha_{t-1}(\mathbb{L}, l)), \end{aligned} \tag{4.14}$$

where $\bar{N}_n^* = \min(\bar{N}_n - \mathbb{I}_{(c\neq 0)}, t)$. Since $Y_n^{-\Delta}, N_n^{-\Delta}$ is only dependent on the first instance $y_n(-\Delta)$ as Figure 4.4(b) shows,

$$P(y_n(-\Delta) = c, Y_n, I_n = 1 | x_n, \bar{N}_n, \boldsymbol{\theta}^i) =$$
$$\beta_{-\Delta}(\{c \neq 0\}, \mathbb{I}_{(c\neq 0)}) p(y_n(1) = c | x_n; \mathbf{w}^i, \mathbf{b}^i).$$

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 $l > \bar{N}_n$, the effective calculation and actual need of storing both forward and backward message is for $0 \leq l \leq \min(\bar{N}_n, t)$.

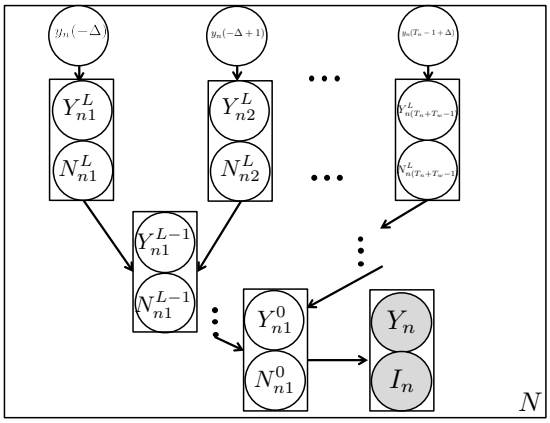
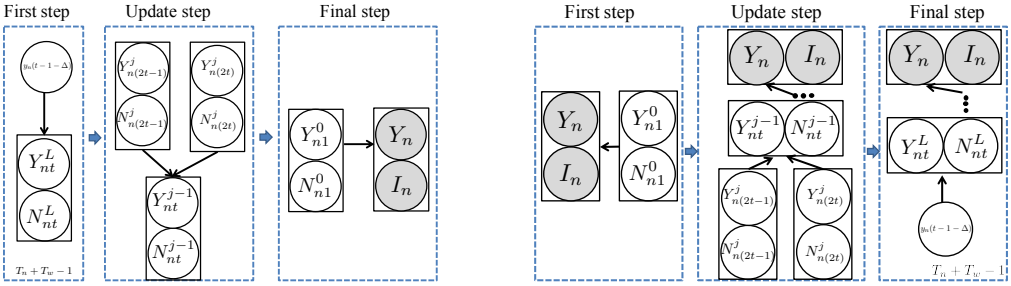


Figure 4.5: Graphical model reformulation as a tree

4.4.2 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 $\mathcal{T}(S_{nt}^j, j)$ with depth $L+1$, where j indicates the tree level, S_{nt}^j is the node



(a) Forward algorithm (b) Backward algorithm

Figure 4.6: Graphical illustration of the tree forward and backward message passing routines (reproduction of [148]).

variable at index t in jth level and $L = \lceil \log_2(T_n + T_w - 1) \rceil$) graph structure reformulation of the original graphical model in Figure 4.3(a) to make the E-step calculation more efficient.

In the complete full binary tree structure, each node of the tree S_t^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 nth tree and N_{nt}^j as the number of non-zero class instances of all ancestors of node t in level j of the nth tree. We present the recursive relation At the leaf's level, we assign the values as:

$$\begin{aligned} Y_{nt}^L &= \{y_n(t - \Delta - 1) | y_n(t - \Delta - 1) \neq 0\}, \\ N_{nt}^L &= \mathbb{I}_{(y_n(t-\Delta-1)\neq 0)} \end{aligned}$$

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 formula:

$$Y_{nt}^{j-1} = Y_{n(2t-1)}^{j} \cup Y_{n(2t)}^{j} \tag{4.15}$$

$$N_{nt}^{j-1} = N_{n(2t-1)}^{j} + N_{n(2t)}^{j} \tag{4.16}$$

for $t = 1, 2, \ldots, 2^{j-1}$. This reformulation gives rise to the tree model in Figure 4.5. Note that $Y_{n1}^{0} = \bigcup_t \{y_n(t) | y_n(t) \neq 0\} = Y_n$ and $N_{n1}^{0} = \sum_t I_{(y_n(t)\neq 0)}$ 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 $P(y_n(t), Y_n, I_n | x_n; \bar{N}_n, \mathbf{w}^i)$ can be computed efficiently with the following three steps:

Step 1: Forward message passing. In this step, the goal is to compute the joint state probabilities $P(Y_{nt}^{j}, N_{nt}^{j} | x_n; \boldsymbol{\theta}^i)$ 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 nth signal by $\mathbb{L} \in 2^{Y_n}$. The forward message is defined as

$$\alpha_t^j(\mathbb{L}, l) \triangleq P(Y_{nt}^{j} = \mathbb{L}, N_{nt}^{j} = l | x_n; \boldsymbol{\theta}^i).$$

At the leaf level, the forward messages are initialized as $\alpha_t^L(\mathbb{L}, l) =$

$$\begin{cases} P(y_n(t - \Delta - 1) = 0 | x_n; \boldsymbol{\theta}^i), & l = 0, \mathbb{L} = \emptyset; \\ P(y_n(t - \Delta - 1) = c | x_n; \boldsymbol{\theta}^i), & l = 1, \mathbb{L} = \{c\}, c \in Y_n; \\ 0, & Otherwise, \end{cases}$$

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

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

for $t = T_n + T_w, T_n + T_w + 1, \ldots, 2^L$.

The update for the forward message of the tth node in j-1th level is calculated by marginalizing over its left parent (the $(2t-1)$th message in jth level) and the right parent ($(2t)$th message in jth level) as

$$\alpha_t^{j-1}(\mathbb{L}, l) = \sum_{\mathbb{A} \subseteq \mathbb{L}} \sum_{\mathbb{B} \subseteq \mathbb{L}} \sum_{a=0}^{l} \mathbb{I}_{(\mathbb{A}+\mathbb{B}=\mathbb{L})} \alpha_{2t-1}^j(\mathbb{A}, a) \alpha_{2t}^j(\mathbb{B}, l-a). \tag{4.17}$$

We summarize the forward message step in Figure 4.6(a).

Step 2: Backward message passing. In this step, the goal is to compute the joint state posterior probability $P(Y_n, I_n = 1 | Y_{nt}^j, N_{nt}^j, x_n; \boldsymbol{\theta}^i, \bar{N}_n)$. We denote a backward message as

$$\beta_t^j(\mathbb{L}, l) \triangleq P(Y_n, I_n = 1 | Y_{nt}^j = \mathbb{L}, N_{nt}^j = l, x_n; \boldsymbol{\theta}^i, \bar{N}_n).$$

The first backward message is initialized as

$$\beta_1^0(\mathbb{L}, l) = \mathbb{I}_{(l \leq \bar{N}_n)} \mathbb{I}_{(\mathbb{L}=Y_n)}.$$

The update equation for the backward messages are calculated as follows:

$$\beta_{2t-1}^j(\mathbb{A}, a) = \sum_{\mathbb{E} \in 2^{Y_n}} \sum_{e=0}^{\bar{N}_n - a} \beta_t^{j-1}(\mathbb{A} \cup \mathbb{E}, a+e) \alpha_{2t}^j(\mathbb{E}, e). \tag{4.18}$$

$$\beta_{2t}^{j}(\mathbb{E}, e) = \sum_{\mathbb{A}\in 2^{Y_n}} \sum_{a=0}^{\bar{N}_n - e} \beta_t^{j-1}(\mathbb{A}\cup\mathbb{E}, a+e)\alpha_{2t-1}^{j}(\mathbb{A}, a). \tag{4.19}$$

We summarize the backward message step in Figure 4.6(b). **Note:** To efficiently calculate and store the forward and backward messages, we consider the following results: (i) $\alpha_t^j(\mathbb{L}, l) = 0$ for $l > 2^{L-j} + 1$. (ii) $\beta_t^j(\mathbb{L}, l) = 0$ for $l > \bar{N}_n$ or $\mathbb{L} \notin 2^{Y_n}$ for $j = 0, 1, \ldots, L$ and $t = 1, \ldots, 2^j$. Where (i) is obtained from the recursive formula of N_{nt}^j in (4.16) with the initialization of N_{nt}^L and (ii) is obtained from the sparsity constraint in (4.2) and the definition of Y_{nt}^j in (4.15) such that each $Y_{nt}^j \subseteq Y_{nt/2}^{j-1}$, and $Y_{n1}^0 = \mathbb{L}$. Based on summary (i) that $\alpha_t^j(\mathbb{L}, l) = 0$ when $l > 2^{L-j}+1$, and (ii) that $\beta_t^j(\mathbb{L}, l) = 0$ when $l > \bar{N}_n$, the effective calculation and actual storing of both forward and backward message is for $0 \leq l \leq \min(\bar{N}_n, 2^{L-j} + 1)$.

Step 3: Joint probability. Finally, the numerator on the RHS of (4.11) for $t = 1, \ldots, T_n$ is computed using the backward message $\beta_t^L(\mathbb{L}, l)$ such that

$$P(y_n(t) = c, Y_n, I_n = 1 | x_n; \boldsymbol{\theta}^i, \bar{N}_n) = \beta_t^L(\{c\}, \mathbb{I}_{(c\neq 0)}) \cdot p(y_n(t) = c | x_n; \mathbf{w}^i, \mathbf{b}^i). \tag{4.20}$$

Convolutive model on tree: Based on update equation of the forward message in (4.17), if we treat each α_t^{j-1} message of a particular set value $\mathbb{L}$ as a discrete signal $\alpha_t^{j-1}(\mathbb{L}, t)$, then the update of each forward message is performing a convolution between $\alpha_{2t-1}^j(\mathbb{A}, t)$ and $\alpha_{2t}^j(\mathbb{B}, t)$. For the update on the backward message in (4.18) and (4.19), the update of backward message signal $\beta_{2t-1}^j(\mathbb{A}, t)$ is a convolution between $\beta_t^{j-1}(\mathbb{A}\cup\mathbb{E}, -t)$ and $\alpha_{2t}^j(\mathbb{E}, t)$ and the update of backward message signal $\beta_{2t}^j(\mathbb{E}, t)$ is a convolution between $\beta_t^{j-1}(\mathbb{A}\cup\mathbb{E}, -t)$ and $\alpha_{2t-1}^j(\mathbb{A}, t)$.

4.4.3 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 nth signal. Therefore, the time complexity for both (i) and (iii) is $\mathcal{O}(\sum_{n=1}^{N}(C+1)FT_nT_w)$. The space complexity is $\mathcal{O}((C+1)F(T_n+T_w-1))$ and $\mathcal{O}((C+1)FT_w)$ 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 $\mathcal{O}(\sum_{n=1}^{N}|Y_n|2^{|Y_n|}(T_n+T_w)\bar{N}_n)$ and $\mathcal{O}(2^{|Y_n|}(T_n+T_w)\bar{N}_n)$ 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 $\mathcal{O}(\sum_{n=1}^{N}4^{|Y_n|}(T_n+T_w)(\log_2\bar{N}_n)^2)$ and $\mathcal{O}(2^{|Y_n|}(T_n+T_w)\log_2\bar{N}_n)$ respectively.

4.5 Prediction

In addition to identifying the analysis words $\mathbf{w}_c$, 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 $x_n^{\text{test}}(t)$ for $t = 1, 2, \ldots, T_n$, the goal is to predict the time instance label signal $\hat{y}_n(t)$ and the signal label Y_n.

4.5.1 Time instance prediction:

For an unlabeled test instance $\mathbf{x}_{nt}^{\text{test}}$, the predicted label is

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

4.5.2 Signal label prediction:

For an unlabeled test signal x_n^{test}, the predicted signal label set using the **union rule** is

$$\hat{Y}_n^U = \cup_{t=-\Delta}^{T_n-1+\Delta} \{\hat{y}_n^{\text{test}}(t) \mid \hat{y}_n^{\text{test}}(t) \neq 0\}.$$

Alternatively, the signal label can be predicted by **maximizing a posterior probability (MAP) rule**:

$$\hat{Y}_n^P = \arg \max_{\mathbb{A} \in \{0,1\}^C} P(Y_n = \mathbb{A}, I_n = 1 | x_n^{\text{test}}; \bar{N}_n, \mathbf{w}, \mathbf{b}),$$

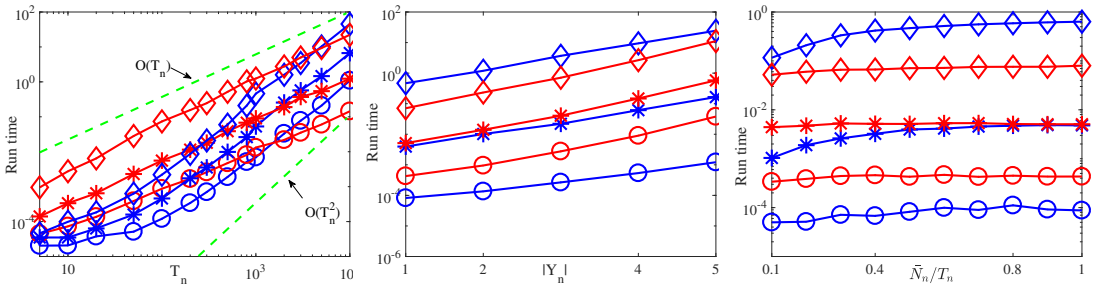
where,

$$P(Y_n = \mathbb{A}, I_n | x_n^{\text{test}}; \bar{N}_n, \mathbf{w}) = \textstyle\sum_{l=0}^{\bar{N}_n} \alpha_{T_n-1+\Delta}(\mathbb{A}, l).$$

4.6 Results and Analysis

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.

4.6.1 Run time analysis



(a) $\bar{N}_n/T_n = 0.2$ (b) $\bar{N}_n/T_n = 0.5$ (c) $|Y_n| = 2$

Figure 4.7: Running time versus $T_n, |Y_n|, \bar{N}_n$. (Blue color for chain and red color for tree algorithm. (a) $\circ : |Y_n| = 1$, $\star : |Y_n| = 3$, $\diamond : |Y_n| = 5$. (b)-(c) $\circ : T_n = 50$, $\star : T_n = 500$, $\diamond : T_n = 5000$.) (reproduction of [148])

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 $\bar{N}_n/T_n \in \{0.1, 0.2, \ldots, 1.0\}$.

Figure 4.7(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 $0.2T_n$. 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

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

Table 4.2: Runtime values for the chain-based and the tree-based E-step calculation as a function of T_n for four scenarios.

the tree approach when T_n is small or when $|Y_n|$ is large. Even though Figure 4.7(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 [94]). Figure 4.7(c) exhibits a near-constant runtime with respect to the sparsity factor $\bar{N}_n/T_n$ for the tree inference. Runtime values for both models are shown in Table 4.2.

4.6.2 Synthetic datasets and results

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.

Data generation: Below we describe the two synthetic datasets.

(i) **Gabor basis dataset:** This dataset is constructed with nine different Gabor filters

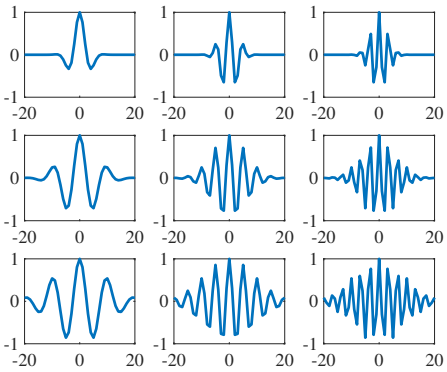


Figure 4.8: Nine Gabor basis used in the experiment (reproduction of [148]).

as 1-D signal templates as shown in 4.8(a):

$$s_{(a,f)}(t) = \cos(2\pi f t)\ e^{-\frac{t^2}{2a^2}},\ \text{for } t = -20, -19, \ldots, 20$$

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 $k \in \{1, ..., m_n\}$, 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 $\mathbb{U}[1, 2]$.

4) We then generate $y_c^n(t) = \sum_{k=1}^{m_n} A_k \mathbb{I}_{(c-c_k)} \mathbb{I}_{(t-t_k)}$ for each class c and generate the signal $\tilde{x}_n = \sum_{c=1}^{9} y_c^n * s^c$, where $s^1(t) = s_{(1,0.1)}(t), s^2(t) = s_{(1,0.2)}(t), \ldots, s^9(t) = s_{(3,0.3)}(t)$. 5) Lastly, we generate the final x_n by adding to $\tilde{x}_n$ 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 $\tilde{x}_n$s and we use SNR_{dB} to control the signal to noise ratio for our final data.

(ii) **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 3×200 as synthetic 2-D signals ($N = 200$, $F = 3$, $T = 200$). For each generated 2-D signal, we then determined the label for each of its time instance t by matching the 3×3 sub-window starting at t to three pre-defined class-specific templates[2] shown in Figure 4.10(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.

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 λ_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 λ_r and T_w and used those to present the prediction performance as

[2]These class templates are defined by selecting the most frequent 3 patterns in the generated 2-D signals.

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 $\text{SNR}_{dB} \in \{-10, -5, 0, 5, 10, 15, 20, 25\}$ (see Figure 4.9(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 [104] (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 $\max_t \tilde{d_k} * x_n^{\text{train}} \mid_t$, where $\tilde{d_k}$ is a time reversed version of d_k ($\tilde{d_k}(t) = d_k(-t)$). 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 [104]. 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 $K \in \{3, 7, 9, 15\}$.

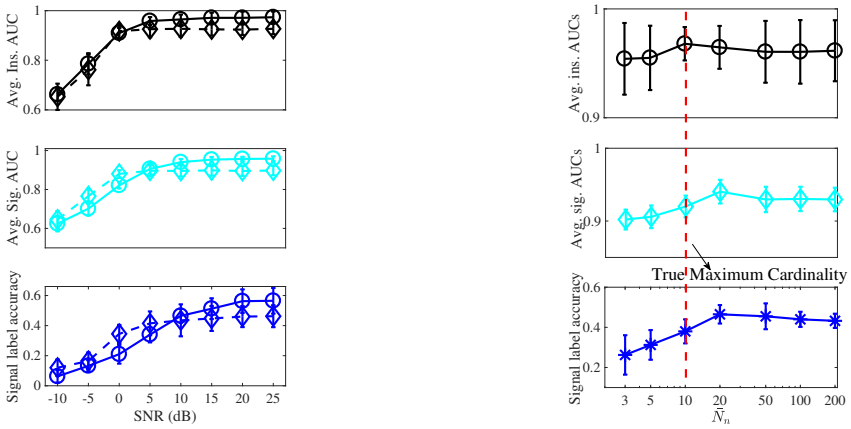
[3] Although the two steps can be combined to yield improved performance, the combination of the two steps requires further research beyond the scope of this paper.

