AN ABSTRACT OF THE THESIS OF

Teresa V. Tjahja for the degree of Master of Science in Computer Science presented on June 1, 2015.

Title: Supervised Hierarchical Segmentation for Bird Bioacoustics

Abstract approved: _____

Xiaoli Z. Fern

Bioacoustics analysis can be used to conduct environmental monitoring by detecting the presence of birds species. This analysis usually involves identifying the species from their calls. In most frameworks, bird song syllables are extracted from audio recordings and individual syllables are input to a classifier to identify the species. Extraction of bird song syllables from audio recordings involves segmenting the bird song signal into individual syllables. However, syllable extraction from in-field recordings poses a challenge due to the presence of environmental noise. For such noisy recordings, supervised segmentation has been observed to perform better than unsupervised approaches. To perform segmentation, recordings are commonly converted to a time-frequency spectrogram. Supervision can then be provided at pixel level and syllable level. In pixel-level supervision, individual pixels are predicted to belong to a syllable, while in syllable-level supervision, the prediction is made for groups of pixels. In this thesis, we propose a supervised hierarchical segmentation approach that learns from both pixel and syllable levels supervision. Experimental results show that the proposed method outperforms existing supervised method that learns only at the pixel level.

Supervised Hierarchical Segmentation for Bird Bioacoustics

by

Teresa V. Tjahja

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 1, 2015
Commencement June 2015

Master of Science thesis of Teresa V. Tjahja presented on June 1, 2015.

APPROVED:

Major Professor, representing Computer Science

Director of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

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

Teresa V. Tjahja, Author

ACKNOWLEDGEMENTS

This work is partially supported by the National Science Foundation grants IIS-1055113, CCF-1254218, and DBI-1356792.

We would like to thank Matthew Betts, Sarah Frey, Adam Hadley, and Jay Sexsmith for their help in collecting the bird song recordings in HJA data, Iris Koski in labeling the data, Lawrence Neal for developing the initial segmentation algorithm that our work builds on, and Forrest Briggs for building the OSU Bioacoustics Tools.

I would also like to thank my advisor, Dr. Xiaoli Z. Fern, and Dr. Raviv Raich, Anh T. Pham and other members of OSU Bioacoustics group for their guidance and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

Chapter 1: Introduction

Recent advances in machine learning and pattern recognition enable applications in a wider range of fields. Among the fields that benefit from such advances is bioacoustics, which combines biology and acoustics in order to study animal vocalization. Studying animal vocalization is useful for species conservation [16], as well as monitoring and mitigating environmental impacts caused by human activities [21].

Bioacoustics studies the sound resulting from animals communication. One of the goals of these studies is to conduct species monitoring. Traditional efforts in species monitoring involve sending human observers to the field, where individual observers listen to the sound in the surrounding environment, then manually detect and identify the species present. This method requires tremendous human efforts, while also facing several limitations such as limited observation time and observer bias. Therefore, an automated method is required to assist human observers in analyzing the animal vocalization. Both vertebrates (e.g. amphibians, reptiles, birds, and mammals) and invertebrates (e.g. insects, spiders, and crustaceans) produce observable sound or vibration to communicate with each other [16]. Among those species that are observable in their natural habitats, birds are considered to be a good indicator of biodiversity and environmental health because they are distributed widely across landscapes [4] and their behavior quickly reflects both local and global critical environmental changes [5].

The main goal in species analysis is to identify bird species. To achieve this goal, most frameworks utilize syllables, which serve as the basic building blocks of bird songs [7]. Consequently, high segmentation accuracy is important to facilitate good detection and identification results. An intermediate goal is then to extract individual bird song syllables from a set of in-field recordings. The presence of environmental noise, such as wind and stream, contaminates the recordings and poses a significant challenge to the extraction process.

Our previous work in [20] introduced a supervised method to extract individual bird syllables from noisy audio recordings. A recording is converted to a time-frequency spectrogram and a classifier is trained by learning from human-annotated spectrograms. The

classifier is used to compute a probability map for each spectrogram, which contains the probability of each pixel belonging to a bird song segment. A global threshold is then applied to the probability map to obtain a binary mask. Each connected component in the binary mask defines a bird song segment in the original spectrogram. This method was shown to outperform the energy-based thresholding approach, which applies a global energy threshold to the original spectrogram to obtain two-dimensional segmentation. Despite the promising results with noisy field recordings, this approach has two limitations. First, determining the global threshold that works optimally for all spectrograms is difficult, since different thresholds may be required to properly extract segments from different spectrograms. Even within a single spectrogram, different segments often are best extracted with different thresholds. Second, the human-annotated examples carry relevant information about bird syllables at the segment level, such as the shapes. This information cannot be captured at the pixel level, at which the existing method learns. In fact, segmentation methods that use a single global threshold and decide whether individual pixels belong to bird syllables, such as the aforementioned energy-thresholding approach, also suffer from these limitations.

Therefore, we propose to utilize both pixel-level and segment-level supervision to overcome the two limitations. We present a segment extraction framework to learn to imitate the behavior of human annotator, so that bird song syllables can be segmented in a way that is consistent with human annotation. To achieve this, we build a segmentation hierarchy containing candidate segments for each spectrogram. Then, to utilize segment-level information about bird song syllables, each candidate segment is assigned a quality score by a predictor trained using segment-level supervision. A subset of the candidate segments that optimizes the overall quality is then extracted by applying a bottom-up selection procedure to each hierarchy.

An initial result of our framework is presented in [25], where segmentation hierarchy is generated by using a set of fixed threshold values and a segment selection procedure that heuristically chooses good-quality segments. In this thesis, we conduct further investigation to the proposed framework by exploring various ways of building the segmentation hierarchy as well as using different features to represent each segment. More importantly, we introduce a new objective for the segmentation selection problem given the generated hierarchy, and present a novel selection algorithm with provable optimality.