Evaluation metric: In computing the instance level detection area-under-the-curve (AUC), we calculate an AUC for each class c and obtain AUC $= 1/C \sum_c \text{AUC}_c$. 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 $P_t = P(y_n(t) = c | x_n^{\text{test}}; \mathbf{w}, \mathbf{b})$ as a test score [76].

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 $1 - \prod_{t=-\Delta}^{T_n - 1 + \Delta}(1 - P_t)$.

Results on synthetic datasets: (i) **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 $\lambda_s = 1, K = 9$ 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 4.9(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.

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 4.9(b). As Figure 4.9(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.



(a) $\bar{N}_n = 20$ (b) $\text{SNR}_{dB} = 10$

Figure 4.9: Gabor basis dataset performance metrics for the WSCADL approach (solid $\circ$) and the GDL-LR approach (dashed $\diamond$) as a function of SNR_{dB} in (a), and for the WSCADL approach as a function of $\bar{N}_n$ in (b) (reproduction of [148]).

In this setting, we allow some of the zero instance labels to be predicted as non-zero. From Figure 4.9(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 4.3. 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

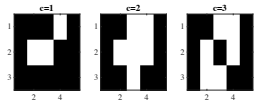
AUCs for the GDL-LR approach is mostly higher than our WSCADL approach. We suspect that since the GDL-LR approach performs an unsupervised dictionary learning followed by a classifier training in a separate fashion, the resulting words may have large variability. We believe that this can be fixed by 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. Hence, we provide the results for the two-step approach only.

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	89.58±4.85
c=7	**87.83±17.19**	**94.32±9.89**	81.26±15.95	83.73±15.15
c=8	**98.40±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 4.3: Gabor basis dataset: Detection AUCs (%) for the WSCADL and the GDL-LR approaches with optimal tuning parameters

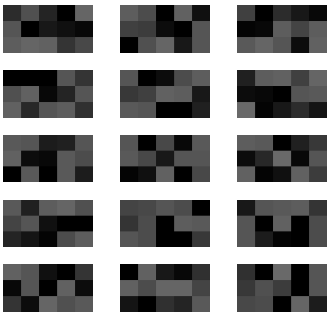
(ii) **Binary patterns dataset:** The hyper-parameters are set to be $\lambda_r = 10^{-2}$, $\bar{N}_n = 3$ and $T_w = 5$ via the aforementioned cross validation. The optimal kernel size 3×5 is slightly higher than the ground truth window size 3×3. For the GDL-LR approach, the optimal dictionary window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

The learned WSCADL words in Figure 4.10(b) and the learned GDL-LR words in 4.10(c) show that WSCADL is able to recover the true patterns while the GDL-LR approach fails. Figure 4.10(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

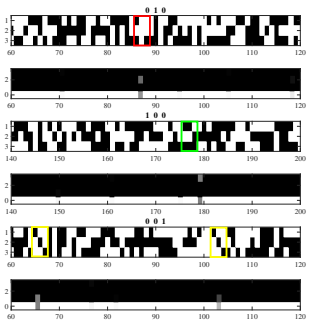


(a) True pattern

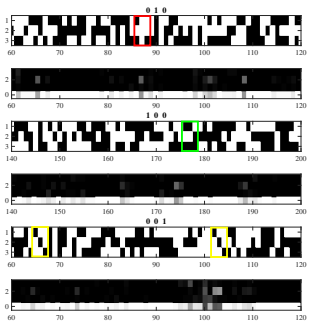
(b) Learned WSCADL words



(c) Learned GDL-LR words



(d) Localization for WSCADL



(e) Localization for GDL-LR

Figure 4.10: Binary patterns dataset setting and results (reproduction of [148]).

Table 4.4.

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**	**99.78±0.45**	56.30±8.77	51.89±10.97

Table 4.4: Binary patterns dataset: Detection AUCs (%) for the WSCADL and the GDL-LR approaches with optimal tuning parameters

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.

4.6.3 Real-world datasets and results

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

(i) **AASP challenge - office live scene dataset:** This dataset consists of audio recordings of sounds taken in an office environment [38]. The training dataset consists of 20 to 22 individual events (such as door slam, phone ringing, and pen drop) recording with varying time from 0.05s to 20s for 16 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.

(ii) **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 [14]. Each recording may contain multiple species.

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 [14]. (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 $\mathbb{R}^{707\times612}$ to $\mathbb{R}^{256\times200}$ and test spectrograms are from $\mathbb{R}^{707\times6120}$ to $\mathbb{R}^{256\times2000}$, on experiment 2, spectrograms are down-sampled from $\mathbb{R}^{707\times6120}$ to $\mathbb{R}^{256\times200}$. For HJA dataset, spectrograms are from $\mathbb{R}^{256\times1249}$

to $\mathbb{R}^{256\times 200}$).

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

(i) **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 $\bar{N}_n \in \{5, 10, 60, 100, 200\}$ 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%.

(ii) **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 tunning parameters considered were: the window size $T_w \in \{10, 20, 50, 100\}$, the cardinality constraint $\bar{N}_n \in \{10, 20, 40, 60, 100, 160, 200\}$ 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.

Real-world 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±18.60	**54.75±3.32**	**77.65±16.74**
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 4.5: Instance level and signal detection AUCs (%) for both experiments across five MC runs.

(i) **Office live event detection:** We compared our WSCADL approach with discriminative dictionary learning approaches: sparse representation-based classification (SRC) [134]; label consistent K-SVD (LCKSVD1,LCKSVD2) [51]; dictionary learning with structured incoherence and shared features (DLSI) [99]; Fisher discrimination dictionary learning (FDDL) [140]; dictionary learning for separating the particularity and the commonality (COPAR) [54]; fast low-rank shared dictionary learning for object classification (LRSDL) [118].

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 $\bar{N}_n = 10$ as shown in Figure 4.11, 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

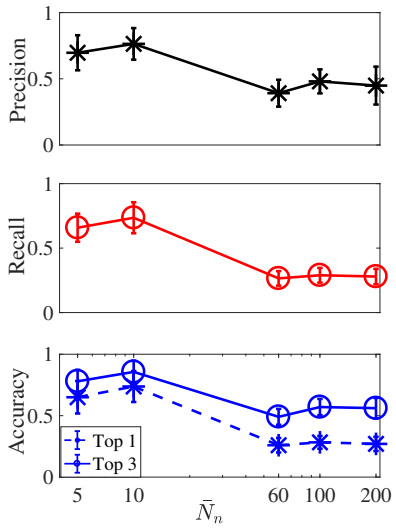


Figure 4.11: Prediction accuracy as a function of the cardinality parameter $\bar{N}_n$ on the AASP dataset (reproduction of [148]).

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 4.12(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 4.12(b). The instance and signal label detection receiver operating curves (ROCs) for the proposed method are shown in Figure 4.12(c) and (d), and the resulting AUCs are shown in Table 4.5). The average and maximum detection AUCs across 16

Table 4.6: Signal evaluation metrics (%) for various methods on HJA dataset. ↓ (↑) next to a metric indicates that the performance improves when the metric is decreased (increased). The results from column MLR to M-NN are extracted from Table 4 in [94].

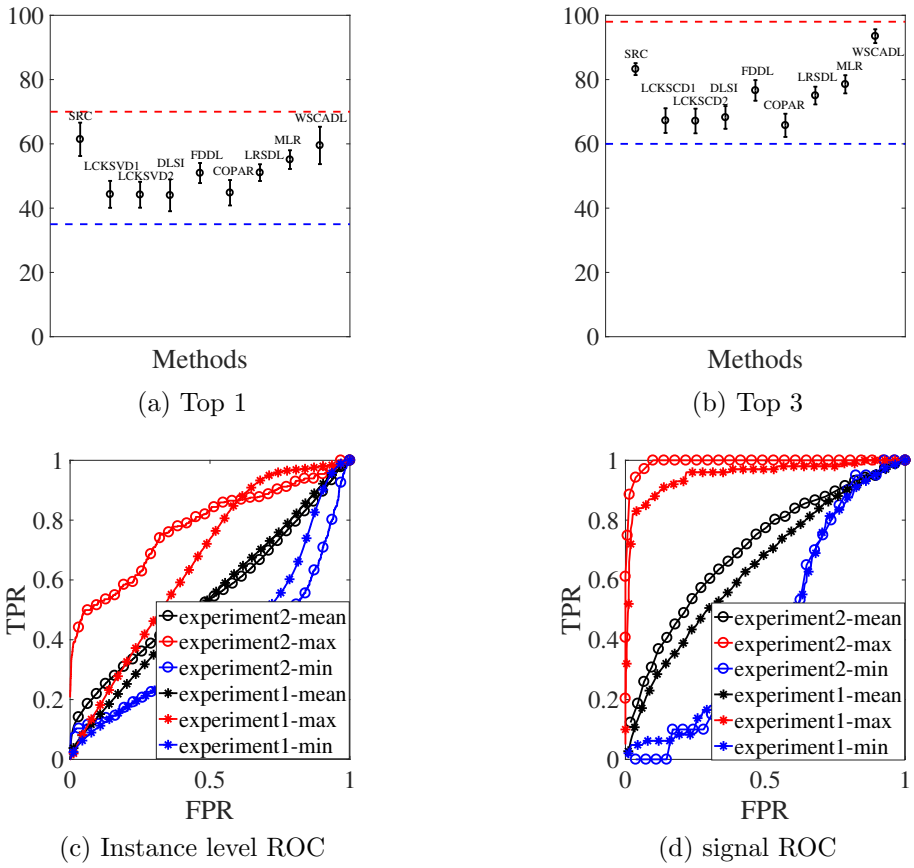
Method	WSCADL	GDL-LR	MLR	SIM	Mfast	LSB	M-SVM	M-NN
↓ Ham-loss	05.5±0.0^P	06.1±0.8	09.6±1.0	15.9±1.5	05.5±1.1	10.6±1.5	**04.5±0.6**	04.7±1.1
↓ rank loss	**02.2±0.4**	03.0±0.7	02.7±0.6	02.2±0.8	02.5±0.7	06.9±1.8	02.7±1.1	02.7±1.1
↑ avg. prec.	**94.6±0.6**	92.4±0.8	94.2±1.2	94.1±1.8	94.1±1.4	89.7±2.6	94.0±2.0	93.9±2.8
↓ one error	04.6±1.8	07.0±2.0	**03.8±1.8**	05.1±3.1	03.7±2.4	03.7±1.7	04.6±2.6	05.3±4.4
↓ coverage	16.2±0.9	17.0±1.9	13.9±1.6	**12.4±1.6**	13.4±1.6	21.7±3.6	13.2±1.6	13.4±1.3

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.

(ii) **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 [94], SIM [16], MIMLfast (short for Mfast) [43], and LSB-CMM (short for LSB) [69]. MIMLSVM (short for M-SVM) [156] and MIMLNN (short for M-NN) [153, 157].

We evaluated all of the approaches using multi-label evaluation metrics from [157]. 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



(a) Top 1

(b) Top 3

(c) Instance level ROC

(d) signal ROC

Figure 4.12: Classification accuracy (%) for the office live training data with mean and standard deviation over 5 MC runs with (a) selecting top 1 class and (b) selecting top 3 classes. Detection ROCs for (c) time instance level and (d) signal of both experiments (reproduction of [148]).

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 4.6 are directly extracted from Table 4 in [94]. 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.

Chapter 5: Weakly-supervised dictionary learning with multiple clusters [1]

In reality, the previously proposed dictionary learning uder the weak-supervision setting, which considers a single dictionary atom per class, may not capture the richness of patterns in data. We consider a slight generalization of the aforementioned model by assuming K dictionary words per class, $\mathbf{w}_{c.1}, \ldots, \mathbf{w}_{c.K}$ for class c. Dictionary words are analogous to dictionary atoms in synthesis dictionary learning.

5.1 The probabilistic modeling

Define the observed data as $\mathbf{D} = \{\mathcal{X}, \mathcal{Y}, I_1, \ldots, I_N\}$ with I_n's all equal to 1, the hidden data as $\mathbf{H} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$, the unknown parameters as $\mathbf{w} = [\mathbf{w}_{0.1}^T, \mathbf{w}_{0.2}^T, \ldots, \mathbf{w}_{C.K}^T]^T, \mathbf{b} = [b_{0.1}, b_{0.2}, \ldots, b_{C.K}]^T$, and the tuning parameters as $\boldsymbol{\theta} = \{\bar{N}_1, \ldots, \bar{N}_N\}$. The probabilistic graphical model for including a cluster component in weakly-supervised convolutive analysis dictionary learning (MC-WSCADL) is shown in Fig. 5.1.

To index a specific word within a class, we introduce a cluster label $z_{nt} \in \{0, \ldots, C\} \times \{1, \ldots, K\}$. The probability of the cluster label z_{nt} is modeled using multiple-class

[1] This chapter is a joint work with Dr. Raviv Raich, Xiaoli Fern and Jinsub Kim. This work was published as: Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly-supervised analysis dictionary learning with cardinality constraints." *In 2016 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 1-5. IEEE, 2016.

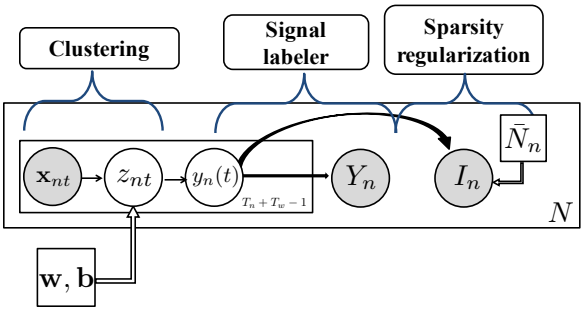


Figure 5.1: The proposed graphical model for MC-WSCADL (reproduction of [146]).

logistic regression [96]

$$P(z_{nt} = c.k|x_n; \mathbf{w}, \mathbf{b}) = \frac{e^{\mathbf{w}_{c.k}^T \mathbf{x}_{nt} + b_{c.k}}}{\sum_{u=0}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{u.v}^T \mathbf{x}_{nt} + b_{u.v}}}. \tag{5.1}$$

Moreover, the instance class label signal $y_n(t) \in \{0, \ldots, C\}$ is simply the class term in z_{nt} hence the probabilistic model for $y_n(t)$ given z_{nt} is given by $P(y_n(t) = u|z_{nt} = c.k) = \mathbb{I}(u = c)$. Consequently, the class label probability model can be obtained by marginalizing out z_{nt} as follows

$$P(y_n(t) = c|x_n; \mathbf{w}, \mathbf{b}) = \frac{\sum_{k=1}^{K} e^{\mathbf{w}_{c.k}^T \mathbf{x}_{nt} + b_{c.k}}}{\sum_{u=0}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{u.v}^T \mathbf{x}_{nt} + b_{u.v}}}. \tag{5.2}$$

In the rest of the paper, we consider only y_{bt} as the latent portion of the model. The probabilistic model for I_n and Y_n follows equations (4.2) and (4.3), respectively.

Complete and incomplete data likelihood: According to the probabilistic graphical model shown in Figure 5.1, the complete data likelihood follows the equation in (4.4) and the incomplete data likelihood follows the equation in (4.5). Calculating the incomplete data likelihood in (4.5) involves enumerating all possible instance labels, which is computationally intractable especially when the number of instance is large.

5.2 Solution approach

To resolve this, we consider an expectation maximization (EM) approach [80]. 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 $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = E_{\mathbf{y}|\mathbf{D};\boldsymbol{\theta}^i}[\log P(\mathbf{y}, \mathbf{D}; \boldsymbol{\theta})]$.
- **M-step**: $\boldsymbol{\theta}^{i+1} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i)$

The auxiliary function for the proposed model is given by

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = \sum_{n=1}^{N} \sum_{t=1}^{T_n} [\sum_{c=0}^{C} P(y_n(t) = c|\mathbf{D}; \bar{N}_n, \boldsymbol{\theta}^i) \cdot$$
$$\log(\sum_{k=1}^{K} e^{\mathbf{w}_{c.k}^T \mathbf{x}_{bt} + b_{c.k}}) - \log(\sum_{u=0}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{u.v}^T \mathbf{x}_{bt} + b_{u.v}})] + const.$$

Inference on E-step: Exact E-step inference has been proposed in [146], the goal is to obtain posterior probability $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i)$, which requires the following three steps:

(i) Calculation of the prior probability requires computing $C \times N$ convolutions between each signal x_n and each class analysis dictionary w_c.

(ii) Forward message passing algorithm to compute each time step $P(Y_n^t, N_n^t; \mathbf{x}_n; \boldsymbol{\theta}^i)$, where Y_n^t is defined as union set up until the tth time instances and N_n^t is the cardinality up until tth time instances.

(iii) Backward message passing algorithm to compute each $P(Y_n, I_n = 1|Y_n^t = \mathbb{L}, N_n^t = l, x_n; \boldsymbol{\theta}^i, \bar{N}_n)$.

Finally, we can obtain the posterior probability by the joint probably $P(y_n(t) = c, Y_n, I_n|$ $x_n, \bar{N}_n, \boldsymbol{\theta}^i)$ and apply the bayes rule.

Update on M-step: To update dictionary words, we maximize the auxiliary function Q. Since Q is a difference of convex functions, we apply the convex-concave procedure (CCCP) [151] to surrogate for Q denoted by $\tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)$. Using the gradient ascent method, we obtain the update rule as follows:

$$\mathbf{w}_{c.k}^{i+1} = \mathbf{w}_{c.k}^{i} + \gamma \frac{\partial \tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_{c.k}} \mid_{\mathbf{w}=\mathbf{w}^i}, \quad b_{c.k}^{i+1} = b_{c.k}^{i} + \gamma \frac{\partial \tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_{c.k} \mid_{b}= b^i}, \tag{5.3}$$

where

$$\begin{aligned} \frac{\partial \tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_{c.k}} &= \sum_{n=1}^{N} \sum_{t=1}^{T_n} [P(y_n(t) = c | Y_n x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) \cdot \\ & \frac{e^{\mathbf{w}_{c.k}^{iT} \mathbf{x}_{nt} + b_{c.k}^{i}}}{\sum_{v=1}^{K} e^{\mathbf{w}_{c.v}^{iT} \mathbf{x}_{bt} + b_{c.v}^{i}}} - \frac{e^{\mathbf{w}_{c.k}^{T} \mathbf{x}_{nt} + b_{c.k}}}{\sum_{u=1}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{u.v}^{T} \mathbf{x}_{nt} + b_{u.v}^{T}}}] \mathbf{x}_{nt}. \end{aligned} \tag{5.4}$$

and

$$\begin{aligned} \frac{\partial \tilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_{c.k}} &= \sum_{n=1}^{N} \sum_{t=1}^{T_n} [P(y_n(t) = c | Y_n x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) \cdot \\ & \frac{e^{\mathbf{w}_{c.k}^{iT} \mathbf{x}_{nt} + b_{c.k}^{i}}}{\sum_{v=1}^{K} e^{\mathbf{w}_{c.v}^{iT} \mathbf{x}_{bt} + b_{c.v}^{i}}} - \frac{e^{\mathbf{w}_{c.k}^{T} \mathbf{x}_{nt} + b_{c.k}}}{\sum_{u=1}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{u.v}^{T} \mathbf{x}_{nt} + b_{u.v}^{T}}}]. \end{aligned} \tag{5.5}$$

5.3 Simulations

In this section, we evaluate the performance of the proposed approach in terms of prediction accuracy at the instance level and at the signal level.

Setting: we randomly generate $B \in \{100, 200\}$ synthetic spectrograms with height (number of frequency bins) $F = 10$ and width (time frames) $T \in \{50, 100\}$ using three different class shapes and two different cluster shapes. We use $T - 6$ maximally overlapped 10×7 windows of the spectrograms as instances. The synthetic spectrograms are

(a) One cluster data (b) Two cluster data

Figure 5.2: (a). Signal labels: $Y_1 = \{2\}, Y_2 = \{1, 2\}, Y_3 = \{1, 2, 3\}$; (b). Signal labels: $Y_1 = \{1, 2, 3\}, Y_2 = \{2\}, Y_3 = \{1, 2, 3\}$ (reproduction of [146]).

generated by randomly placing one of the predefined shapes into the corresponding position based on the instance label. The instance label for each spectrogram is generated from a Dirichlet distribution with high prior on the novel class. To make the synthetic dataset close to the real-world dataset, each spectrogram contains some phenomenon of overlapping with different classes and each syllable shape is at different intensity level for the same class, see Fig. 5.2 for an example.

Empirical Performance Analysis: In this part, we evaluate both instance prediction accuracy and signal level prediction accuracy of our proposed method by varying different values of the sparse regularization parameter. In order to make a comparison, we propose a fully-supervised convolutive analysis dictionary learning (FSCDL). In FSCDL training, we feed the true instance label as the posterior probability of $y_n(t)$'s to the M-step, and we perform a gradient ascent updates of the weights in (5.3). $\bar{N}_n$ is set to be in $\{5, 10, 15, 20, 25, 30, 35, 40\}$, when we choose $T = 50$. We use 10 fold cross-validation with 80 training spectrograms and 20 independent test spectrograms to examine both instance prediction accuracy and the signal level prediction accuracy. To guarantee the boundedness of the solution, we add an L_2-regularization term $\lambda \sum_{ck} \|\mathbf{w}_{c.k}\|^2$ with $\lambda = 0.0001$ to the M-step. The discriminative dictionary by setting $C = 3$ learned in

both WSCDL approach and FSCDL approach are shown in the Fig. 5.3(a)-(e). The prediction accuracy for the one cluster dataset and two cluster dataset are shown in the Fig. 5.3(f)-(g).

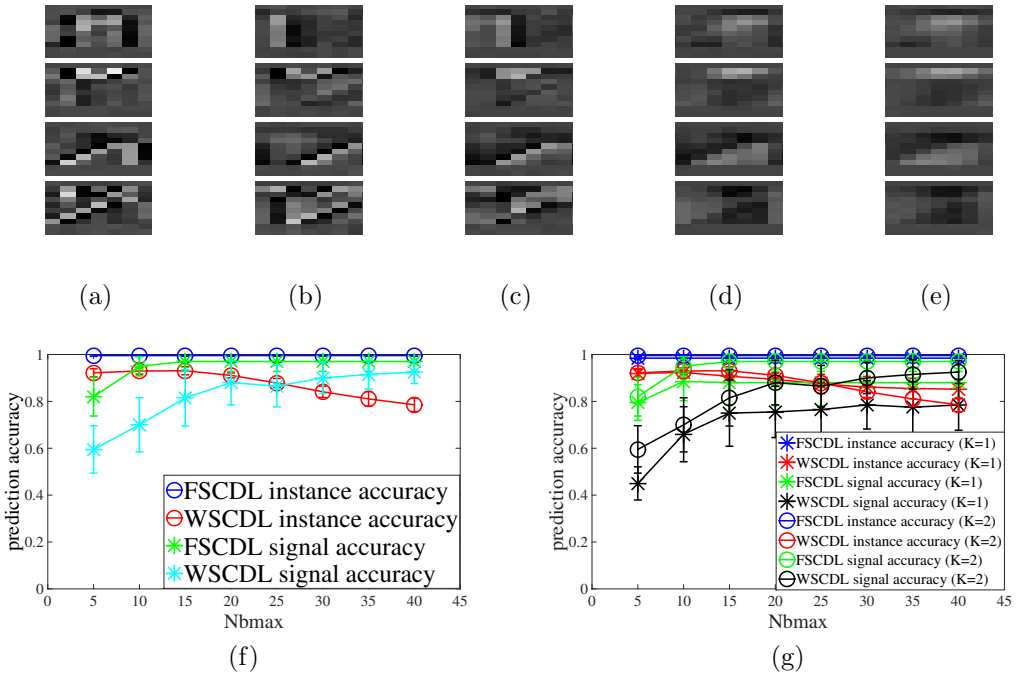


Figure 5.3: (a) FSCDL words; WSCDL words with (b) $\bar{N}_n = 5$; (c) $\bar{N}_n = 10$; (d) $\bar{N}_n = 25$; and (e) $\bar{N}_n = 40$. prediction accuracy for $B = 100$ and $T = 50$ (f) on one cluster dataset; and (g) on two cluster dataset with $K = 1$ and $K = 2$ (reproduction of [146]).

The results show that the WSCDL approaches FSCDL performance with a prediction accuracy that is only a little worse than the FSCDL in terms of instance and signal level prediction. When $\bar{N}_n$ is set to be too small, the signal level prediction accuracy drops. When we increase $\bar{N}_n$, the smearing effect will occur and the instance prediction accuracy drops. Additionally, the performance can be further optimized by tuning $\bar{N}_n$. See Figure 5.3(f) for an example. Figure 5.3(g) illustrates that using more analysis

words per class increases both instance level and signal level prediction accuracy for a two clusters dataset.

Chapter 6: Multiple-scaled weakly-supervised dictionary learning [1]

Dictionary atoms are usually designed to have the same dimension. However, in many scenarios patterns in data are naturally present at different scales such that it is necessary to allow dictionary atoms to be of different dimensions. For example, consider the spectrograms of sound events shown in Fig. 6.1. The sound pattern of throat chuckle and knock are short and occur multiple times, while the sound patterns of alert and phone rings are almost four times longer. When label information is provided, dictionary learning is tailored to perform well on the classification task. The previously proposed convolutive analysis dictionary learning with weak supervision [148] only consider a universal scale for the dictionary. The approach was applied to the acoustic scene analysis application. However, natural sound patterns or acoustic sound elements can vary in duration. To capture such patterns with a single scale, one would need to consider a large duration window to capture and differentiate the different patterns. This can results in over-fitting and potential increase in computational complexity. In this paper, we extend the model of You et al.'s [148] to accommodate for multiple scales and learn a multi-scale dictionary, which allows the intrinsic characteristics of a family of signals to be captured more efficiently.

[1] This chapter is a joint work with Dr. Raviv Raich, Xiaoli Fern and Jinsub Kim. This work was published as: Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly Supervised Learning of Multiple-Scale Dictionaries." *In 2018 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 100-104, 2018.

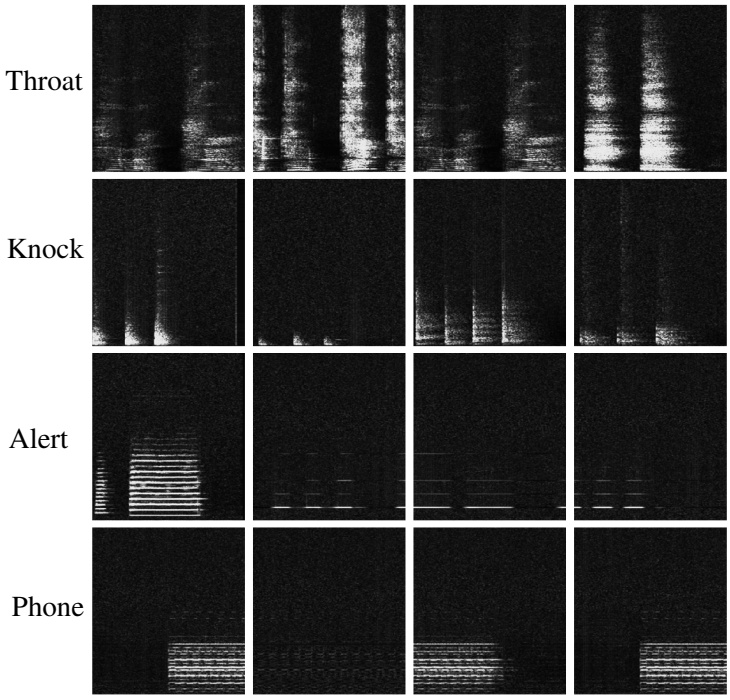


Figure 6.1: Spectrograms of acoustic sound events for different sound type. Sound events vary by the number of occurrences and the duration of each event from one class to another (reproduction of [149]).

6.1 The multi-scale model

Consider the convolution between two signals $x_n * w_c$, of lengths T_n and T_w respectively. The convolution can be formulated as converting signal x_n to a set of $T_n + T_w - 1$ vectors, each as a T_w windowed portion of the signal x_n. For simplicity, we assume T_w is odd and denote $\Delta = (T_w - 1)/2$. Using this notation, the convolution operation $w_c * x_n \mid_t$ can be replaced with $\mathbf{w}_c^T \mathbf{x}_{nt}$ as $x_n * w_c \mid_t = \sum_{\tau=-\Delta}^{\Delta} x_n(t-\tau)w(\tau) = \mathbf{w}_c^T \mathbf{x}_{nt}$, $\forall t = -\Delta, -\Delta+1, \ldots, T_n - 1 + \Delta$. Vector $\mathbf{x}_{nt} \in \mathbb{R}^{T_w}$ is defined as $\mathbf{x}_{nt} = [x_n(t+\Delta), x_n(t+\Delta-1), \ldots, x_n(t-\Delta)]^T$, and $\mathbf{w}_c \in \mathbb{R}^{T_w}$ is $\mathbf{w}_c = [w_c(-\Delta), w_c(-\Delta+1), \ldots, w_c(\Delta)]^T$.

We introduce the scale variable to allow a dictionary word to capture much longer patterns. For each class, we consider a dictionary word for each of the multiples scales $k = 1, 2, \ldots, K$ by introducing a dependence of the dictionary words on c and k, w_{ck}. For a particular scale k in class c, we formulate the convolution step as:

$$\mathbf{w}_{ck}^T \mathbf{M} \mathbf{x}_{nt}^k = \sum_{\tau=-\Delta}^{\Delta} w_{ck}(\tau) \sum_{\tau'=-k\Delta}^{k\Delta} M(\tau, \tau') x_n(t-\tau')$$

for $t = -\Delta, -\Delta+1, \ldots, T_n - 1 + \Delta$. $\mathbf{M}$ could be a subsampling matrix such that $M(\tau, \tau') = \mathbb{I}(\tau = k\tau')$ with k as the subsampling factor. Since subsampling may distorts the signal x_n, we use a random projection matrix here as $\mathbf{M} \in \mathbb{R}^{T_w \times kT_w}$ such that it transforms each kT_w windowed instance of signal x_n into a random vector of length T_w. Denote the transformed signal $x_n'^k$ with each T_w window portioned vector format as

$$\mathbf{x}_{nt}'^k = \mathbf{M} \mathbf{x}_{nt}^k = [m_1 * x_n \mid_t, m_2 * x_n \mid_t, \ldots, m_{T_w} * x_n \mid_t],$$

where each $m_i \in \mathbb{R}^{kT_w}$ is a random projection signal. We have $\mathbf{w}_{ck}^T \mathbf{M} \mathbf{x}_{nt}^k = \mathbf{w}_{ck}^T \mathbf{x}_{nt}'^k$. The aforementioned one-dimensional signal model can be extended to a two-dimensional

signal model (e.g., spectrograms) by following the approach in [148].

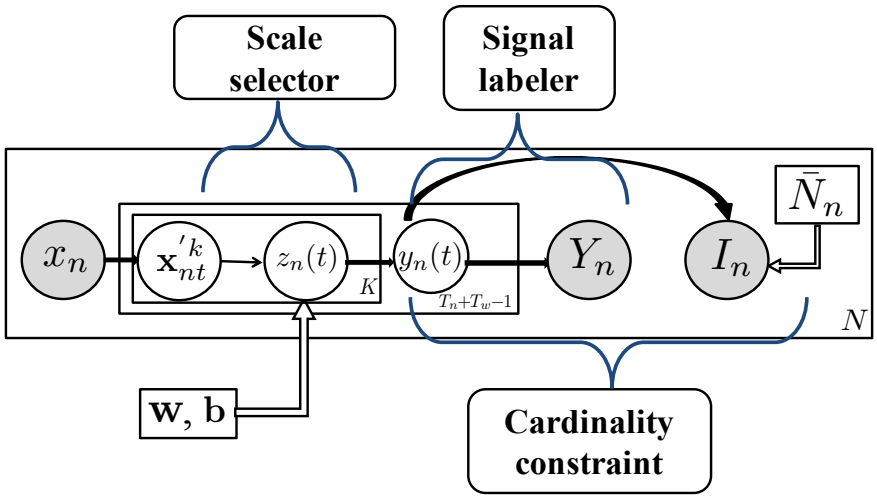


Figure 6.2: The probabilistic graphical model of MS-WSCADL (reproduction of [149]).

Probabilistic graphical model: To formulate a multi-scale weakly supervised dictionary learning, we introduce a probabilistic graphical model for multiple-scale weakly-supervised convolutive analysis dictionary learning (MS-WSCADL) in Fig. 6.2. To allow the dictionary words to operate at multiple scales on the observed signals, we introduce a time scale instance label signal. To index a specific scale within a class, we denote $z_n(t)$ as the time scale instance label signal. The probability of the scale instance label signal $z_n \in \{11, 12, \ldots, CK\}$ is modeled using multi-class logistic regression [96]:

$$P(z_n(t) = ck|\mathbf{x}_n; \mathbf{w}, \mathbf{b}) = \frac{e^{\mathbf{w}_{ck}^T \mathbf{x}_{nt}^k + b_{ck}}}{1 + \sum_{u=1}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{uv}^T \mathbf{x}_{nt}^v + b_{uv}}}. \tag{6.1}$$

Moreover, the instantaneous class label is given by

$$P(y_n(t) = c|\mathbf{x}_n; \mathbf{w}, \mathbf{b}) = \frac{\sum_{k=1}^{K} e^{\mathbf{w}_{ck}^T \mathbf{x}_{nt}^k + b_{ck}}}{1 + \sum_{u=1}^{C} \sum_{v=1}^{K} e^{\mathbf{w}_{uv}^T \mathbf{x}_{nt}^v + b_{uv}}}, \tag{6.2}$$

and

$$P(y_n(t)=0|\mathbf{x}_n;\mathbf{w},\mathbf{b})=\frac{1}{1+\sum_{u=1}^{C}\sum_{v=1}^{K}e^{\mathbf{w}_{uv}^T\mathbf{x}_{nt}^v+b_{uv}}}. \tag{6.3}$$

In the rest of the paper, we only consider y_n as the latent portion of the model. The probabilistic model for I_n and Y_n follows equations (4.2) and (4.3), respectively. Note that $\bar{N}_n$ is treated as a tuning (or hyper-) parameter of the graphical model. The smaller the constraint value $\bar{N}_n$ is, the sparser the label signal $y_n(t)$ becomes.

Complete and incomplete data likelihood: Since the multi-scale part is marginalized out in the model, the complete data likelihood follows exactly the same as the equation in (4.4) and the incomplete data likelihood follows the equation in (4.5). Calculating the incomplete data likelihood in (4.5) involves enumerating all possible instance labels, which is computationally intractable especially when the number of instance is large.