Evaluation of our proposed method is conducted on a set of noisy in-field recordings.

The results suggest that the segment-level quality of the solution produced by our proposed method is significantly better, while the pixel-level performance is comparable to the method in [20].

The remainder of this thesis is organized as follows. Chapter 2 discusses existing algorithms for bird song segmentation and a related framework for image segmentation. Chapter 3 describes the problem statement, while Chapter 4 provides the details of our framework. In Chapter 5, we present the results of our framework. Finally, Chapter 6 concludes our experiments and identifies potential directions for future work and improvements.

Chapter 2: Related Work

2.1 Related Work in Bird Song Segmentation

Approaches to bird song segmentation have been mainly unsupervised. Extraction of individual bird syllables is usually done by analyzing signal information such as frequency and energy or amplitude to determine the boundary of each syllable.

In [11] and [12], bird song signal is split into sinusoidal pulses, starting from the highest peak. Each pulse then represents a syllable. To refine the segmentation results, post-processing operations are done based on the size, energy, and distance between syllables [14].

In [23] and [10], an iterative time-domain algorithm is used to estimate the threshold to extract individual syllables. First, smooth energy envelope is computed for the bird song signal. A threshold for background noise level is then calculated iteratively until convergence is achieved. The threshold is then applied to the energy envelope to obtain individual bird syllables. A similar method is used in [24], where two values, i.e., the mean and standard deviations of the background noise level based on the energy envelope, instead of a single threshold, are used.

Meanwhile, [6] analyzes the frequency of bird song signal to determine the endpoints of individual bird syllables. The signal is sampled with a rectangular window and each frame is represented with a frequency bin. Fourier transform is then applied to each frame, resulting in a magnitude and phase spectrum. The principle frequency, i.e., the spectral peak of the frame, is then calculated. A feature vector consisting of the the principle frequency and its magnitude are then calculated for each frame. The starting and ending points of a syllable are then determined by finding the frames whose peak magnitudes satisfy a threshold. In [8], similar analysis of frequency bins is used to find the endpoints of individual syllables, but the input signal is first segmented using RS method in [22] before Fourier transform is applied for sampling.

In [26], an entropy-based endpoints detection is introduced. It is observed that the entropy of a block in a time-frequency spectrogram drops at the start of a signal and

increases at the end. The starting and ending points of a syllable are then determined by using a modified version of Bayesian change point detection [1].

Domain-specific human input is used in [2] and [15], where the input is provided in the form of manually-defined bird song templates. For each spectrogram and a set of templates defined manually by human experts, the goal is to find the order of occurrence and the starting and ending points of the respective template in the time-domain. This alignment is done using Dynamic Time Warping (DTW) algorithm. Evaluation is done on bird song recordings obtained in a lab setting where each animal is housed individually, where the recordings are relatively clean. These methods also do not utilize valuable information contained in the templates other than for matching the data to the templates. Consequently, their performances depend greatly on the quality of the templates.

All of the efforts mentioned above perform bird song segmentation on time-frequency spectrogram. However, the input signal is split only along the time domain (i.e., the horizontal axis). Such algorithms do not work well for field recordings where there are multiple birds singing at a given time. In such a case, multiple syllables at different frequencies occupy the same time period. Thus, segmentation along both time and frequency domains is required to accurately extract individual segments.

2.2 Related Work in Hierarchical Segmentation

Hierarchical segmentation methods have been shown to improve segmentation quality for scene and document images. One such work is presented in [13], where a segmentation framework based on multitree dictionaries is proposed. The framework builds a segmentation tree starting from a single tiling (treated as the root node), which consists of a rectangle of the whole image. Every rectangle in the tiling is then split further into multiple rectangle constructing different tilings via sequences of binary splits. The process is continued until individual pixels are obtained, or the rectangles are marked as "never be split again". Multiple trees are generated, and the tree that minimizes the sum of the cost of the individual rectangles is selected to be the final segmentation. The cost function for the rectangles is specific to the application domain. The minimization problem is then solved with a recursive algorithm that builds the optimal solution for each subtree up to the whole tree.

The framework in [13] is further improved in [27], where the input image is modeled

as Spatial Random Trees (SRTs), which are motivated by Probabilistic Context Free Grammar (PCFG) in natural language processing. Each node in the tree represents a symbol produced according to a grammar, and it has a probability derived from its construction. The cost of each pixel is defined as the negative log of its probability. To obtain the final segmentation by minimizing the cost of the tree, the MAP tree of the image is then computed.

In [3], multiscale contour detection method is used to facilitate hierarchical segmentation. Their multiscale contour detector builds from the Pb detector introduced in [19], where the function $Pb(x, y, \theta)$ measures the difference in local image brightness, color, and texture to calculate the posterior probability of boundary at location (x, y) in the image with orientation θ. The gradient, which is the basic building block in computing the Pb contour detector, is computed at multiple scale for each channel in brightness, color, and texture. The resulting cues are then linearly combined into a single signal. A globalization method based on spectral clustering is used to produce closed contours, thus splitting the original image into regions. The contour image is then used to build a segmentation tree using a greedy graph-based region merging algorithm. The individual regions become the leaf nodes, and the most similar regions are combined iteratively, until the entire image is merged in the root node. The segmentation result is obtained by selecting a scale (i.e., a level in the tree), which is determined manually.

The work in [17] uses a similar approach to hierarchical segmentation by first dividing the original image into regions. A probability map is computed using a pixel-level classifier for the original image, and region boundaries of the image are computed by using morphological watershed algorithm on the probability map. A merge tree is then constructed by merging the regions, starting from two with the highest merging saliency, until the entire image is combined at the root. Each node is then assigned a potential, which indicates the likelihood of its children merging into it. The final segmentation is then obtained by selecting a subset of the nodes from the tree via a greedy approach, i.e., by selecting first the node with the highest potential and removing conflicting nodes, until every node is either selected or removed.

The goal of hierarchical segmentation methods discussed in this section is to divide the entire input image into regions. In contrast, our framework aims to identify and detect the locations of bird syllables in the original image (i.e., spectrogram).

Chapter 3: Problem Statement

We consider the problem of segmenting individual syllables from audio recording represented with a spectrogram. The input is an audio signal $A(t)$, which is converted into a time-frequency spectrogram $S(t, f)$. The horizontal axis represents time, while the vertical axis represents frequency. The intensity of each pixel (time-frequency unit) in the spectrogram indicates the energy (amplitude). The goal is to extract individual bird syllables that from the spectrogram $S(t, f)$. The output is sets of pixels, where each set corresponds to a single utterance of bird vocalization, i.e., a bird syllable.

Often, there is ambiguity about whether a signal should be regarded as a single or multiple utterances. In our work, rather than arbitrarily define a single utterance, we take a supervised approach and assume that a set of example spectrograms have been carefully annotated (segmented) by a human expert. Since the human expert has knowledge of about the application domain, the examples provided should follow a coherent scheme about what constitutes a good syllable. Therefore, our goal is to learn from the examples to perform segmentation in the same manner.

Figure 3.1 illustrates the problem. Each spectrogram contains patterns that correspond to the input signal. Both bird song and noise segments may be present in a spectrogram. In the illustration, patterns with solid lines represent valid bird syllables, while patterns with dotted lines are noise.

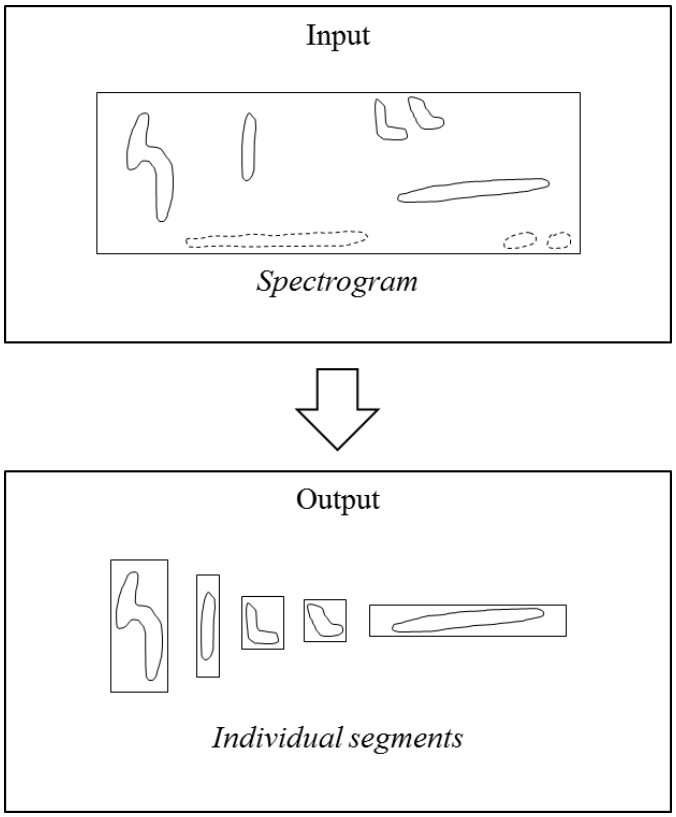

Figure 3.1: Illustration of the segmentation problem. The input are a spectrogram and a set of human-annotated spectrograms. The result is a collection of sets of pixels, where each set of pixel represents a bird syllable.[1]

[1]The spectrogram used in this figure is simply an illustration. An actual spectrogram has more noise and the patterns are often not very obvious.

Chapter 4: Proposed Method

The general idea of our proposed framework is to build a hierarchy of candidate segments and extract segments from different levels in the hierarchy. This allows for more flexibility, in the sense that no global threshold needs to be pre-determined and different segments in one spectrogram can be extracted from different levels. Once the segmentation hierarchy is constructed, good-quality segments are extracted. To do this, a quality predictor is used to assign a quality score to each candidate segment in the hierarchy. The quality predictor is trained with segment-level information available from the human-annotated spectrograms. The output of our framework is a binary segmentation mask, where each connected component represent a segment. The flowchart of our proposed framework is presented in Figure 4.1.

There are three key steps in our framework: (1) generating a segmentation hierarchy, (2) predicting the quality of each candidate segment in the hierarchy based on a learned segment quality model, and (3) selecting good-quality segments to form the final segmentation result. In the following sections, we explain these key steps in details.

4.1 Generating Segmentation Hierarchy

First, let us define what we mean by segmentation hierarchy. A segmentation hierarchy consists of a collection of nested candidate segments for an input spectrogram. In the top (first) level, segments are generally under-segmented, possibly grouping multiple segments into one. As it goes down the hierarchy, the segments are shrunk or split into smaller segments. This implies that segments at a lower level are enclosed within segments at a higher level. Formally, such segmentation hierarchy can be represented as a forest $\mathcal{F} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$, where each tree represents a set of nested segments occupying the same location in the original spectrogram. Let us define each tree as $\mathcal{T}_p = \{s_1, \ldots, s_m\}$, $p = 1, \ldots, n$. Each segment s_i, $i = 1, \ldots, m$, is defined as a set of pixels. The forest $\mathcal{F}$ then has the following properties:

- All trees are disjoint, $\mathcal{T}_p \cap \mathcal{T}_q = \emptyset$, $p, q = 1, \ldots, n$, $p \neq q$. (A candidate segment

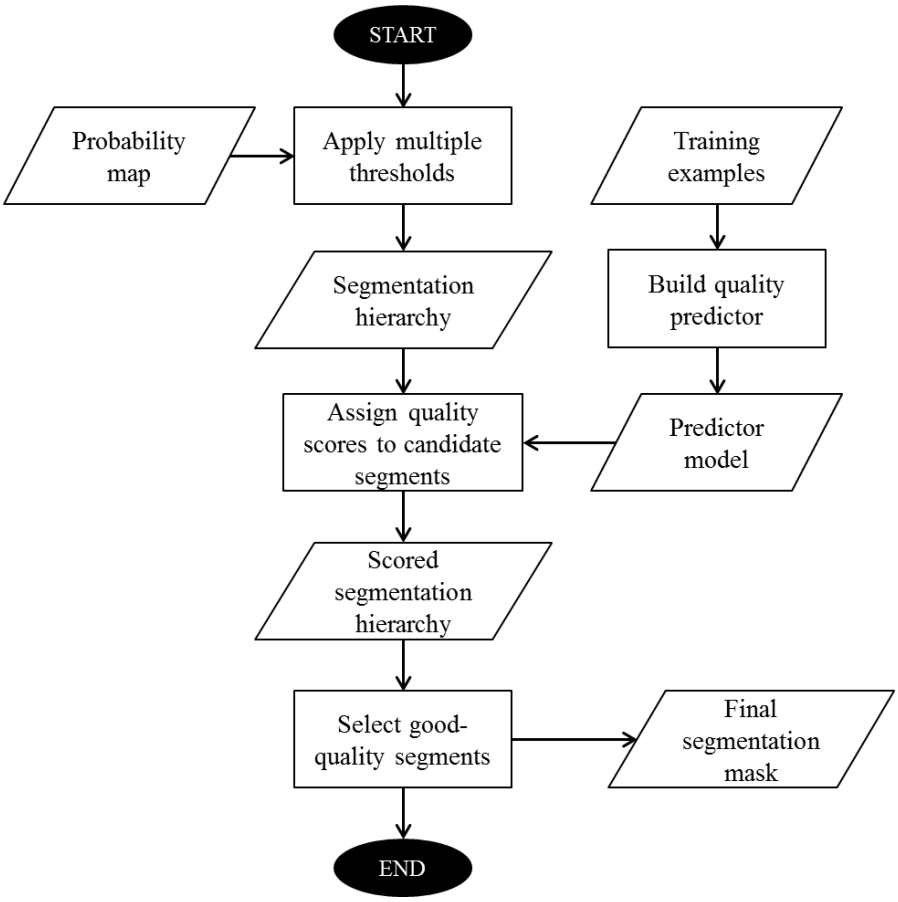

Figure 4.1: The flowchart of our proposed framework for extracting bird song segments.

belongs to only one tree.)

- If $s_i, s_j \in \mathcal{T}_p$ and s_i is the ancestor of s_j, then $s_j \subseteq s_i$, and vice versa. If s_i and s_j do not share a single path, then $s_i \cap s_j \neq \emptyset$.
- If $s_i \in \mathcal{T}_p$, $s_j \in \mathcal{T}_q$, and $p \neq q$, then $s_i \cap s_j = \emptyset$. (Segments in different trees do not overlap.)

Figure 4.2 illustrates a segmentation hierarchy produced for an input spectrogram. The corresponding segmentation trees are illustrated in Figure 4.3.

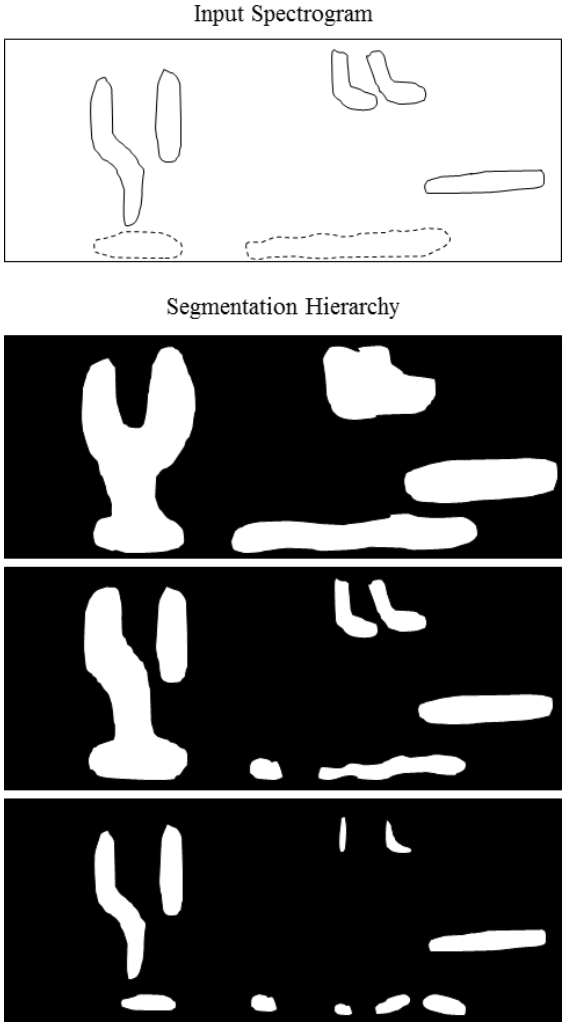

Figure 4.2: An illustrated example of a segmentation hierarchy. Each white blob in the hierarchy represents a candidate segment.