6.2 Solution approach

To resolve this, we consider an expectation maximization (EM) approach [80], following the similar procedure in the WSCADL model. However, the auxiliary function for the proposed multiple-scale model is changed to

$$\begin{aligned}Q(\boldsymbol{\theta},\boldsymbol{\theta}^i)=const.+\sum_{n=1}^{N}\sum_{t=-\Delta}^{T_n-1+\Delta}[\sum_{c=1}^{C}P(y_n(t)=c|\mathcal{D};\bar{N}_n,\boldsymbol{\theta}^i)\cdot\\ \log(\sum_{k=1}^{K}e^{\mathbf{w}_{ck}^T\mathbf{x}_{nt}^k+b_{ck}})-\log(1+\sum_{u=1}^{C}\sum_{k=1}^{K}e^{\mathbf{w}_{uk}^T\mathbf{x}_{nt}^k+b_{uk}})].\end{aligned}$$

To update dictionary words, we minimize the negative auxiliary function $-Q(\boldsymbol{\theta},\boldsymbol{\theta}^i)$. Since $-Q$ is a difference of convex functions, we apply the convex-concave procedure

(CCCP) [151] to surrogate for $-Q$ denoted by $\tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)$ as

$$\tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n-1+\Delta} [\log(1 + \sum_{u=1}^{C} \sum_{k=1}^{K} e^{\mathbf{w}_{uk}^T \mathbf{x}_{nt}^k + b_{uk}})$$
$$- \sum_{c=1}^{C} \sum_{k=1}^{K} P(y_n(t) = c|\mathcal{D}; \bar{N}_n, \boldsymbol{\theta}^i) \frac{e^{\mathbf{w}_{ck}^{iT} \mathbf{x}_{nt}^k + b_{ck}^i} (\mathbf{w}_{ck}^T \mathbf{x}_{nt}^k + b_{ck})}{\sum_{k=1}^{K} e^{\mathbf{w}_{uk}^{iT} \mathbf{x}_{nt}^k + b_{uk}^i}}].$$

Using the momentum method, we obtain the update rule as follows:

$$b_{ck}^{i+1} = b_{ck}^i - \gamma \frac{\partial \tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_{ck}} \mid_{\boldsymbol{\theta}=\boldsymbol{\theta}^i} + \eta(b_{ck}^i - b_{ck}^{i-1}),$$
$$\mathbf{w}_{ck}^{i+1} = \mathbf{w}_{ck}^i - \gamma \frac{\partial \tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_{ck}} \mid_{\boldsymbol{\theta}=\boldsymbol{\theta}^i} + \eta(\mathbf{w}_{ck}^i - \mathbf{w}_{ck}^{i-1}),$$

where γ is the learning rate and η is the momentum parameter, which can be obtained by exact line search. The gradient terms are:

$$\frac{\partial \tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial b_{ck}} = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n-1+\Delta} [P(z_n(t) = ck|\mathbf{x}_n; \mathbf{w}, \mathbf{b}) -$$
$$P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) \frac{e^{\mathbf{w}_{ck}^{iT} \mathbf{x}_{nt}^k + b_{ck}^i}}{\sum_{v=1}^{K} e^{\mathbf{w}_{cv}^{iT} \mathbf{x}_{nt}^v + b_{cv}^i}}], \tag{6.4}$$

and

$$\frac{\partial \tilde{Q}_{neg}(\boldsymbol{\theta}, \boldsymbol{\theta}^i)}{\partial \mathbf{w}_{ck}} = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n-1+\Delta} [P(z_n(t) = ck|\mathbf{x}_n; \mathbf{w}, \mathbf{b}) -$$
$$P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i) \frac{e^{\mathbf{w}_{ck}^{iT} \mathbf{x}_{nt}^k + b_{ck}^i}}{\sum_{v=1}^{K} e^{\mathbf{w}_{cv}^{iT} \mathbf{x}_{nt}^v + b_{cv}^i}}] \mathbf{x}_{nt}^k. \tag{6.5}$$

The momentum step in (6.4) and (6.5) requires calculating $P(y_n(t) = c|Y_n, x_n, I_n; \bar{N}_n, \boldsymbol{\theta}^i)$, which is one of the main challenges of EM inference on our proposed model. The Brute-force manner requires enumerating all possible values of $y_n(t)$ that results in an

exponential complexity with respect to the number of time instances. Since the brute-force approach is calculation prohibitive, we directly apply the efficient chain approach in [148].

6.3 The experimental results

In this part, we evaluate our proposed approach on a synthetic dataset. We compare our multi-scale approach with a uni-scale approach in [148] both on the synthetic data and a real-world data. In addition, we compare the proposed multi-scale approach with an alternative multi-scale approach on the real-world dataset.

Synthetic data analysis: We introduce the following two synthetic datasets:

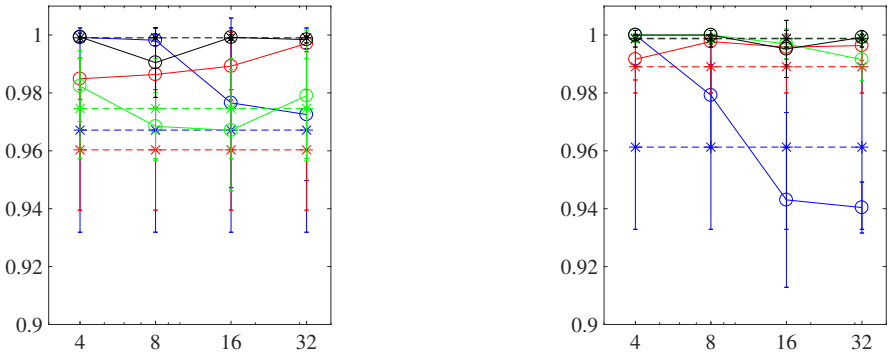
(i) **Uni-scale dataset:** we randomly generate $N = 50$ synthetic data with height (number of frequency bins) $F = 10$ and width (time frames) $T = 50$ using four different class shapes. The synthetic data are generated by randomly placing one of the predefined shapes into the corresponding position based on the instance label. The instance label for each spectrogram is generated from a Dirichlet distribution with high prior on the zero class. To make the synthetic dataset close to the real-world dataset, each spectrogram contains some phenomenon of overlapping with different classes and each syllable shape is at different intensity level for the same class.

(ii) **Multi-scale dataset:** We use four different scales on the predefined class 1 shape as four different classes. The scales are picked as $\{1, 2, 4, 8\}$ times of the original scale of class 1. The data are generated followed by the same procedure as previously stated.

Experimental setting: To evaluate our proposed approach, we trained our model both on the uni-scale and multi-scale dataset. We first separated the dataset into 40 sig-

nals for the training and the remaining independent 10 signals for the validation. We used 5 Monte-Carlo (MC) runs to tune the regularization term $\lambda \in \{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{0}, 10^{2}\}$ and the cardinality constraints $\bar{N}_1 = \bar{N}_2 = \ldots = N_N \in \{5, 10, 20, 50, 80, 100\}$. We evaluated the average signal label prediction accuracy on each pair of parameter value with the iteration number set to 5000. Using cross-validation for prediction accuracy, we found that $\lambda = 10^{-4}$ and $\bar{N}_n = 5$ produces the highest classification accuracy and used those to present the prediction performance.

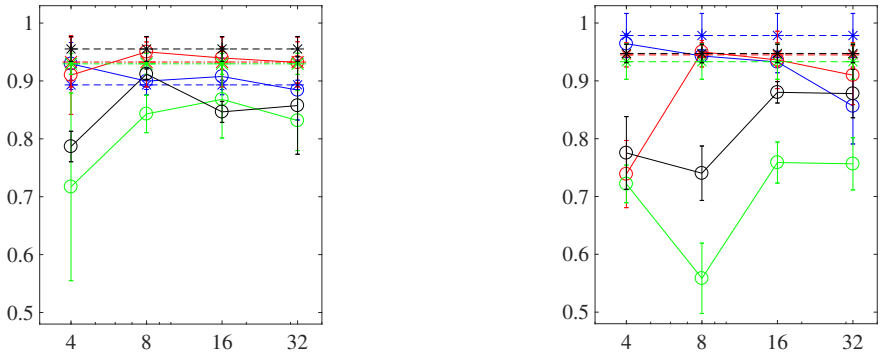
For both uni-scale and multi-scale dataset, we compare our proposed multi-scale learning algorithm with the uni-scale algorithm in [148] with various window size $T_w \in \{4, 8, 16, 32\}$ and multiple scales $K \in \{1, 2, 3, 4\}$. We evaluate the performance on both algorithms for both datasets with the 5 MC training output on the additional 50 test signals.



(a) instance label AUC (b) signal label AUC

Figure 6.3: On uni-scale dataset: comparison of performance (AUCs) between uni-scale with various T_w and multi-scale algorithm for $T_w = 4, K = [1, 2]$. (∘—: uni-scale algorithm; ⋆- - - -·: multi-scale algorithm; **Blue, red, green, black**: class 1,2,3,4) (reproduction of [149]).

Synthetic results: We evaluate both approaches in terms of instance level and



(a) instance label AUC

(b) signal label AUC

Figure 6.4: On multi-scale dataset: comparison of performance between uni-scale and multi-scale algorithm on various T_w (reproduction of [149]).

signal level detection accuracy by measuring area under the receiver operating characteristic curve (AUC). As Fig. 6.3 shows on the uni-scale data, the uni-scale approach produces highest AUC in window size $T_w = 4$ for almost all four classes, and the multi-scale approach performs similarly for class 3 and 4 and a little worse on class 1 and 2. However, as Fig. 6.4 shows on the multi-scale data, the uni-scale algorithm performs differently for each class with various window size ($T_w = 4$ produces highest AUC for class 1, $T_w = 8$ for class 2 and $T_w = 16$ for class 3 and 4). While the multi-scale algorithm outperforms the uni-scale algorithm significantly for all four classes, especially for class 3 and 4. To detect various scale patterns for the uni-scale algorithm, we have to pick a large enough window size that covers almost the largest scale. The uni-scale algorithm training time grows linearly as the size of window size in Tab. 6.1, therefore, picking a large window size is not only costly but also results in overfitting. However, the multi-scale algorithm accommodates the problem by various scales in a comparable training time. The running time of the multi-scale algorithm is between the running time of $T_w = 4$ and $T_w = 8$ of the uni-scale algorithm, which is 2.5 times faster than the

uni-scale algorithm picking window size of 16.

T_w	uni-4	uni-8	uni-16	uni-32	multi-4
time	266.1±9.8	553.5±19.5	1071.2±40.4	3505.0±178.5	419.3±15.5

Table 6.1: Training time (in s) for various window size in the uni-scale algorithm and the multi-scale algorithm with $T_w = 4, K = [1, 4]$.

Real-world data analysis: To evaluate the multi-scale algorithm on the real-world applications, we choose the office live test dataset on the AASP challenge [38]. This dataset consists of seven roughly three minutes long sound recording of 16 different classes on the live office scenario. In this experiment, our task is to detect the presence and absence of class events on the new recording.

Data preprocessing and experimental setting: We first chunk the recordings into 10s chunks and transform them to spectrograms. We apply the noise whitening procedure [14] and the down-sampling from $\mathbb{R}^{707\times6120}$ to $\mathbb{R}^{256\times200}$. To compare our proposed multi-scale approach with the uni-scale approach, we first tune the uni-scale approach with the window size $T_w \in \{10, 20, 50, 100\}$, the cardinality constraint $\bar{N}_n \in \{10, 20, 40, 60, 100, 160, 200\}$ along with a regularization term $\lambda \in \{10^{-6}, 10^{-4}, 10^{-2}, 10^{-1}, 1, 10\}$. After we obtained the optimal parameter value $T_w = 20, \bar{N}_n = 10, \lambda = 10^{-4}$. We heuristically choose a window size of 10, $\bar{N}_n = 10, \lambda = 10^{-4}$ and use scale of $1, 4$ for our multi-scale approach. After trained on the 126 spectrograms of the data, we evaluate a detection performance for both multi-scale approach and uni-scale approach on the rest 31 spectrograms.

Competing algorithm: To the best of our knowledge, since there is no state-of-the-art algorithm for multi-scale weakly-supervised dictionary learning, we combine a generative dictionary learning approach in [104] as sparse coding and a spatial pyramid matching (SPM) algorithm in [138, 139] for scale-invariant as a competing algorithm.

Although the algorithm is in the fully-supervision setting, we can still detect the presence and absence of a given class in the signal label by using C linear-SVM classifiers.

Results on real-world data: The result in Tab. 6.2 shows that our proposed multi-scale approach outperforms the uni-scale approach on almost all classes in terms of instance level AUCs, especially for *throat* and *switch* classes. The proposed multi-scale algorithm performances similarly to uni-scale algorithm in terms of signal level AUCs. Since the office live sound dataset contains various length of recurring patterns, the uni-scale approach may not be suitable to capture and classify those sound events. Our proposed multi-scale approach can detect those short patterns and long patterns at the same time. In addition, both multi-scale and uni-scale algorithms outperforms the competing SPM algorithm, and the SPM algorithm are not designed to provide the instance level classification task. We suspect the weakness of the competing SPM algorithm is due to non-trivial combination of the two approaches and the combined approach is designed with full-supervision, instead of weak-supervision setting. Further research is needed to improve the performance, which is beyond the scope of this paper.

Class	uni-ins.	multi-ins.	uni-sig.	multi-sig.	SPM-sig.
throat	50.21±12.17	**67.90±17.14**	74.88±4.01	**82.48±12.08**	39.83±14.00
knock	52.55±13.93	**57.84±17.14**	70.61±12.12	**77.58±13.86**	40.13±13.79
pageturn	52.98±5.64	**61.94±10.34**	**79.09±16.45**	77.89±6.76	56.10±11.54
phone	46.25±20.15	**52.84±10.85**	**65.95±22.82**	63.06±17.04	39.70±14.75
switch	46.33±3.57	**65.52±11.75**	**88.08±8.43**	84.24±6.80	46.95±24.72
average	54.75±10.60	**58.37±6.66**	**70.19±13.09**	68.62±13.77	43.88±3.64

Table 6.2: List 5 class example and the average across all 16 classes of instance level and signal detection AUCs(%) for both approaches.

Chapter 7: Conclusion and Future research

7.1 Summary

In this work, we presented both **generative** and **discriminative** convolutive dictionary learning approaches for time-series analysis. In particular, in the proposed generative dictionary learning, we formulated a convolutive dictionary learning objectives to efficiently extract dictionary and sparse activations from time-series data. This approach combined the power of estimating spectra-temporal patterns given by the convolutive model and the computational complexity savings associated with the random projection approach. Additionally, we addressed the boundary effect arising in a collection of discontinuous times-series data. Furthermore, we introduced a step-size selection criterion to improve convergence rate when updating the activations and the dictionary words. Real-world results suggested that the proposed approach can be used to efficiently represent bird audio recordings and to solve denoising, syllable discovering and classification problems.

In discriminative dictionary learning, we developed a novel probabilistic model that aims to learn 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 agiven 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 was 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 outperformed the generative approach. Additionally, we showed that the proposed approach yielded 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.

When the data is rich in class patterns (contains multiple cluster patterns for a class), the extension of the multiple-cluster approach is needed. Therefore, we considered a multiple dictionary words per class to accommodate for classes with multiple modes. The experimental esults sroh what thetmodel with multiple cluste- componentrperformed better than the one that does not use the clustering component. Since the patterns in the time-series data are not always in the same scale, the weak-supervised dictionary learning is not able to capture the difference duration scales of the fundamental patterns in the time-series data analysis. We proposed a learning strategy to adapt different scales among classes in the analysis dictionary under the weak-supervision setting. We showed our approach outperformed the uni-scale approach and a competing approach in the multi-scale dataset and real-world dataset.

7.2 The contributions of the work

Our contributions are as follows:

(A) For the generative dictionary learning:

(i) We first developed a random projected convolutive dictionary learning approach to extract patterns from time-series data [103, 104].

(ii) We derived a set of iterations with a choice of step-size that guarantees monotonically decreasing objective [103, 104].

(iii) We presented an *application* of the proposed approach for (1) denoising spectrograms of in-situ recordings of bird songs, which are corrupted by rain noise, (2) unsupervised bird syllable discovery and (3) supervised classification of birdsong recordings [103, 104].

(B) For the discriminative convolutive dictionary learning:

(i) We developed a novel discriminative probabilistic model for analysis dictionary learning under the *weak-supervision* setting [148].

(ii) We used an alternative approach for cardinality (or sparsity) constraints as implicit observations in a graphical model as opposed to commonly used norm regularization [148]. This approach allowed for localization of the patterns-of-interest.

(iii) We introduced a novel framework for efficient message passing using a reformulation of the proposed graphical model both as a chain and as a tree [148]. This reformulation yields a near-linear exact probability calculation that alleviates the need for approximate inference.

(iv) We extended the weakly-supervised dictionary learning model to learn multiple dictionary clusters per class [146].

(v) We developed a multiple-scale dictionary learning model under the weak-supervision setting [149]. Results indicate that multi-scale dictionaries can improve classification performance.

7.3 List of Publications

Below is a list of publications associated with this dissertation:

Journal papers

1. Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly supervised dictionary learning." *IEEE Transactions on Signal Processing* 66, no. 10 (2018): 2527-2541.

2. Ruiz-Muñoz, José Francisco, Zeyu You, Raviv Raich, and Xiaoli Z. Fern. "Dictionary learning for bioacoustics monitoring with applications to species classification." *Journal of Signal Processing Systems* 90, no. 2 (2018): 233-247.

Conference papers

3. Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly Supervised Learning of Multiple-Scale Dictionaries." *In 2018 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 100-104, 2018.

4. Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Discriminative recurring signal detection and localization." *In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2377-2381. IEEE, 2017.

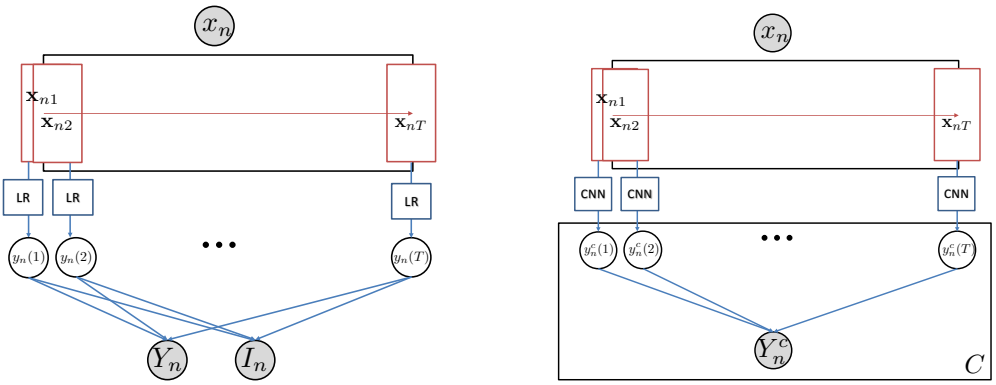
5. Zeyu You, Raviv Raich, Xiaoli Z. Fern, and Jinsub Kim. "Weakly-supervised analysis dictionary learning with cardinality constraints." *In 2016 IEEE Statistical*

Signal Processing Workshop (SSP), pp. 1-5. IEEE, 2016.

6. Ruiz-Muñoz, José Francisco, Zeyu You, Raviv Raich, and Xiaoli Z. Fern. "Dictionary extraction from a collection of spectrograms for bioacoustics monitoring." *In 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1-6. IEEE, 2015.

7.4 Future research

Chapters 4-6 introduced and developed modeling and inference methods for weakly-supervised learning of time-series. Throughout, a linear model was used for the probabilistic model of the instance level label, i.e., a logistic regression model. In particular, a linear model in the raw-data time series. However, in some of the real-world applications, linear classifier applied to raw time-series data may yield sub-optimal results in terms of classification performance. In the past decade, learning using deep neural networks, has become the golden standard for many applications including object recognition in computer vision [105], sound event recognition in audio analysis [119], and speaker recognition in speech analysis [108]. For many of the aforementioned application areas, the replacement of carefully crafted domain specific features with deep network featurization was demonstrate to yield significant improvements in terms of classification performance. The formulation we considered in this work, lends itself to a natural extension to the deep network featurization. This is accomplished by replacing the linear features in our model, e.g., $\mathbf{x}_{nt}$, with a nonlinear transformation $\mathbf{g}(\mathbf{x}_{nt})$ obtained by a deep net suited for the application domain.