Our framework builds from the previous work presented in [20], where a probability is computed for every pixel in the input spectrogram, resulting in a probability map for each spectrogram. To generate the segmentation hierarchy in our framework, the idea is to apply a set of k thresholds, instead of a single threshold, to the probability map. Figure 4.4 shows a spectrogram with the corresponding probability map. Determining the threshold values is then an important step. Several factors should be considered dur-

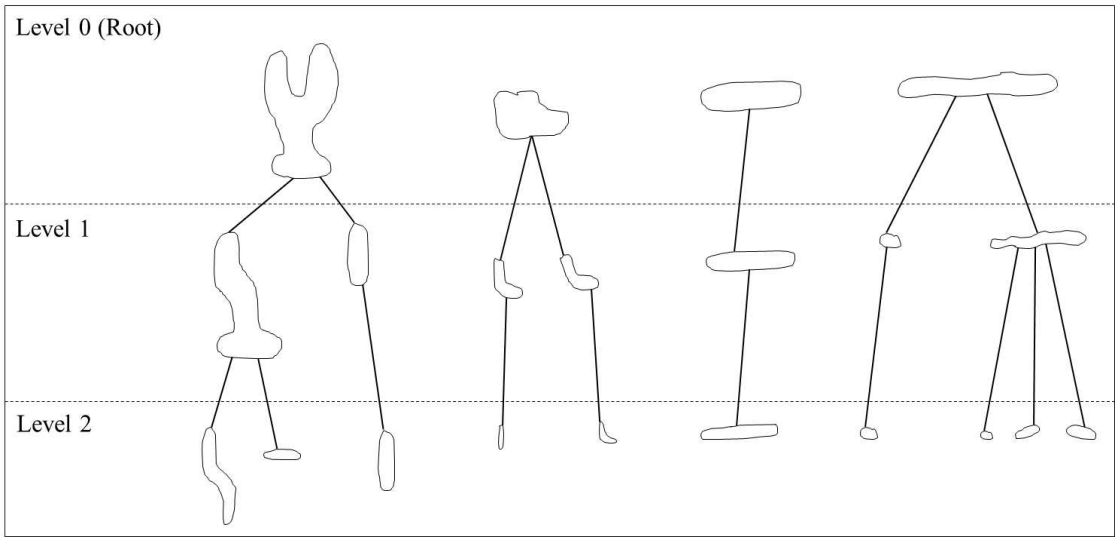

Figure 4.3: The segmentation trees representing the hierarchy in Figure 4.2.

ing this step. For instance, a large k produces deeper hierarchy, and thus provides more choices in terms of candidate segments to choose from. But, it is also computationally more expensive. With the right thresholds, smaller k may be sufficient to properly form syllables at the right levels. In our work, we consider two options for determining the thresholds: fixed and adaptive.

Fixed thresholds. A general recipe for selecting the fixed set of thresholds is to determine the fixed minimum and maximum values, then compute the thresholds with a fixed increment. The same set of values is used for every probability map. Given the minimum and maximum values π_{min} and π_{max}, the thresholds $\{\theta_1, \ldots, \theta_k\}$ are computed as

$$\theta_i = \pi_{min} + i \times \delta \tag{4.1}$$

where $i = 1, \ldots, k$ and δ is calculated as

$$\delta = \frac{\pi_{max} - \pi_{min}}{k} \tag{4.2}$$

Adaptive thresholds. In contrast to the fixed set of thresholds, which uses the same values for all probability maps, the values in the adaptive set of thresholds are computed based on the range of probabilities in each map. The individual threshold values are

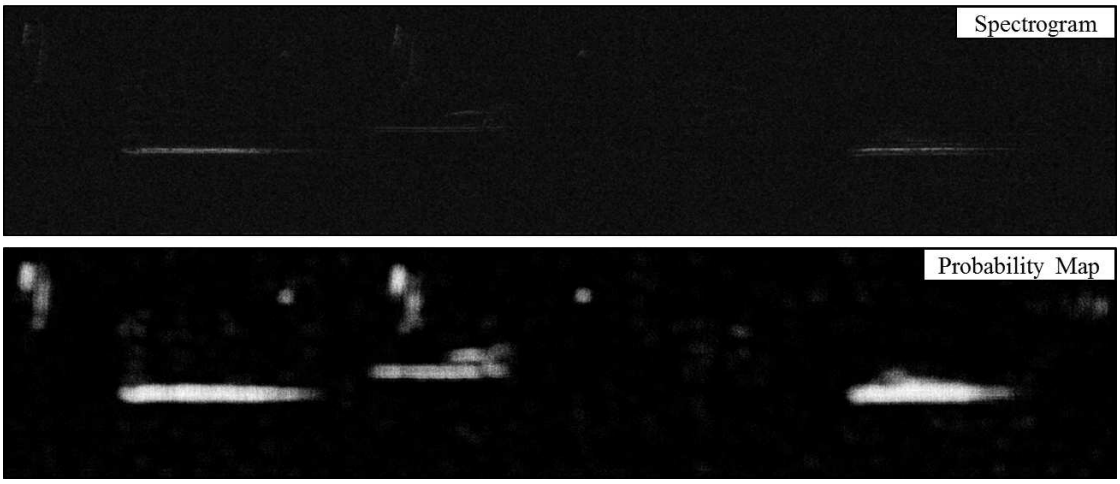

Figure 4.4: A spectrogram and its probability map. In the probability map, pixels with higher intensity have higher probability of belonging to foreground objects, i.e., bird song segments.

computed based on (4.1) and (4.2), only now the lowest and highest probabilities in the input map are used as π_{min} and π_{max}, respectively.