(a) The original model (b) The deep learning model

Figure 7.1: A systematic plot of the weak-supervised learning models: (a) The original graphical model, (b) The deep learning model

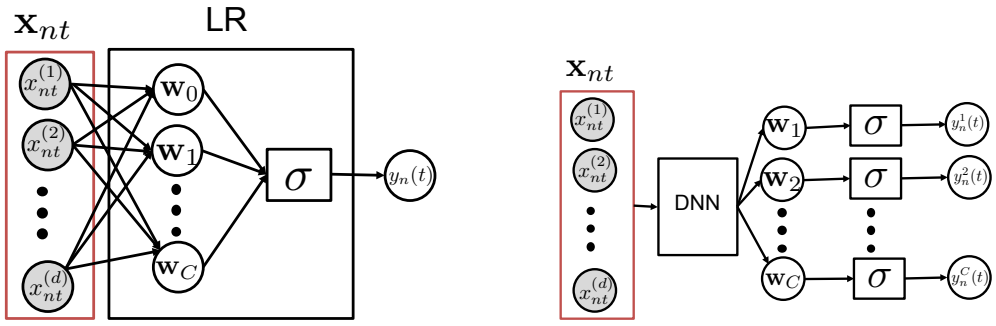
7.4.1 Preliminary idea

In Chapters 4-6, we introduced a weakly-supervised learning framework for time-series data, where each time-series signal x_n is going through a linear system that produces a set of scores for determining the probability of the time-instance multi-class labels $y_n(t) \in \{0, 1, \ldots, C\}$ as shown in Figure 7.1(a). However, in real-world applications, data examples are often not linearly separable. To increase classification performance when data are more complex, we incorporate a deep learning model (such as CNN) into the proposed weak-supervision learning system. The main idea is illustrated in Figure 7.1(b).

Time-instance labeler: The original time-instance labeler in (3.1) shown in Figure 7.2(a) is a linear model that maps the input time-series signal into a probabilistic scoring of determining a multi-class label $y_n(t)$ for each windowed time-instance. Here, we propose using a deep learning model as shown in Figure 7.2(b) in place of the linear model. As a preliminary effort to test the proposed approach, we consider

a simpler graphical model for the labeling mechanism. We replace the original multi-class single instance label assumption $y_n(t) \in \{0, 1, \ldots, C\}$ with a binary multi-label $y_n(t) = [y_n^1(t), y_n^2(t), \ldots, y_n^C(t)]$ where each $y_n^c(t) \in \{0, 1\}$ represents the binary decision corresponding to the presence and absence of a particular class c at the tth time window. This change allows us to have a separate signal level decision for each class thereby simplifying the objective to minimize.

Signal labeler: Instead of using a union assumption in (A.3) of chapter 4 for combining the instance labels $y_n(t)$ into a signal label Y_n, we consider using an OR rule such that the probability of the signal label $P(Y_c^i = 1)$ for indicating the class c being present is $P(Y_c^i = 1) = 1 - \prod_t P(y_n^c(t) = 0)$. Additionally, we omit the sparsity regularization term for simplicity. Note that the resulting probability is expressed in closed-form in terms of the instance level probabilities avoiding the marginalization used in the model of Chapters 4-6.



(a) The original model (b) The deep learning model

Figure 7.2: The time-instance labeler models: (a) The original graphical model, (b) The deep learning model

7.4.2 Preliminary results

To examine the proposed idea, we first considered the toy example problem of detecting the presence or absence of a target digit in 0-9 in a hand written sequence of digits. Moreover, we want to evaluate the capability of our method to localize the target digit in the sequence. To examine the applicability of the idea real-world applications, we considered applying the method to the HJA dataset as described in Chapter 4 Section 4.6.3.

7.4.2.1 MNIST results

Data Generation: We created images of a sequence of digits by placing $m \in [1, 10]$ hand-written digits (28×28 dimensions) from the MNIST dataset [25] into a 28×280 dimensional blank image as follows. (1) First, we generate the center index locations as $\mathcal{L} = \{l_k | l_k = k * 28 + 14, k = 0, 1, \ldots, 9\}$; (2) We then generate each signal label Y_n using a Bernoulli distribution; (3) For the generation of a negative example x_n that does not contain a target digit $Y_n = 0$, we randomly select m locations from the location indexes $\mathcal{L}$ and place the randomly selected m non-target digits into the corresponding locations plus an offset $p \sim \mathcal{U}[-10, 10]$; (4) For a positive example x_n that contains target digit $Y_n = 1$, we pick a positive event number $n \in [1, \min(4, m)]$ and place n randomly selected target digits into n randomly selected locations from $\mathcal{L}$ plus an offset p. The remaining $m - n$ randomly selected non-target digits are placed into the corresponding $m - n$ remaining locations plus an offset p. Note that the overlapping digits will add the intensities from each digits, since placing each digit into the blank image is additive.

Methods: We implemented the proposed approach using a linear model with only 1 dense layer and a CNN model with 3 layers (2 convolutional layers and 1 dense layer). The

CNN model is shown in Figure 7.3. The loss function is designed using a cross entropy loss between the signal label Y_n and its probability $P(Y_n) = (1 - \prod_{t=1}^{T}(1 - p(y_n(t) = 1))^{Y_n}(\prod_{t=1}^{T}(1 - p(y_n(t) = 1))^{1-Y_n}$, where $p(y_n(t) = 1)$ is modeled by the aforementioned CNN model.

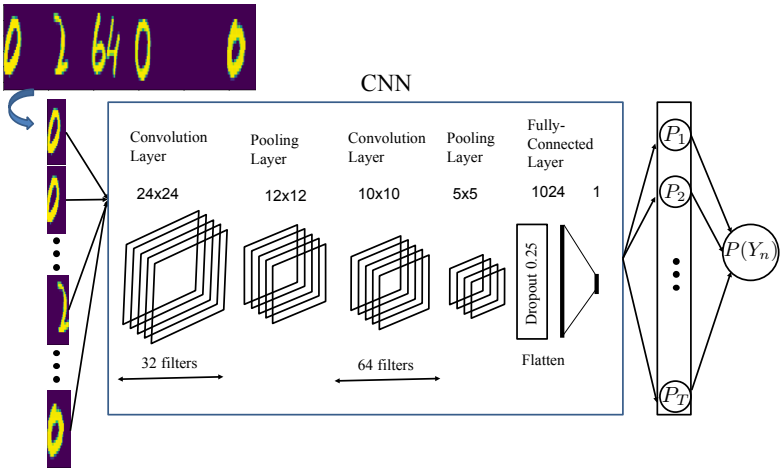
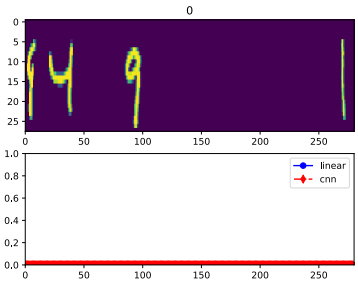


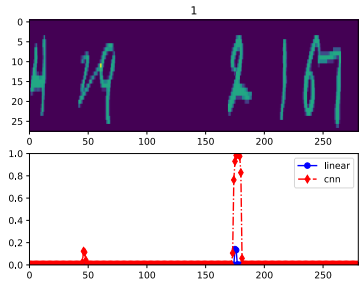
Figure 7.3: The deep learning model

Results: First, we consider a qualitative assessment of the proposed approach by examining the capability of the approach to resolve the location of target digits in the digit-sequence image and comparing it to a linear model. Figure 7.4 shows the instance level probability as a function position (i.e., instance index) using both the linear model and CNN model along side the digit-sequence image example. In the plot, both positive image examples that contain the target digit of 2 and negative image examples that do not contain the target digit are examined. We observe that in some cases, the linear model fails to localize and detect the target number while CNN model correctly localize and detect the target digit 2.

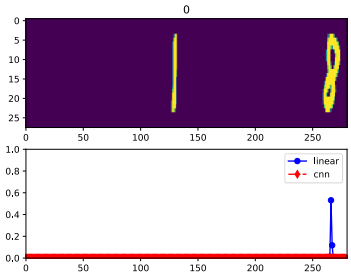
Next, we examine the approach quantitatively by evaluating classification performance at both instance-level and signal-level for both models. To prevent the repeated



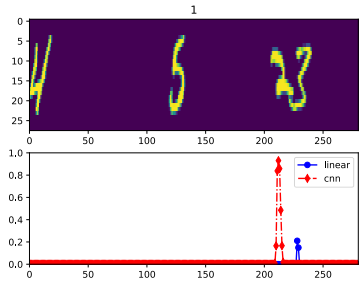
(a) Negative example 1



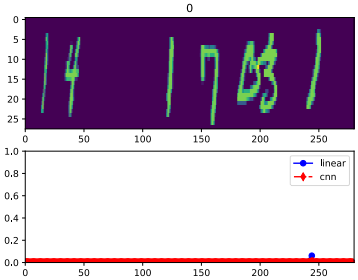
(b) Positive example 1



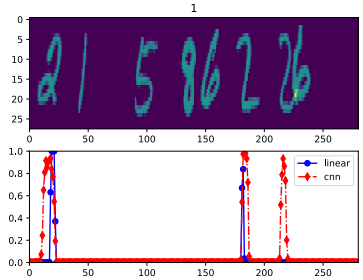
(c) Negative example 2



(d) Positive example 2



(e) Negative example 3



(f) Positive example 3

Figure 7.4: 3 examples of prediction on MNIST data in the weakly-labeled setting with the linear model and the CNN model.

counting for each of the positive events (containing the target digit 2), the instance-level predictions are generated using the following mechanism: (1) For each of the positive events (instance-level ground truth with positive label), we use a sub-window with length of 11 centered at its location index l and measure the predicted probability by $\max\{P(y_n(l-5)), P(y_n(l-4)), \ldots, P(y_n(l+5))\}$. (2) The corresponding ground truth is converted from $\{y_n(l-5)=0, y_n(l-4)=0, \ldots, y_n(l)=1, y_n(l+1)=0, \ldots, y_n(l+5)\}$ to a single label $y_n(l)=1$.

To show the effect on different window size, we choose a various window size in $\{14, 28, 56\}$ with 2000 training examples generated from the MNIST training set and the the performance on 10 independent realizations of 2000 examples generated from the MNIST test sets. Instance-level and signal-level performance evaluation metrics (average±standard deviation) for various window sizes are provided in Table 7.1.

Metric	14-ins	14-sig	28-ins	28-sig	56-ins	56-sig
Linear model						
Accuracy	99.88±00.02	93.89±00.35	99.89±00.03	92.75±00.48	99.72±00.02	92.63±00.59
AUC	97.60±00.37	97.62±00.27	97.17±00.21	97.12±00.42	95.56±00.34	96.61±00.36
F1	81.35±01.69	93.71±00.39	81.62±00.77	92.36±00.61	64.47±01.61	92.40±00.63
CNN model						
Accuracy	99.95±00.00	97.73±00.35	**99.96±00.00**	**98.44±00.24**	99.96±00.00	98.01±00.25
AUC	99.98±00.00	99.71±00.08	**99.99±00.02**	**99.84±00.06**	99.93±00.02	99.76±00.08
F1	92.30±01.08	97.71±00.35	**94.71±00.31**	**98.43±00.25**	93.79±00.54	98.00±00.26

Table 7.1: Performance results (%) on the MNIST sequenced data for various window size by using 0/1 signal-labels in the training.

The results show that the window size of 28 produces the highest average values in terms of accuracy and F1 for linear model. The window size of 28 produces highest average values for all metric in CNN model. When the window size is small, it may not contain the entire digit. When the window size is large, multiple segments of digits may appear within a window. In addition, the CNN model of window size 28 has the highest average values in terms of accuracy, AUC and F1 scores. The performance gain

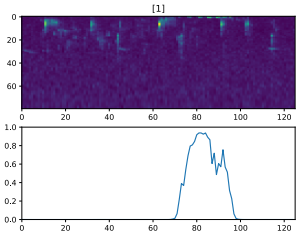
of instance-level F1 score is increased from 81% with the linear model to 94% using the CNN model.

7.4.2.2 HJA results

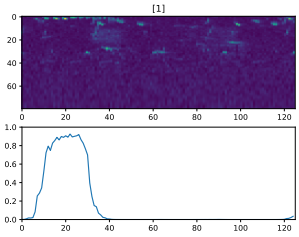
In Chapter 4 subsection 4.6.3, we have introduced the HJA bioacoustic dataset. In this subsection, we use this dataset for performance evaluation. We use the same deep learning architecture as the one in Figure 7.3 to examine the performance of the proposed idea on the HJA dataset. Since the HJA data contains 13 different bird species (i.e., classes), we test the classification of each class as a separate binary classification problem. The bag label probability of presence or absence of each class is obtained from the presence or absence of the class at each time instance (independent of other classes). We consider using a qualitative assessment by examining the capability of the approach to resolve the location of each bird species as in Figure 7.5.

Figure 7.5 shows 10 positive examples of 5 out of the 13 classes. For each class, two spectrograms (containing bird chirps from the target class) are shown. Additionally, instance-level prediction probabilities are displayed below each of the spectrograms. The results show that for classes such as Brown Creeper and Red-Breasted Nuthatch, their class chirps are correctly detected and localized. For some classes such as Varied Thrush and Hammonds Flycatcher, not all chirps are identified, some chirps seem to have low detection probabilities. Other classes such as Olive-sided Flycatcher, the chirps are not detected that flat low probabilities are observed.

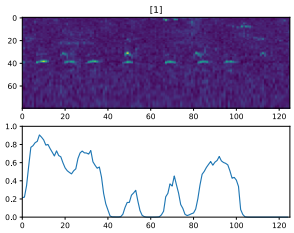
Figure 7.6 shows 6 examples of multi-labeled spectrograms and corresponding prediction probabilities over the 13 classes. The results show a subset of the classes are correctly detected and localized whenever high probabilities are observed around their



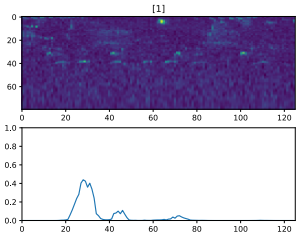
(a) Brown Creeper exp. 1



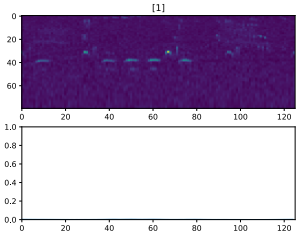
(b) Brown Creeper exp. 2



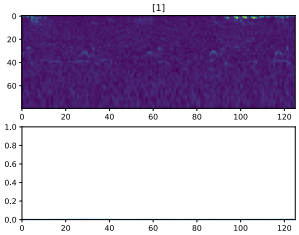
(c) Red-Breasted Nuthatch exp. 1



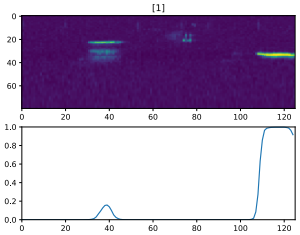
(d) Red-Breasted Nuthatch exp. 2



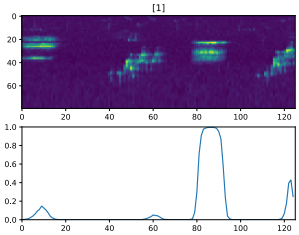
(e) Olive-sided Flycatcher exp. 1



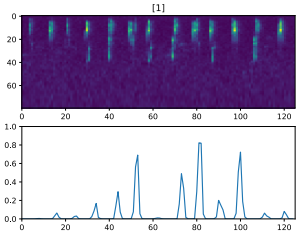
(f) Olive-sided Flycatcher exp. 2



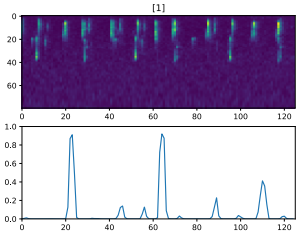
(g) Varied Thrush exp. 1



(h) Varied Thrush exp. 2

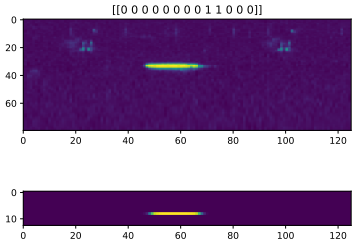


(i) Hammonds Flycatcher exp. 1

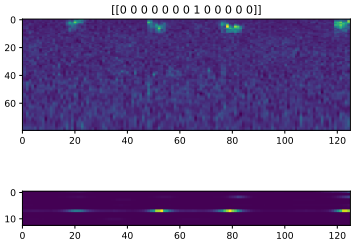


(j) Hammonds Flycatcher exp. 2

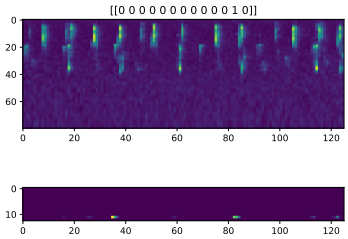
Figure 7.5: HJA data prediction on selected classes in the binary scenario.



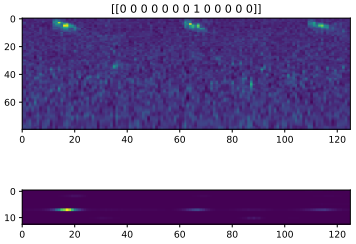
(a) Example 1



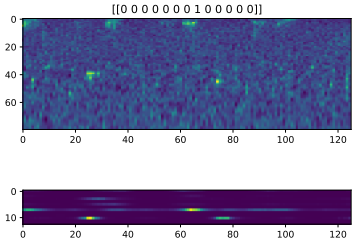
(b) Example 2



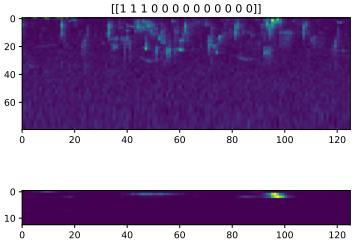
(c) Example 3



(d) Example 4



(e) Example 5



(f) Example 6

Figure 7.6: HJA data prediction in the multi-labeled scenario.

class bird chirps.