In our work, the probability map is computed from the input spectrogram with a Random Forest classifier. However, our method is generally applicable to other ways of producing the hierarchy, as long as the hierarchy has the properties of the forest as described above. In addition to thresholding the probability map, we also experimented directly thresholding the spectrograms (i.e., energy-based thresholding), but the results are not robust, thus are not included in this thesis.

Once the segmentation hierarchy is constructed, the next task is to select good segments among the candidates. In the following sections, we will describe a method to evaluate the quality of each candidate segment and an algorithm to select the segments.

4.2 Assigning Quality Scores to Candidate Segments

A segmentation hierarchy produced by the above method contains many nested segments. Our goal is to select among them the segments with the best qualities. To decide the quality of a candidate segment, we take a learning-based approach and train a regression

model with human-annotated spectrograms.

4.2.1 Quality Measure

To build the regression model, a segmentation hierarchy is generated for each spectrogram in the training set. We then compute a ground-truth quality score for each candidate segment using the segmentation provided by human annotator. Specifically, a quality score $y = [0, 1]$ is computed based on the segment's overlap ratio with the ground-truth segments in human-annotated spectrograms. Consider a segment s_i in the segmentation hierarchy of a particular spectrogram and a corresponding ground-truth segment g_j. Both s_i and g_j are represented as binary masks. The quality of a segment s_i is zero if the segment does not overlap with any ground-truth segment in their corresponding spectrogram. Otherwise, the overlap ratio is calculated as their Jaccard index, i.e., the intersection divided by the union of two sets, where each segment is considered as a set, as follows

$$\begin{aligned} J(s_i, g_j) &= \frac{|s_i \cap g_j|}{|s_i \cup g_j|} \\ &= \frac{\sum_{t=t_s}^{t_e} \sum_{f=f_s}^{f_e} I\{s_i(t,f) = 1 \wedge g_j(t,f) = 1\}}{\sum_{t=t_s}^{t_e} \sum_{f=f_s}^{f_e} I\{s_i(t,f) = 1 \vee g_j(t,f) = 1\}} \end{aligned} \tag{4.3}$$

where (t, f) represents the time-frequency coordinate in the original spectrogram, with (t_s, f_s) and (t_e, f_e) are the coordinates of the starting and ending points. If s_i overlaps with more than one ground truth segment, thus resulting in more than one possible values, the maximum value is used.

4.2.2 Segment Features

To accurately predict the segment quality, we need descriptive features to represent each candidate segment. We design the features based on the idea that a good segment closely resemble human-created segments. Thus, each ground-truth segment in the training set is treated as a template. Each candidate segment is then represented by its resemblance

to the templates. Formally, a candidate segment is represented with a set of template-based features, $\mathbf{x} = [x_1, \ldots, x_{|T|}]$, where T is the set of templates. In a set of training spectrograms, there can be hundreds of ground-truth segments. To speed up subsequent calculation and reduce computational complexity, we reduce the dimension of the feature vector $\mathbf{x}$ by clustering all ground-truth segments using K-means clustering. The cluster medoids are then taken as templates.

Given a set of templates T, to represent a segment s_i, the *resemblance* of s_i to each template u_j, $j = 1, \ldots, |T|$ is computed as follows. Since each bird call can have a frequency range, the frequencies of segments with similar shapes can differ slightly. To capture this property, two segments s_i and u_j are aligned based on their energy peaks. We allow segments to be aligned only if the frequencies of their peaks are within ϵ_{freq} range from each other in the time-frequency spectrogram. For instance, let (t_i, f_i) and (t_j, f_j) be the peak coordinates of s_i and u_j, respectively. Then,

$$R(s_i, u_j) = \begin{cases} J(s_i, u_j), & \text{if } |f_i - f_j| < \epsilon_{freq} \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

The alignment process is illustrated in Figure 4.5. There can be more than one peak in a segment, so three peaks from s_i and three peaks from u_j are randomly selected. This results in nine pairs of peaks, $\{\langle (t_{i1}, f_{i1}), (t_{j1}, f_{j1}) \rangle, \ldots, \langle (t_{i9}, f_{i9}), (t_{j9}, f_{j9}) \rangle\}$, and nine resemblance values, $\{R_1(s_i, u_j), \ldots, R_9(s_i, u_j)\}$. So, the j-th feature of a segment s_i is calculated as

$$x_j = \max_{k=1,\ldots,9} R_k(s_i, u_j) \tag{4.5}$$

Once they are aligned, their Jaccard index as defined in (4.3) is calculated to measure the resemblance $R(s_i, u_j)$.

4.2.3 Regression Model

The segment quality predictor is built using Support Vector Regression (SVR) [9] with a linear kernel. To select the optimal regularization parameter, 10-fold cross validation is performed at the spectrogram-level. For each training (i.e., human-annotated) spectrogram, a segmentation hierarchy is constructed. Next, the feature vector $\mathbf{x}$ and the

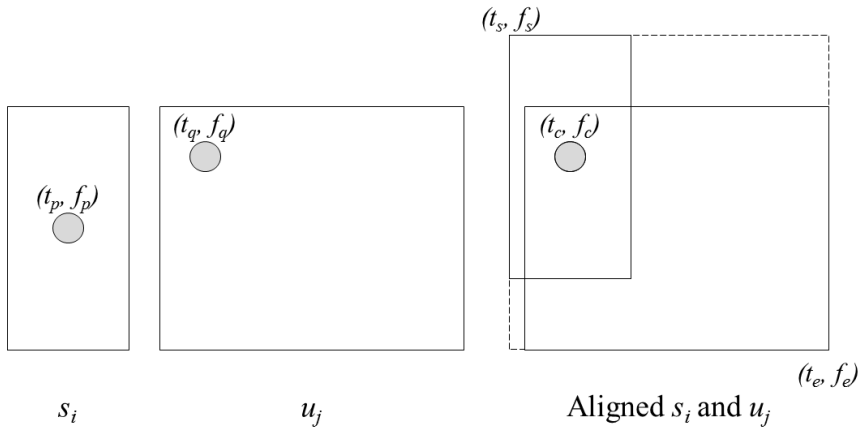

Figure 4.5: Alignment of a segment s_i and a template u_j based on energy peaks.