Both the binary labeled scenario and multi-labeled scenario results show some of the classes can not be correctly detected and localized that is due to several potential reasons: (1) Training data is limited, (2) Low signal-to-noise ratio, (3) Different duration for different classes, and (4) The model is not complex enough to discriminate between different classes. In the future, we will focus on addressing these issues in order to increase the classification performance. The ideas involves the following components: (1) Find data augmentation schemes that are suitable for bioacoustics data; (2) Apply noise reduction techniques or dimension reduction to the spectrogram; and (3) Search over different hyper-parameters of the window size, the network size, or the architectures.

Bibliography

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11):4311–4322, 2006.

[2] Witali Aswolinskiy, René Felix Reinhart, and Jochen Steil. Time series classification in reservoir-and model-space. *Neural Processing Letters*, 48(2):789–809, 2018.

[3] Behnam Babagholami-Mohamadabadi, Amin Jourabloo, Mohammadreza Zolfaghari, and Mohammad T Manzuri Shalmani. Bayesian supervised dictionary learning. In *UAI Application Workshops*, pages 11–19. Citeseer, 2013.

[4] Roland Badeau and M Plumbley. Multichannel high resolution NMF for modelling convolutive mixtures of non-stationary signals in the time-frequency domain. *Transactions on Audio, Speech and Language Processing*, 22(11):1670–1680, 2013.

[5] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[6] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[7] Debrup Banerjee, Kazi Islam, Gang Mei, Lemin Xiao, Guangfan Zhang, Roger Xu, Shuiwang Ji, and Jiang Li. A deep transfer learning approach for improved post-traumatic stress disorder diagnosis. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 11–20. IEEE, 2017.

[8] Richard Baraniuk. Compressive sensing. *IEEE signal processing magazine*, 24(4), 2007.

[9] Daniele Barchiesi and Mark D Plumbley. Dictionary learning of convolved signals. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5812–5815. IEEE, 2011.

[10] Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems*, pages 899–907, 2013.

[11] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[12] Daniel T. Blumstein, Daniel J. Mennill, Patrick Clemins, Lewis Girod, Kung Yao, Gail Patricelli, Jill L. Deppe, Alan H. Krakauer, Christopher Clark, Kathryn A. Cortopassi, Sean F. Hanser, Brenda McCowan, Andreas M. Ali, and Alexander N. G. Kirschel. Acoustic monitoring in terrestrial environments using microphone arrays: applications, technological considerations and prospectus. *Journal of Applied Ecology*, 48(3):758–767, 2011.

[13] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[14] Forrest Briggs, Xiaoli Z. Fern, and Raviv Raich. Rank-loss support instance machines for miml instance annotation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 534–542. ACM, 2012.

[15] Forrest Briggs, Balaji Lakshminarayanan, Lawrence Neal, Xiaoli Z Fern, Raviv Raich, Sarah J K Hadley, Adam S Hadley, and Matthew G Betts. Acoustic classification of multiple simultaneous bird species: a multi-instance multi-label approach. *The Journal of the Acoustical Society of America*, 131(6):4640–4650, June 2012.

[16] Forrest Briggs, Balaji Lakshminarayanan, Lawrence Neal, Xiaoli Z Fern, Raviv Raich, Sarah JK Hadley, Adam S Hadley, and Matthew G Betts. Acoustic classification of multiple simultaneous bird species: A multi-instance multi-label approach. *The Journal of the Acoustical Society of America*, 131(6):4640–4650, 2012.

[17] Peter J Brockwell, Richard A Davis, and Stephen E Fienberg. *Time Series: Theory and Methods: Theory and Methods*. Springer Science & Business Media, 1991.

[18] Aline Cabasson and Olivier Meste. Time delay estimation: a new insight into the woody's method. *IEEE signal processing letters*, 15:573–576, 2008.

[19] Kin-Pong Chan and Wai-Chee Fu. Efficient time series matching by wavelets. In *icde*, page 126. IEEE, 1999.

[20] Zhengping Che, Yu Cheng, Shuangfei Zhai, Zhaonan Sun, and Yan Liu. Boosting deep learning risk prediction with generative adversarial networks for electronic health records. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 787–792. IEEE, 2017.

[21] Huanhuan Chen, Fengzhen Tang, Peter Tino, Anthony G Cohn, and Xin Yao. Model metric co-learning for time series classification. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[22] Huanhuan Chen, Fengzhen Tang, Peter Tino, and Xin Yao. Model-based kernel for efficient time series analysis. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 392–400. ACM, 2013.

[23] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista. The ucr time series classification archive. 2015.

[24] Oana G Cula and Kristin J Dana. 3d texture recognition using bidirectional feature histograms. *International Journal of Computer Vision*, 59(1):33–60, 2004.

[25] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[26] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[27] Oliver Faust, Yuki Hagiwara, Tan Jen Hong, Oh Shu Lih, and U Rajendra Acharya. Deep learning for healthcare applications based on physiological signals: a review. *Computer methods and programs in biomedicine*, 2018.

[28] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *arXiv preprint arXiv:1809.04356*, 2018.

[29] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Evaluating surgical skills from kinematic data using convolutional neural networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 214–221. Springer, 2018.

[30] Tomoko G Fujii, Maki Ikebuchi, and Kazuo Okanoya. Auditory responses to vocal sounds in the songbird nucleus taeniae of the amygdala and the adjacent arcopallium. *Brain, behavior and evolution*, 87(4):275–289, 2016.

[31] Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Localizing objects with smart dictionaries. In *European Conference on Computer Vision*, pages 179–192. Springer, 2008.

[32] Mehrdad J Gangeh, Ahmed K Farahat, Ali Ghodsi, and Mohamed S Kamel. Supervised dictionary learning and sparse representation-a review. *arXiv preprint arXiv:1502.05928*, 2015.

[33] Mehrdad J. Gangeh, Ahmed K. Farahat, Ali Ghodsi, and Mohamed S. Kamel. Supervised dictionary learning and sparse representation-a review. *CoRR*, abs/1502.05928, 2015.

[34] Mehrdad J Gangeh, Ali Ghodsi, and Mohamed S Kamel. Dictionary learning in texture classification. In *International Conference Image Analysis and Recognition*, pages 335–343. Springer, 2011.

[35] Mehrdad J Gangeh, Ali Ghodsi, and Mohamed S Kamel. Kernelized supervised dictionary learning. *IEEE Transactions on Signal Processing*, 61(19):4753–4767, 2013.

[36] Yue Gao, Rongrong Ji, Wei Liu, Qionghai Dai, and Gang Hua. Weakly supervised visual dictionary learning by harnessing image attributes. *IEEE Transactions on Image Processing*, 23(12):5400–5411, 2014.

[37] Shaghayegh Gharghabi, Shima Imani, Anthony Bagnall, Amirali Darvishzadeh, and Eamonn Keogh. Matrix profile xii: Mpdist: A novel time series distance measure to allow data mining in more challenging scenarios. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 965–970. IEEE, 2018.

[38] Dimitrios Giannoulis, Emmanouil Benetos, Dan Stowell, Mathias Rossignol, Mathieu Lagrange, and Mark D Plumbley. Detection and classification of acoustic scenes and events: An ieee aasp challenge. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 1–4. IEEE, 2013.

[39] Ernst Haselsteiner and Gert Pfurtscheller. Using time-dependent neural networks for eeg classification. *IEEE transactions on rehabilitation engineering*, 8(4):457–463, 2000.

[40] Nima Hatami, Yann Gavet, and Johan Debayle. Classification of time-series images using deep convolutional neural networks. In *Tenth International Conference on Machine Vision (ICMV 2017)*, volume 10696, page 106960Y. International Society for Optics and Photonics, 2018.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[42] Qinghua Hu, Rujia Zhang, and Yucan Zhou. Transfer learning for short-term wind speed prediction with deep neural networks. *Renewable Energy*, 85:83–95, 2016.

[43] Sheng-Jun Huang and Zhi-Hua Zhou. Fast multi-instance multi-label learning. *arXiv preprint arXiv:1310.2049*, 2013.

[44] David R Hunter and Kenneth Lange. A tutorial on mm algorithms. *The American Statistician*, 58(1):30–37, 2004.

[45] Andrey Ignatov. Real-time human activity recognition from accelerometer data using convolutional neural networks. *Applied Soft Computing*, 62:915–922, 2018.

[46] Shima Imani, Frank Madrid, Wei Ding, Scott Crouter, and Eamonn Keogh. Matrix profile xiii: Time series snippets: A new primitive for time series data mining. In *2018 IEEE International Conference on Big Knowledge (ICBK)*, pages 382–389. IEEE, 2018.

[47] Atsushi Inoue and Mototsugu Shintani. Bootstrapping gmm estimators for time series. *Journal of Econometrics*, 133(2):531–555, 2006.

[48] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *science*, 304(5667):78–80, 2004.

[49] Maria G Jafari and Mark D Plumbley. Fast dictionary learning for sparse representations of speech signals. *IEEE Journal of Selected Topics in Signal Processing*, 5(5):1025–1031, 2011.

[50] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Learning a discriminative dictionary for sparse coding via label consistent k-svd. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1697–1704. IEEE, 2011.

[51] Zhuolin Jiang, Zhe Lin, and Larry S Davis. Label consistent k-svd: Learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2651–2664, 2013.

[52] Bela Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91–97, 1981.

[53] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286, 2001.

[54] Shu Kong and Donghui Wang. A dictionary learning approach for classification: separating the particularity and the commonality. In *European Conference on Computer Vision*, pages 186–199. Springer, 2012.

[55] Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Te-Won Lee, and Terrence J Sejnowski. Dictionary learning algorithms for sparse representation. *Neural computation*, 15(2):349–396, 2003.

[56] Gert R Lanckriet and Bharath K Sriperumbudur. On the convergence of the concave-convex procedure. In *Advances in neural information processing systems*, pages 1759–1767, 2009.

[57] Svetlana Lazebnik and Maxim Raginsky. Supervised learning of quantizer codebooks by information loss minimization. *IEEE transactions on pattern analysis and machine intelligence*, 31(7):1294–1309, 2009.

[58] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. Data augmentation for time series classification using convolutional neural networks. In *ECML/PKDD workshop on advanced analytics and learning on temporal data*, 2016.

[59] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[60] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International journal of computer vision*, 43(1):29–44, 2001.

[61] Xiao-Chen Lian, Zhiwei Li, Changhu Wang, Bao-Liang Lu, and Lei Zhang. Probabilistic models for supervised dictionary learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2305–2312, 2010.

[62] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Pranav Patel. Finding motifs in time series. In *Proc. of the 2nd Workshop on Temporal Data Mining*, pages 53–68, 2002.

[63] Jessica Lin, Rohan Khade, and Yuan Li. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, 39(2):287–315, 2012.

[64] Jessica Lin and Yuan Li. Finding approximate frequent patterns in streaming medical data. In *2010 IEEE 23rd International Symposium on Computer-Based Medical Systems (CBMS)*, pages 13–18. IEEE, 2010.

[65] Sangdi Lin and George C Runger. Gcrnn: Group-constrained convolutional recurrent neural network. *IEEE transactions on neural networks and learning systems*, (99):1–10, 2017.

[66] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

[67] Jason Lines, Sarah Taylor, and Anthony Bagnall. Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 1041–1046. IEEE, 2016.

[68] Chien-Liang Liu, Wen-Hoar Hsaio, and Yao-Chung Tu. Time series classification with multivariate convolutional neural network. *IEEE Transactions on Industrial Electronics*, 2018.

[69] Liping Liu and Thomas G Dietterich. A conditional multinomial mixture model for superset label learning. In *Advances in neural information processing systems*, pages 557–565, 2012.

[70] Qingju Liu, Wenwu Wang, Philip J B Jackson, Mark Barnard, Josef Kittler, and Jonathon Chambers. Source separation of convolutive and noisy mixtures using audio-visual dictionary learning and probabilistic time-frequency masking. *IEEE Transactions on Signal Processing*, 61(22):5520–5535, 2013.

[71] Qianli Ma, Lifeng Shen, Weibiao Chen, Jiabin Wang, Jia Wei, and Zhiwen Yu. Functional echo state network for time series classification. *Information Sciences*, 373:1–20, 2016.

[72] Julien Mairal, Francis Bach, and Jean Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012.

[73] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

[74] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R Bach. Supervised dictionary learning. In *Advances in neural information processing systems*, pages 1033–1040, 2009.

[75] Pankaj Malhotra, Vishnu TV, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Timenet: Pre-trained deep recurrent neural network for time series classification. *arXiv preprint arXiv:1706.08838*, 2017.

[76] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[77] Pierre-François Marteau. Time warp edit distance with stiffness adjustment for time series matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):306–318, 2009.

[78] Nijat Mehdiyev, Johannes Lahann, Andreas Emrich, David Enke, Peter Fettke, and Peter Loos. Time series classification using deep learning for process planning: A case from the process industry. *Procedia Computer Science*, 114:242–249, 2017.

[79] Roni Mittelman. Time-series modeling with undecimated fully convolutional neural networks. *arXiv preprint arXiv:1508.00317*, 2015.

[80] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[81] Frank Moosmann, Bill Triggs, and Frederic Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Twentieth Annual Conference on Neural Information Processing Systems (NIPS'06)*, pages 985–992. MIT Press, 2006.

[82] Alex Nanopoulos, Rob Alcock, and Yannis Manolopoulos. Feature-based classification of time-series data. *International Journal of Computer Research*, 10(3):49–61, 2001.

[83] Minh Hoai Nguyen, Lorenzo Torresani, Fernando De La Torre, and Carsten Rother. Weakly supervised discriminative localization and classification: a joint learning process. In *IEEE 12th International Conference on Computer Vision*, pages 1925–1932. IEEE, 2009.

[84] Henri J Nussbaumer. *Fast Fourier transform and convolution algorithms*, volume 2. Springer Science & Business Media, 2012.

[85] Henry Friday Nweke, Ying Wah Teh, Mohammed Ali Al-Garadi, and Uzoma Rita Alo. Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. *Expert Systems with Applications*, 2018.

[86] Tim Oates, Laura Firoiu, and Paul R Cohen. Clustering time series with hidden markov models and dynamic time warping. In *Proceedings of the IJCAI-99 workshop on neural, symbolic and reinforcement learning methods for sequence learning*, pages 17–21. Citeseer, 1999.

[87] Paul D. O'Grady and Barak A. Pearlmutter. Convolutive non-negative matrix factorisation with a sparseness constraint. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2006)*, pages 427–432, Maynooth, Ireland, September 2006.

[88] Paul D. O'Grady and Barak A. Pearlmutter. Discovering speech phones using convolutive non-negative matrix factorisation with a sparseness constraint. *Neurocomputing*, 72(1-3):88–101, 2008.

[89] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

[90] Alex S Park and James R Glass. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197, 2008.

[91] Florent Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Transactions on pattern analysis and machine intelligence*, 30(7):1243–1256, 2008.

[92] Gabriel Peyré and Jalal M Fadili. Learning analysis sparsity priors. In *Sampta'11*, pages 4–pp, 2011.

[93] Luke Pfister and Yoram Bresler. Learning sparsifying filter banks. In *SPIE Optical Engineering+ Applications*, pages 959703–959703. International Society for Optics and Photonics, 2015.

[94] Anh Pham, Raviv Raich, and Xiaoli Fern. Dynamic programming for instance annotation in multi-instance multi-label learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[95] Anh Pham, Raviv Raich, Xiaoli Fern, and Jesús P Arriaga. Multi-instance multi-label learning in the presence of novel class instances. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 2427–2435, 2015.

[96] Anh T Pham, Raviv Raich, and Xiaoli Z Fern. Simultaneous instance annotation and clustering in multi-instance multi-label learning. In *IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2015.

[97] Davood Rafiei and Alberto Mendelzon. Efficient retrieval of similar time sequences using dft. *arXiv preprint cs/9809033*, 1998.

[98] Deepta Rajan and Jayaraman J Thiagarajan. A generative modeling approach to limited channel ecg classification. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 2571–2574. IEEE, 2018.

[99] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3501–3508. IEEE, 2010.

[100] Saiprasad Ravishankar and Yoram Bresler. Learning sparsifying transforms. *IEEE Transactions on Signal Processing*, 61(5):1072–1086, 2013.

[101] Justin Romberg. Imaging via compressive sampling [introduction to compressive sampling and recovery via convex programming]. *IEEE Signal Processing Magazine*, 25(2):14–20, 2008.

[102] Ron Rubinstein, Tomer Peleg, and Michael Elad. Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model. *IEEE Transactions on Signal Processing*, 61.3 (2013)::661–677, 2013.

[103] JF Ruiz-Muñoz, Zeyu You, Raviv Raich, and Xiaoli Z Fern. Dictionary learning for bioacoustics monitoring with applications to species classification. *Journal of Signal Processing Systems*, pages 1–15, 2016.

[104] José Francisco Ruiz-Muñoz, Zeyu You, Raviv Raich, and Xiaoli Z Fern. Dictionary extraction from a collection of spectrograms for bioacoustics monitoring. In *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*, pages 1–6. IEEE, 2015.

[105] Philippe G Schyns. Object recognition: Complexity of recognition strategies. *Current Biology*, 28(7):R313–R315, 2018.

[106] Joan Serraa, Santiago Pascualb, and Alexandros Karatzogloua. Towards a universal neural network encoder for time series. *Artificial Intelligence Research and Development: Current Challenges, New Trends and Applications*, 308:120, 2018.

[107] Paris Smaragdis. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. In *Independent Component Analysis and Blind Signal Separation*, pages 494–499. Springer, 2004.

[108] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333. IEEE, 2018.

[109] Dan Stowell and Mark D. Plumbley. Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning. *PeerJ*, 2, 2014.

[110] Nils Strodthoff and Claas Strodthoff. Detecting and interpreting myocardial infarction using fully convolutional neural networks. *Physiological measurement*, 2018.

[111] Yoshiki Tanaka, Kazuhisa Iwamoto, and Kuniaki Uehara. Discovery of time-series motif from multi-dimensional data based on mdl principle. *Machine Learning*, 58(2-3):269–300, 2005.

[112] RK Tripathy and U Rajendra Acharya. Use of features from rr-time series and eeg signals for automated classification of sleep stages in deep neural network framework. *Biocybernetics and Biomedical Engineering*, 38(4):890–902, 2018.