training label y are calculated. Now, instead of splitting the individual segments in the hierarchies from the training spectrograms, the spectrograms themselves are split into training and validation sets. For each regularization parameter, candidate segments from the training sets are then used to train the regression model, which in turn is used to predict the quality of the segments in the validation set. The selection algorithm explained in Section 4.3 is then used to form the final segmentation result of each spectrogram in the validation set. The segmentation result is evaluated with the segment-level evaluation metric described in Section 5.1.2, resulting in a score for each model. The regularization parameter that produces the maximum score is then selected.

4.3 Selecting Segments with A Bottom-Up Approach

Once the segment hierarchy is constructed and each candidate segment is assigned a quality score, we need to select a final set of segments from the hierarchy to generate the final segmentation result in the form of binary mask. Given a segmentation hierarchy, a desired solution should satisfy three criteria:

1. The solution should contain high quality segments.
2. The segments that form the final result should not overlap with each other.
3. The solution should provide a good coverage of the hierarchy. Otherwise, a simple

degenerate solution can be obtained by simply picking a single segment with the highest quality.

Based on these three factors, we pose the segment selection problem as an optimization problem.

For the purpose of our discussion, let us first define a *legal set* as follows.

Definition 1. A *legal set* of a hierarchy is a subset of segments selected from the trees in the hierarchy such that no two segments in the subset have ancestor-descendant relationship.

Definition 1 prevents extraction of overlapping segments, thus satisfying Criterion 2. This definition leads to the definition of a *maximum legal set*, which satisfies Criterion 3.

Definition 2. A *maximum legal set* is a legal set such that no more segment can be added into the set without violating the definition of a legal set.

For a given hierarchy, there can be many maximum legal sets. Our goal is to select the maximum legal set that optimizes an objective measuring the overall quality of the selected set. Specifically, we define the objective to be in the following form:

$$\max \sum_{s_i \in S^*} w_i q_i \tag{4.6}$$

with s_i is a segment in the solution set S^*, w_i is the weight of segment s_i, and q_i is its predicted quality.

Note that the segmentation hierarchy of a spectrogram is a collection of trees where different trees do not overlap. Our objective is completely decomposable, so for convenience we will consider the selection process for each tree separately, and focus on the problem of selecting the maximum legal set from a single tree of candidate segments.

Given a tree $\mathcal{T}$, we now consider how we can define the weights to make an appropriate objective. One obvious choice is to use uniform weight of 1. However, this will introduce an inherent bias in our selection to prefer larger sets. To mediate this bias, we can set the weight to be $\frac{1}{|S^*|}$, making the objective a simple average quality. This, however, makes the weight a function of the solution $|S^*|$, causing the objective to be intractable. A final factor of consideration is that we prefer larger segments to smaller ones as they are more

likely to contain complete information to help in later tasks such as species recognition. Based on these considerations, we assign a weight to each candidate segment according to its size. The weight of a root segment is 1, while a non-root segment inherits the weight of its parent, proportional to its size among its siblings. Formally, the weight is defined as

$$w_i = \begin{cases} 1, & \text{if } s_i \text{ is a root segment} \\ w_p v_i, & \text{otherwise} \end{cases} \tag{4.7}$$

where w_p is the weight of the parent of segment s_i, and v_i is the size of s_i divided by the total size of s_i and its siblings,

$$v_i = \frac{|s_i|}{|s_i| + \sum_{s_j \in B_i} |s_j|} \tag{4.8}$$

where $|s_i|$ represents the number of units (pixels) in segment s_i and B_i is the set of siblings of s_i. Each segment in the hierarchy has a quality score q_i assigned by the predictor and a weight w_i. The weighted quality of s_i is then defined as

$$\phi_i = w_i q_i \tag{4.9}$$

So, the objective function in (4.6) can be written as

$$\max \sum_{s_i \in S^*} \phi_i \tag{4.10}$$

We now have the following problem. Given a segmentation tree $\mathcal{T}$ where each segment (i.e., node) s_i is assigned a value ϕ_i, we want to select the *legal set*[1] that optimizes the objective in (4.10). This problem has a key property as stated in Lemma 1.

Lemma 1. For any segmentation tree $\mathcal{T}$, let s_r be the root of $\mathcal{T}$, and $s_1, s_2, ..., s_k$ be children of s_r. An optimal solution for $\mathcal{T}$ consists of either s_r, or $\cup_{i=1}^{k} Q_i^*$, where Q_i^* is an optimal solution of the subtree rooted at s_i.

Proof. Let $Q_{opt}(T)$ be an optimal solution for $\mathcal{T}$. $Q_{opt}(T)$ will either contain s_r or not. If it does contain s_r, it will not contain any other nodes in $\mathcal{T}$ because all other nodes are descendent of s_r. Now, consider the case where $Q_{opt}(T)$ does not contain s_r. Let

[1] Any *legal set* that optimizes the objective in (4.10) should always be a *maximum legal set*.

$\mathcal{T}_1, ..., \mathcal{T}_k$ be the subtrees rooted at $s_1, ..., s_k$, respectively. Let $Q_i = Q_{opt} \cap \mathcal{T}_i$. Because of the optimality of Q_{opt} for $\mathcal{T}$, Q_i must be an optimal solution for $\mathcal{T}_i$. In this case, we have $\mathcal{T} = \cup_{i=1}^{k} Q_i^*$. □

Lemma 1 suggests the following recursive rule. The optimal solution for a segmentation tree $\mathcal{T}$ (i.e., a subtree rooted at the root node s_r) can be obtained by comparing the optimal results at s_r and its children. If the quality of the root is greater than the sum of qualities of its children, then the root is selected as the final solution. Otherwise, the union of the optimal solution of the root's children is selected.