[113] Munenori Uemura, Morimasa Tomikawa, Tiejun Miao, Ryota Souzaki, Satoshi Ieiri, Tomohiko Akahoshi, Alan K Lefor, and Makoto Hashizume. Feasibility of an ai-based measure of the hand motions of expert and novice surgeons. *Computational and mathematical methods in medicine*, 2018, 2018.

[114] Manik Varma and Andrew Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.

[115] Manik Varma and Andrew Zisserman. A statistical approach to material classification using image patch exemplars. *IEEE transactions on pattern analysis and machine intelligence*, 31(11):2032–2047, 2009.

[116] Ravichander Vipperla, Simon Bozonnet, Dong Wang, and Nicholas Evans. Robust speech recognition in multi-source noise environments using convolutive non-negative matrix factorization. *Proc. CHiME*, pages 74–79, 2011.

[117] Michail Vlachos, Jessica Lin, Eamonn Keogh, and Dimitrios Gunopulos. A wavelet-based anytime algorithm for k-means clustering of time series. In *In Proc. Workshop on Clustering High Dimensionality Data and Its Applications*. Citeseer, 2003.

[118] Tiep H Vu and Vishal Monga. Learning a low-rank shared dictionary for object classification. In *IEEE International Conference on Image Processing (ICIP)*, pages 4428–4432. IEEE, 2016.

[119] Chien-Yao Wang, Jia-Ching Wang, Andri Santoso, Chin-Chin Chiang, and Chung-Hsien Wu. Sound event recognition using auditory-receptive-field binary pattern and hierarchical-diving deep belief network. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(8):1336–1351, 2018.

[120] Dong Wang, Ravichander Vipperla, and Nicholas W D Evans. Online pattern learning for non-negative convolutive sparse coding accepted for publication. In *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication, August 28-31, Florence, Italy*, 2011.

[121] Dong Wang, Ravichander Vipperla, and Nicholas WD Evans. Online pattern learning for non-negative convolutive sparse coding. In *INTERSPEECH*, pages 65–68, 2011.

[122] Fei Wang and Ping Li. Efficient nonnegative matrix factorization with random projections. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 281–292. SIAM, 2010.

[123] Jingyuan Wang, Ze Wang, Jianfeng Li, and Junjie Wu. Multilevel wavelet decomposition network for interpretable time series analysis. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2437–2446. ACM, 2018.

[124] Shuqin Wang, Gang Hua, Guosheng Hao, and Chunli Xie. A cycle deep belief network model for multivariate time series classification. *Mathematical Problems in Engineering*, 2017, 2017.

[125] Xinggang Wang, Baoyuan Wang, Xiang Bai, Wenyu Liu, and Zhuowen Tu. Max-margin multiple-instance dictionary learning. In *ICML*, pages 846–854, 2013.

[126] Xinggang Wang, Zhuotun Zhu, Cong Yao, and Xiang Bai. Relaxed multiple-instance svm with application to object discovery. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1224–1232, 2015.

[127] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *Knowledge and Data Engineering, IEEE Transactions on*, 25(6):1336–1353, 2013.

[128] Zhiguang Wang and Tim Oates. Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[129] Zhiguang Wang and Tim Oates. Imaging time-series to improve classification and imputation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[130] Zhiguang Wang and Tim Oates. Spatially encoding temporal correlations to classify temporal data using convolutional neural networks. *arXiv preprint arXiv:1509.07481*, 2015.

[131] Zhiguang Wang, Wei Song, Lu Liu, Fan Zhang, Junxiao Xue, Yangdong Ye, Ming Fan, and Mingliang Xu. Representation learning with deconvolution for multivariate time series classification and visualization. *arXiv preprint arXiv:1610.07258*, 2016.

[132] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[133] John Winn, Antonio Criminisi, and Thomas Minka. Object categorization by learned universal visual dictionary. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1800–1807. IEEE, 2005.

[134] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2009.

[135] Daniel Wu, Ambuj Singh, Divyakant Agrawal, Amr El Abbadi, and Terence R Smith. Efficient retrieval for browsing large image databases. In *Proceedings of the fifth international conference on Information and knowledge management*, pages 11–18. Citeseer, 1996.

[136] Lin Wu, Yang Wang, and Shirui Pan. Exploiting attribute correlations: A novel trace lasso-based weakly supervised dictionary learning method. *IEEE Transactions on Cybernetics*, 2016.

[137] Jin Xie, Lei Zhang, Jane You, and David Zhang. Texture classification via patch-based sparse texton learning. In *IEEE International Conference on Image Processing*, pages 2737–2740. IEEE, 2010.

[138] Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009.

[139] Jianchao Yang, Kai Yu, and Thomas Huang. Supervised translation-invariant sparse coding. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3517–3524. IEEE, 2010.

[140] Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang. Fisher discrimination dictionary learning for sparse representation. In *International Conference on Computer Vision*, pages 543–550. IEEE, 2011.

[141] Meng Yang, Lei Zhang, Jian Yang, and David Zhang. Metaface learning for sparse representation based face recognition. In *2010 IEEE International Conference on Image Processing*, pages 1601–1604. IEEE, 2010.

[142] Lexiang Ye and Eamonn Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data mining and knowledge discovery*, 22(1-2):149–182, 2011.

[143] Chin-Chia Michael Yeh, Nickolas Kavantzas, and Eamonn Keogh. Matrix profile vi: Meaningful multidimensional motif discovery. In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 565–574. IEEE, 2017.

[144] Chin-Chia Michael Yeh and Yi-Hsuan Yang. Supervised dictionary learning for music genre classification. In *Proceedings of the 2Nd ACM International Conference on Multimedia Retrieval*, ICMR '12, pages 55:1–55:8, New York, NY, USA, 2012. ACM.

[145] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 1317–1322. IEEE, 2016.

[146] Zeyu You, Raviv Raich, Xiaoli Z Fern, and Jinsub Kim. Weakly-supervised analysis dictionary learning with cardinality constraints. In *2016 IEEE Statistical Signal Processing Workshop (SSP)*, pages 1–5. IEEE, 2016.

[147] Zeyu You, Raviv Raich, Xiaoli Z Fern, and Jinsub Kim. Discriminative recurring signal detection and localization. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2377–2381. IEEE, 2017.

[148] Zeyu You, Raviv Raich, Xiaoli Z Fern, and Jinsub Kim. Weakly supervised dictionary learning. *IEEE Transactions on Signal Processing*, 66(10):2527–2541, 2018.

[149] Zeyu You, Raviv Raich, Xiaoli Z Fern, and Jinsub Kim. Weakly supervised learning of multiple-scale dictionaries. In *2018 IEEE Statistical Signal Processing Workshop (SSP)*, pages 100–104. IEEE, 2018.

[150] Zeyu You, Raviv Raich, and Yonghong Huang. An inference framework for detection of home appliance activation from voltage measurements. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6033–6037. IEEE, 2014.

[151] Alan L Yuille, Anand Rangarajan, and AL Yuille. The concave-convex procedure (cccp). *Advances in neural information processing systems*, 2:1033–1040, 2002.

[152] Haichao Zhang, Yanning Zhang, and Thomas S Huang. Simultaneous discriminative projection and dictionary learning for sparse representation based classification. *Pattern Recognition*, 46(1):346–354, 2013.

[153] Min-Ling Zhang and Zhi-Hua Zhou. Multi-label learning by instance differentiation. In *AAAI*, volume 7, pages 669–674, 2007.

[154] Qiang Zhang and Baoxin Li. Discriminative K-SVD for dictionary learning in face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2691–2698. IEEE, 2010.

[155] Lu Huanzhang Chen Shangfeng Liu Junliang Zhao, Bendong and Dongya Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.

[156] Zhi-Hua Zhou and Min-Ling Zhang. Multi-instance multi-label learning with application to scene classification. In *Advances in neural information processing systems*, pages 1609–1616, 2006.

[157] Zhi-Hua Zhou, Min-Ling Zhang, Sheng-Jun Huang, and Yu-Feng Li. Multi-instance multi-label learning. *Artificial Intelligence*, 176(1):2291–2320, 2012.

[158] Yan Zhu, Zachary Zimmerman, Nader Shakibay Senobari, Chin-Chia Michael Yeh, Gareth Funning, Abdullah Mueen, Philip Brisk, and Eamonn Keogh. Matrix profile ii: Exploiting a novel algorithm and gpus to break the one hundred million barrier for time series motifs and joins. In *Data Mining (ICDM), 2016 IEEE 16th International Conference on*, pages 739–748. IEEE, 2016.

[159] Ali Ziat, Edouard Delasalles, Ludovic Denoyer, and Patrick Gallinari. Spatio-temporal neural networks for space-time series forecasting and relations discovery. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 705–714. IEEE, 2017.

[160] Michael Zibulevsky and Barak A Pearlmutter. Blind source separation by sparse decomposition in a signal dictionary. *Neural computation*, 13(4):863–882, 2001.

APPENDICES

Appendix A: Derivation of complete data likelihood

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

$$P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi}) =$$
$$P(\mathcal{X}, \mathcal{Y}, I_1 = 1, \ldots, I_N = 1, y_1, \ldots, y_N; \mathbf{w}, \mathbf{b}, \bar{N}_1, \ldots, \bar{N}_N).$$

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:

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

Apply the probabilistic graphic structure in Fig. 4.2, we have

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

Plug in the model formulation in (4.2) and (4.3) and due to conditional independence assumption of each time instance label, we arrive the final formulation of the complete

data likelihood as:

$$P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi}) =$$
$$P(\mathcal{X}) \prod_{n=1}^{N} [\mathbb{I}_{(Y_n = \cup_{t=-\Delta}^{T_n - 1 + \Delta} y_n(t))} + \mathbb{I}_{(Y_n \cup \{0\} = \cup_{t=-\Delta}^{T_n - 1 + \Delta} y_n(t))}]$$
$$\mathbb{I}_{(\sum_{t=-\Delta}^{T_n - 1 + \Delta} \mathbb{I}(y_n(t) \neq 0) \leq \bar{N}_n)} \prod_{t=-\Delta}^{T_n - 1 + \Delta} P(y_n(t) | x_n, \mathbf{w}, \mathbf{b}).$$

Appendix B: 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 conditioned on the observed data. Therefore, the auxiliary function is formulated as:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = E_{\mathbf{H}|\mathbf{D};\boldsymbol{\theta}^i,\boldsymbol{\phi}}[\log P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi}))].$$

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

$$\begin{aligned}\log P(\mathcal{D}, \mathcal{H}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \log P(\mathcal{X}) + \sum_{n=1}^{N} \log[\mathbb{I}_{(Y_n = \cup_{t=-\Delta}^{T_n-1+\Delta} y_n(t))} \\ + \mathbb{I}_{(Y_n \cup \{0\} = \cup_{t=-\Delta}^{T_n-1+\Delta} y_n(t))}] + \log(\mathbb{I}_{(\sum_{t=-\Delta}^{T_n-1+\Delta} \mathbb{I}(y_n(t)\neq 0) \leq \bar{N}_n)}) \\ + \sum_{t=-\Delta}^{T_n-1+\Delta} \log P(y_n(t)|x_n, \mathbf{w}, \mathbf{b}).\end{aligned}$$

Since the hidden data is only associated with each time instance label signal $y_1, \ldots, y_N$, the expectation of $P(I_n = 1|y_n; \bar{N}_n)$ and $P(Y_n|y_n)$ are constant. Therefore the auxiliary function is computed as:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n-1+\Delta} E_{y_n(t)|\mathbf{D};\boldsymbol{\theta}^i,\boldsymbol{\phi}}[\log P(y_n(t)|x_n, \mathbf{w}, \mathbf{b})] + const.$$

Since $\log P(y_n(t)|x_n, \mathbf{w}, \mathbf{b}) = \mathbb{I}_{(y_n(t)=c)}(\mathbf{w}_c^T \mathbf{x}_{nt} + b_c) - \log(\sum_{u=0}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{nt} + b_u})$, the final formulation of the auxiliary function is

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^i) = \sum_{n=1}^{N} \sum_{t=-\Delta}^{T_n - 1 + \Delta} [\sum_{c=0}^{C} P(y_n(t) = c|\mathbf{D}; \bar{N}_n, \mathbf{w}^i) \cdot \mathbf{w}_c^T \mathbf{x}_{nt} + b_c - \log(\sum_{u=0}^{C} e^{\mathbf{w}_u^T \mathbf{x}_{nt} + b_u})] + const.$$

Appendix C: 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 $\alpha_t(\mathbb{L}, l) = P(Y_n^t = \mathbb{L}, N_n^t = l|x_n; \boldsymbol{\theta}^i)$ and the marginal probability

$$P(Y_n^t = \mathbb{L}, N_n^t = l|x_n; \boldsymbol{\theta}^i) = \sum_{Y_n^{t-1}} \sum_{N_n^{t-1}} \sum_{y_n(t)} P(Y_n^t = \mathbb{L}, N_n^t = l, Y_n^{t-1}, N_n^{t-1}, y_n(t)|x_n; \boldsymbol{\theta}^i).$$

The forward message passing update rule can be formulated by marginalizing the previous state variables (Y_n^{t-1}, N_n^{t-1}) and the current time instance $y_n(t)$. Rely on the v-structure on the update step of the chain structure in Fig. 4.4(a) and the chain rule of the joint probability $(P(A, B, C) = P(A|B, C)P(B)P(C))$ such that

$$\begin{aligned} &P(Y_n^t = \mathbb{L}, N_n^t = l, Y_n^{t-1}, N_n^{t-1}, y_n(t)|x_n; \boldsymbol{\theta}^i) \\ &= P(Y_n^t = \mathbb{L}, N_n^t = l|Y_n^{t-1}, N_n^{t-1}, y_n(t)) \cdot \\ &\quad P(y_n(t)|x_{nt}; \mathbf{w}^i, \mathbf{b}^i) P(Y_n^{t-1}, N_n^{t-1}|x_n; \boldsymbol{\theta}^i), \end{aligned}$$

we have

$$\begin{aligned} \alpha_t(\mathbb{L}, l) = \sum_{\mathbb{A} \in 2^{Y_n}} \sum_{a=0}^{t-1} \sum_{c=0}^{C} &P(Y_n^t = \mathbb{L}, N_n^t = l|Y_n^{t-1} = \mathbb{A} \\ &, N_n^{t-1} = a, y_n(t) = c) P(y_n(t) = c|x_{nt}; \mathbf{w}^i, \mathbf{b}^i) \\ &P(Y_n^{t-1} = \mathbb{A}, N_n^{t-1} = a|x_n; \boldsymbol{\theta}^i). \end{aligned}$$

According to (4.9) and (4.10) , the conditional probability follows a deterministic rule such that $P(Y_n^t = \mathbb{L}, N_n^t = l | Y_n^{t-1} = \mathbb{A}, N_n^{t-1} = a, y_n(t) = c) = \mathbb{I}(\mathbb{L} = \mathbb{A} \cup \{c\})\mathbb{I}(l = a + \mathbb{I}(c \neq 0))$. Therefore, the update rule of the forward message passing is:

$$\begin{aligned}\alpha_t(\mathbb{L}, l) = \sum_{\mathbb{A} \in 2^{Y_n}} \sum_{a=0}^{t-1} \sum_{c=0}^{C} & \mathbb{I}(\mathbb{L} = \mathbb{A} \cup \{c\})\mathbb{I}(l = a + \mathbb{I}(c \neq 0)) \\ & \cdot P(y_n(t) = c | x_n; \mathbf{w}^i)\alpha_{t-1}(\mathbb{A}, a)\end{aligned}$$

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

$$\begin{aligned}\alpha_t(\mathbb{L}, l) = & P(y_n(t) = 0 | x_n; \mathbf{w}^i)\alpha_{t-1}(\mathbb{L}, l) \\ & + \sum_{c=1}^{C} P(y_n(t) = c | x_n; \mathbf{w}^i)\mathbb{I}(l \neq 0) \\ & [\alpha_{t-1}(\mathbb{L}, l-1) + \mathbb{I}(c \in \mathbb{L})\alpha_{t-1}(\mathbb{L}_{\setminus c}, l)].\end{aligned}$$

Appendix D: 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 $\beta_{t-1}(\mathbb{L}, l) = P(Y_n, I_n = 1|Y_n^{t-1} = \mathbb{L}, N_n^{t-1} = l, x_n; \boldsymbol{\theta}^i, \bar{N}_n)$ and the marginal probability

$$P(Y_n, I_n = 1|Y_n^{t-1}, N_n^{t-1}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) = \sum_{Y_n^t} \sum_{N_n^t} \sum_{y_n(t)} P(Y_n, I_n = 1, Y_n^t, N_n^t, y_n(t)|Y_n^{t-1}, N_n^{t-1}, x_n; \boldsymbol{\theta}^i, \bar{N}_n).$$

Rely on the v-structure on the update step of the chain structure in Fig. 4.4 (b) and the chain rule of the conditional probability ($P(A, B|C) = P(A|B, C)P(B|C)$), we have

$$\begin{aligned} &P(Y_n, I_n = 1, Y_n^t, N_n^t, y_n(t)|Y_n^{t-1}, N_n^{t-1}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\ &= P(Y_n, I_n = 1|Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1}, y_n(t), x_n; \boldsymbol{\theta}^i, \bar{N}_n) \cdot \\ &\quad P(Y_n^t, N_n^t|Y_n^{t-1}, N_n^{t-1}, y_n(t)) P(y_n(t)|x_{nt}; \mathbf{w}^i, \mathbf{b}^i). \end{aligned}$$

Given the current time joint state node (Y_n^t, N_n^t), the observed node (Y_n, I_n) is independent of the previous joint state node (Y_n^{t-1}, N_n^{t-1}) and the current time instance $y_n(t)$, so $P(Y_n, I_n = 1|Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1}, y_n(t), x_n; \boldsymbol{\theta}^i, \bar{N}_n) = P(Y_n, I_n = 1|Y_n^t, N_n^t, x_n; \boldsymbol{\theta}^i, \bar{N}_n)$. Combining the above two equations, we obtain the update rule of the backward message

passing as:

$$\begin{aligned}
&\beta_{t-1}(\mathbb{L}, l) \\
&= \sum_{\mathbb{A}\in 2^{Y_n}} \sum_{a=0}^{t} \sum_{c=0}^{C} P(Y_n, I_n = 1 | Y_n^t = \mathbb{A}, N_n^t = a, \mathbf{X}_n; \bar{N}_n, \mathbf{w}) \\
&\quad P(Y_n^t = \mathbb{A}, N_n^t = a | Y_n^{(t-1)} = \mathbb{L}, N_n^{t-1} = l, y_n(t) = c) \\
&\quad P(y_n(t) = c | x_n; \mathbf{w}^i, \mathbf{b}^i)
\end{aligned}$$

Since $P(Y_n^t = \mathbb{L}, N_n^t = l | Y_n^{t-1} = \mathbb{A}, N_n^{t-1} = a, y_n(t) = c) = \mathbb{I}(\mathbb{L} = \mathbb{A} \cup \{c\})\mathbb{I}(l = a + \mathbb{I}(c \neq 0))$ and each one of $\mathbb{A}, a$ is only limited to one value for a particular class c, therefore, the update rule of the forward message passing is:

$$\begin{aligned}
&\beta_{t-1}(\mathbb{L}, l) \\
&= \sum_{c=0}^{C} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{I}_{(c\neq 0)}) P(y_n(t) = c | x_n, \mathbf{w}^i, \mathbf{b}^i).
\end{aligned}$$

Appendix E: Derivation of joint probability on chain

To calculate the joint probability $P(y_n(t) = c, Y_n, I_n = 1|x_n; \boldsymbol{\theta}^i, \bar{N}_n)$, we apply a conditional rule that

$$P(y_n(t) = c, Y_n, I_n = 1|x_n; \boldsymbol{\theta}^i, \bar{N}_n) =$$
$$P(Y_n, I_n = 1|y_n(t) = c, x_n; \boldsymbol{\theta}^i, \bar{N}_n) p(y_n(t) = c|x_n; \mathbf{w}^i, \mathbf{b}^i).$$

Once each time instance label $y_n(t)$ is known, the observed state node (Y_n, I_n) is independent of the observed signal x_n and parameter $\boldsymbol{\theta}$, so

$$P(Y_n, I_n|y_n(t) = c, x_n; \boldsymbol{\theta}^i, \bar{N}_n) = P(Y_n, I_n|y_n(t) = c; \bar{N}_n).$$

Since $P(Y_n, I_n|y_n(t) = c; \bar{N}_n)$ can be obtained by marginalizing out the joint state nodes of both (Y_n^t, N_n^t) and (Y_n^{t-1}, N_n^{t-1}),

$$P(Y_n, I_n|y_n(t) = c; \bar{N}_n) = \sum_{Y_n^t} \sum_{N_n^t} \sum_{Y_n^{t-1}} \sum_{N_n^{t-1}}$$
$$P(Y_n, I_n, Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1}|y_n(t) = c; \bar{N}_n)$$

Apply the chain rule of the conditional probability ($P(A,B|C) = P(A|B,C)P(B|C)$),

$$\begin{aligned}
&P(Y_n, I_n, Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1} | y_n(t) = c; \bar{N}_n) \\
&= P(Y_n, I_n | Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1}, y_n(t), x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&\quad P(Y_n^{t-1}, N_n^{t-1} | x_n; \boldsymbol{\theta}^i) p(y_n(t) = c | x_n; \mathbf{w}^i)
\end{aligned}$$

Given the current time joint state node (Y_n^t, N_n^t), the observed node (Y_n, I_n) is independent of the previous joint state node (Y_n^{t-1}, N_n^{t-1}) and the current time instance $y_n(t)$, so $P(Y_n, I_n = 1 | Y_n^t, N_n^t, Y_n^{t-1}, N_n^{t-1}, y_n(t), x_n; \boldsymbol{\theta}^i, \bar{N}_n) = P(Y_n, I_n = 1 | Y_n^t, N_n^t, x_n; \boldsymbol{\theta}^i, \bar{N}_n) = \beta_t(Y_n^t, N_n^t)$. Combining the above equations, applying the deterministic rule $P(Y_n^t = \mathbb{L}, N_n^t = l | Y_n^{t-1} = \mathbb{A}, N_n^{t-1} = a, y_n(t) = c) = \mathbb{I}(\mathbb{L} = \mathbb{A} \cup \{c\}) \mathbb{I}(l = a + \mathbb{I}(c \neq 0))$ and applying the definition of the forward message $P(Y_n^{t-1}, N_n^{t-1} | x_n; \boldsymbol{\theta}^i) = \alpha_{t-1}(Y_n^{t-1}, N_n^{t-1})$, the joint probability is performed as:

$$\begin{aligned}
&P(y_n(t) = c, Y_n, I_n = 1 | x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&= \sum_{\mathbb{A} \in 2^{Y_n}} \sum_{a=0}^{t-1} \sum_{\mathbb{L} \in 2^{Y_n}} \sum_{l=0}^{t} \mathbb{I}(\mathbb{A} = \mathbb{L} \cup \{c\}) \mathbb{I}(a = l + \mathbb{I}(c \neq 0)) \\
&\quad \alpha_{t-1}(\mathbb{L}, l) \beta_t(\mathbb{A}, a) \\
&= \sum_{\mathbb{L} \in 2^{Y_n}} \sum_{l=0}^{\bar{N}_n^*} \beta_t(\mathbb{L} \cup \{c \neq 0\}, l + \mathbb{I}(c \neq 0)) \alpha_{t-1}(\mathbb{L}, l)) \\
&\quad p(y_n(t) = c | x_n; \mathbf{w}^i),
\end{aligned}$$

where $\bar{N}_n^* = \min(\bar{N}_n - \mathbb{I}(c \neq 0), t)$.

Appendix F: 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 $\alpha_t^{j-1}(\mathbb{L}, l) = P(Y_{nt}^{j-1} = \mathbb{L}, N_{nt}^{j-1} = l | x_n; \boldsymbol{\theta}^i)$ and the marginal probability

$$P(Y_{nt}^{j-1}, N_{nt}^{j-1} | x_n; \boldsymbol{\theta}^i) = \sum_{Y_{n(2t-1)}^j} \sum_{N_{n(2t-1)}^j} \sum_{Y_{n(2t)}^j} \sum_{N_{n(2t)}^j} P(Y_{nt}^{j-1}, N_{nt}^{j-1}, Y_{n(2t-1)}^j, N_{n(2t-1)}^j, Y_{n(2t)}^j, N_{n(2t)}^j | x_n; \boldsymbol{\theta}^i)$$

According to the v-structure of the update step in Fig. 4.6(a) , the joint probability can be decomposed as:

$$\begin{aligned} &P(Y_{nt}^{j-1}, N_{nt}^{j-1}, Y_{n(2t-1)}^j, N_{n(2t-1)}^j, Y_{n(2t)}^j, N_{n(2t)}^j | x_n; \boldsymbol{\theta}^i) \\ &= P(Y_{nt}^{j-1}, N_{nt}^{j-1} | Y_{n(2t-1)}^j, N_{n(2t-1)}^j, Y_{n(2t)}^j, N_{n(2t)}^j) \cdot \\ &\quad P(Y_{n(2t-1)}^j, N_{n(2t-1)}^j | x_n; \boldsymbol{\theta}^i) P(Y_{n(2t)}^j, N_{n(2t)}^j | x_n; \boldsymbol{\theta}^i) \end{aligned}$$

Due to the deterministic rule between $(Y_{nt}^{j-1}, N_{nt}^{j-1})$ and $(Y_{n(2t-1)}^j, N_{n(2t-1)}^j), (Y_{n(2t)}^j, N_{n(2t)}^j)$ as proposed in (4.15) and (4.16), $P(Y_{nt}^{j-1}, N_{nt}^{j-1} | Y_{n(2t-1)}^j, N_{n(2t-1)}^j, Y_{n(2t)}^j, N_{n(2t)}^j) = \mathbb{I}(Y_{nt}^{j-1} =^j_{n(2t-1)} \cup Y_{n(2t)}^j) \mathbb{I}(N_{nt}^{j-1} = N_{n(2t-1)}^j + N_{n(2t)}^j)$. Combining the above the equations, we obtain the

update rule of the forward message passing on tree as:

$$\begin{aligned}\alpha_t^{j-1}(\mathbb{L}, l) = & \sum_{\mathbb{A}\in 2^{Y_n}} \sum_{a=0}^{\bar{N}_n^{**}} \sum_{\mathbb{E}\in 2^{Y_n}} \sum_{e=0}^{\bar{N}_n^{**}} \mathbb{I}(\mathbb{L} = \mathbb{A}\cup\mathbb{E})\mathbb{I}(l = a+e) \\ & \cdot \alpha_{2t-1}^{j}(\mathbb{A}, a)\alpha_{2t}^{j}(\mathbb{E}, e) \\ = & \sum_{\mathbb{A}\subseteq\mathbb{L}} \sum_{a=0}^{l} \alpha_{2t-1}^{j}(\mathbb{A}, a)\alpha_{2t}^{j}(\mathbb{L}\setminus\mathbb{A}, l-a),\end{aligned}$$

where $\bar{N}_n^{**} = \min(\bar{N}_b, 2^{L-j}) + 1$.

Appendix G: Derivation of backward message passing on tree

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

$$\begin{aligned}
&P(Y_n, I_n = 1|Y^j_{n(2t-1)}, N^j_{n(2t-1)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
=&\textstyle\sum_{Y^j_{n(2t)}} \sum_{N^j_{n(2t)}} \sum_{Y^{j-1}_{nt}} \sum_{N^{j-1}_{nt}} P(Y_n, I_n = 1, Y^j_{n(2t)}, \\
&N^j_{n(2t)}, Y^{j-1}_{nt}, N^{j-1}_{nt}|Y^j_{n(2t-1)}, N^j_{n(2t-1)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n)
\end{aligned}$$

and

$$\begin{aligned}
&P(Y_n, I_n = 1|Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
=&\textstyle\sum_{Y^j_{n(2t-1)}} \sum_{N^j_{n(2t-1)}} \sum_{Y^{j-1}_{nt}} \sum_{N^{j-1}_{nt}} P(Y_n, I_n = 1, Y^j_{n(2t-1)}, \\
&N^j_{n(2t-1)}, Y^{j-1}_{nt}, N^{j-1}_{nt}|Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n).
\end{aligned}$$

According to the v-structure of the update step in Fig. 4.6(b), the joint probabilities can be decomposed as:

$$\begin{aligned}
&P(Y_n, I_n = 1, Y^j_{n(2t)}, N^j_{n(2t)}, Y^{j-1}_{nt}, N^{j-1}_{nt} \\
&\qquad |Y^j_{n(2t-1)}, N^j_{n(2t-1)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&= P(Y^{j-1}_{nt}, N^{j-1}_{nt} | Y^j_{n(2t)}, N^j_{n(2t)}, Y^j_{n(2t-1)}, N^j_{n(2t-1)}) \\
&\qquad P(Y_n, I_n = 1 | Y^{j-1}_{nt}, N^{j-1}_{nt}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&\qquad P(Y^j_{n(2t)}, N^j_{n(2t)} | x_n; \boldsymbol{\theta}^i)
\end{aligned}$$

and

$$\begin{aligned}
&P(Y_n, I_n = 1, Y^j_{n(2t-1)}, N^j_{n(2t-1)}, Y^{j-1}_{nt}, N^{j-1}_{nt} \\
&\qquad |Y^j_{n(2t)}, N^j_{n(2t)}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&= P(Y^{j-1}_{nt}, N^{j-1}_{nt} | Y^j_{n(2t)}, N^j_{n(2t)}, Y^j_{n(2t-1)}, N^j_{n(2t-1)}) \\
&\qquad P(Y_n, I_n = 1 | Y^{j-1}_{nt}, N^{j-1}_{nt}, x_n; \boldsymbol{\theta}^i, \bar{N}_n) \\
&\qquad P(Y^j_{n(2t-1)}, N^j_{n(2t-1)} | x_n; \boldsymbol{\theta}^i).
\end{aligned}$$

Due to the deterministic rule that

$$\begin{aligned}
&P(Y^{j-1}_{nt}, N^{j-1}_{nt} | Y^j_{n(2t-1)}, N^j_{n(2t-1)}, Y^j_{n(2t)}, N^j_{n(2t)}) \\
&= \mathbb{I}(Y^{j-1}_{nt} =^j_{n(2t-1)} \cup Y^j_{n(2t)}) \mathbb{I}(N^{j-1}_{nt} = N^j_{n(2t-1)} + N^j_{n(2t)}),
\end{aligned}$$

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

$$\begin{aligned}\beta^j_{2t-1}(\mathbb{A},a) = &\sum_{\mathbb{L}\in 2^{Y_n}}\sum_{l=0}^{\bar{N}^{**}_n}\sum_{\mathbb{E}\in 2^{Y_n}}\sum_{e=0}^{\bar{N}^{**}_n}\mathbb{I}(\mathbb{L}=\mathbb{A}\cup\mathbb{E})\mathbb{I}(l=a+e)\\ &\beta^{j-1}_t(\mathbb{L},l)\alpha^j_{2t}(\mathbb{E},e)\\ =&\sum_{\mathbb{E}\in 2^{Y_n}}\sum_{e=0}^{\bar{N}^{**}_n}\beta^{j-1}_t(\mathbb{A}\cup\mathbb{E},a+e)\alpha^j_{2t}(\mathbb{E},e).\end{aligned}$$

and

$$\begin{aligned}\beta^j_{2t}(\mathbb{E},e) = &\sum_{\mathbb{L}\in 2^{Y_n}}\sum_{l=0}^{\bar{N}^{**}_n}\sum_{\mathbb{E}\in 2^{Y_n}}\sum_{e=0}^{\bar{N}^{**}_n}\mathbb{I}(\mathbb{L}=\mathbb{A}\cup\mathbb{E})\mathbb{I}(l=a+e)\\ &\beta^{j-1}_t(\mathbb{L},l)\alpha^j_{2t-1}(\mathbb{A},a)\\ =&\sum_{\mathbb{A}\in 2^{Y_n}}\sum_{a=0}^{\bar{N}^{**}_n}\beta^{j-1}_t(\mathbb{A}\cup\mathbb{E},a+e)\alpha^j_{2t-1}(\mathbb{A},a).\end{aligned}$$

Appendix H: Detail of computational analysis

H.1 E-step chain inference

Time complexity is ($\mathcal{O}(\sum_{n=1}^{N}|Y_n|2^{|Y_n|}\bar{N}_nT_n)$): 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 $(\mathbb{L}, l)$, therefore, the computational complexity is $\mathcal{O}((|Y_n|+1)2^{|Y_n|}\min(t,\bar{N}_n))$. Since each time step is only depend on the previous time step, the overall computational complexity is

$$\mathcal{T}_n^c(t) = \mathcal{T}_n^c(t-1) + \mathcal{O}((|Y_n|+1)2^{|Y_n|}\min(t,\bar{N}_n)).$$

After solving this recursive formula, we have $\mathcal{T}_n^c = \mathcal{O}(|Y_n|2^{|Y_n|}\bar{N}_nT_n)$. Therefore, the overall chain inference needs a computational complexity of $\sum_{n=1}^{N}\mathcal{T}_n^c = \mathcal{O}(\sum_{n=1}^{N}|Y_n|2^{|Y_n|}\bar{N}_nT_n)$.

Space complexity is ($\mathcal{O}(2^{|Y_n|}T_n\bar{N}_n)$): For E-step chain inference with both forward and backward message passing, each forward and backward message requires $\mathcal{O}(2^{|Y_n|}\min(t+1,\bar{N}_n+1))$. Since each time step is only depend on the previous time step, the overall space complexity is calculated as:

$$\mathcal{S}_n^c(t) = \mathcal{S}_n^c(t-1) + \mathcal{O}(2^{|Y_n|}\min(t+1,\bar{N}_n+1)).$$

Solving this recursive formula, we obtain $\mathcal{S}_n^c = \sum_{t=1}^{T_n}2^{|Y_n|}\min(t+1,\bar{N}_n+1) = \mathcal{O}(2^{|Y_n|}T_n\bar{N}_n)$.

H.2 E-step tree inference

Time complexity is ($\mathcal{O}(\sum_{n=1}^{N} 4^{|Y_n|}(\log_2 \bar{N}_n)^2 T_n)$): 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 $(Y^j_{n(2t-1)}, N^j_{n(2t-1)})$ and $(Y^j_{n(2t)}, N^j_{n(2t)})$, therefore, the computational complexity is $\mathcal{O}(4^{|Y_n|}(\min(\bar{N}_n, 2^{L-j}) + 1)^2)$. However, the updates of the forward and backward messages on the tree for controlling sparsity l are operating in a convolutive nature. When $\bar{N}_n$ is large, we rely on FFT and Inverse of FFT to speedup such that the convolution complexity will become $\mathcal{O}((\min(\bar{N}_n, 2^{L-j}) + 1)\log(\min(\bar{N}_n + 1, 2^{L-j}) + 1))$. 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

$$\mathcal{T}_n^{\mathrm{tr}(j)}(t) = \mathcal{T}_n^{\mathrm{tr}(j+1)}(2t) + \mathcal{T}_n^{\mathrm{tr}(j+1)}(2t-1) + \mathcal{O}(4^{|Y_n|} X \log X),$$

where $X = \min(\bar{N}_n, 2^{L-j}) + 1$ and $1 \le t \le T_n/2^{L-j}, 1 \le j \le L$. After solving this recursive formula, we have $\mathcal{T}_n^{\mathrm{tr}} = \mathcal{O}(4^{|Y_n|}(\log_2 \bar{N}_n)^2 T_n)$. The overall computational complexity of tree approach is $\sum_{n=1}^{N} \mathcal{T}_n^{\mathrm{tr}} = \sum_{n=1}^{N} \mathcal{O}(4^{|Y_n|}(\log_2 \bar{N}_n)^2 T_n)$. **space complexity is ($\mathcal{O}(2^{|Y_n|} T_n \log_2 \bar{N}_n)$):** For tree forward and backward message passing, each node requires $\mathcal{O}(2^{|Y_n|}\min(2^j + 1, \bar{N}_n + 1))$ space. Since current instance on tree level j only depend on previous two parents' at $j + 1$, the recursive formula is:

$$\mathcal{S}_n^{\mathrm{tr}(j)}(t) = \mathcal{S}_n^{\mathrm{tr}(j+1)}(2t) + \mathcal{S}_n^{\mathrm{tr}(j+1)}(2t-1) + \mathcal{O}(2^{|Y_n|} X),$$

where $X = \min(\bar{N}_n, 2^{L-j}) + 1$ and $1 \le t \le T_n/2^{L-j}, 1 \le j \le L$. Solving it, we obtain $\mathcal{S}_n^{\mathrm{tr}} = \sum_{j=0}^{L} 2^{|Y_n|}(\min(2^j, \bar{N}_n) + 1) T_n/2^j = \mathcal{O}(2^{|Y_n|} T_n \log_2 \bar{N}_n)$.