Given the selection problem and Lemma 1, we propose a bottom-up selection algorithm to solve the problem. Our algorithm incrementally builds up the solution for each subtree in the segmentation tree $\mathcal{T}$, starting from the smallest subtrees, until the optimal solution for the whole tree is obtained. For each of the smallest subtree, i.e., the leaf node, the optimal solution consists of the node itself. For any other subtree rooted at a non-leaf node, the optimal solution is obtained by applying the recursive rule as stated in Lemma 1. The pseudocode of our proposed algorithm is presented in Algorithm 1.

Formally, Algorithm 1 can be viewed as a dynamic programming algorithm. Let σ_i^d be the best objective value achievable for the subtree rooted at s_i and d the depth of s_i. Then, we have the following recurrence:

$$\sigma_i^d = \begin{cases} \phi_i, & \text{if } d = \text{MaxDepth}(\mathcal{T}) \\ \max\left\{\phi_i, \sum\limits_{s_j \in C_i} \sigma_j^{d+1}\right\}, & \text{otherwise} \end{cases} \tag{4.11}$$

where C_i is the set of child segments of s_i.

The prior approach in [20] corresponds to selecting a fixed level in the segment hierarchy for any spectrogram, since it uses only one global threshold to obtain the final segmentation mask. Our proposed approach provides more flexibility by enabling selection of segments from different levels for different spectrograms, as well as within a single spectrogram.

Algorithm 1 Pseudocode of the segments selection.

function SELECTSEGMENTS($\mathcal{T}$)
  $s_r \leftarrow \text{ROOT}(\mathcal{T})$
  for $d \leftarrow \text{MAXDEPTH}(\mathcal{T}), \ldots, 0$ **do**
    $S \leftarrow \{s : \text{DEPTH}(s) = d,\ s \in \mathcal{T}\}$ ▷ S is the set of all nodes at depth d
    for all $s_i \in S$ **do**
      if $d = \text{MAXDEPTH}(\mathcal{T})$ **then** ▷ Base case
        $Q_i^* \leftarrow \{s_i\}$
        $\sigma_i^d \leftarrow \phi_i$
      else
        $C_i \leftarrow \{c : \text{PARENT}(c) = s_i,\ c \in \mathcal{T}\}$ ▷ C_i is the set of the children of s_i
        if $\phi_i > \sum_{s_j \in C_i} \sigma_j^{d+1}$ **then**
          $Q_i^* \leftarrow \{s_i\}$
          $\sigma_i^d \leftarrow \phi_i$
        else
          $Q_i^* \leftarrow \bigcup_{s_j \in C_i} Q_j^*$
          $\sigma_i^d \leftarrow \sum_{s_j \in C_i} \sigma_j^{d+1}$
  return Q_r^*

Chapter 5: Evaluation

In this section, we evaluate our proposed framework and compare it to energy-based thresholding segmentation approach and the prior work in [20] with pixel-level supervised segmentation and single-level thresholding, which will be referred to as Neal's method in the subsequent discussion.

5.1 Evaluation Metrics

The energy-based thresholding, Neal's method, and our framework generates a binary mask for every spectrogram. In a binary mask M, a pixel $M(t, f)$ is assigned 1 if the pixel belongs to a bird song segment and 0 otherwise. Every connected component in the binary mask indicates the location of bird syllables in the original spectrogram. To determine how well each method performs, each system-generated binary mask is compared against the corresponding ground-truth mask annotated by human. We use two types of metrics in the evaluation.

5.1.1 Pixel-level Measure

The first metric measures the pixel-level coverage of a segmentation. This is done by calculating the True Positive Rate (TPR) and False Positive Rate (FPR) measured at pixel level, which are defined as

$$TPR = \frac{TP}{(TP + FN)} \text{ and } FPR = \frac{FP}{(FP + TN)}$$

Given a system-generated binary mask M_b and the ground-truth mask M_g, each component is calculated as follows:

- TP: the number of pixels, each denoted by a time-frequency unit (t, f), where $M_b(t, f) = 1$ and $M_g(t, f) = 1$.

- FP: the number of pixels, each denoted by a time-frequency unit (t, f), where $M_b(t, f) = 1$ and $M_g(t, f) = 0$.
- TN: the number of pixels, each denoted by a time-frequency unit (t, f), where $M_b(t, f) = 0$ and $M_g(t, f) = 0$.
- FN: the number of pixels, each denoted by a time-frequency unit (t, f), where $M_b(t, f) = 0$ and $M_g(t, f) = 1$.

Despite being widely used as a standard measure to evaluate the quality of segmentation results, such pixel-level measures are limited to capturing the quantity of ground-truth pixels covered by the segmentation result. More importantly, it does not assess how well the system-generated segments match the ground-truth at the segment level.

Example 1: Consider the results from two different segmentation methods. In the first segmentation, a single segment in the ground-truth mask is fragmented into two segments located very close to each other. In the second segmentation, the single segment is slightly shrunk, but it remains a single segment. The TPR and FPR scores of both segmentation results will be very similar, or the first segmentation may have slightly better scores. However, the result from the second segmentation is conceptually preferable since it maintains the completeness of the segment.

5.1.2 Segment-level Measure

To address the limitation of the pixel-level measure, we propose a novel segment-level quality measure. The work in [18] introduces a similar method to measure the quality of boundary segmentation. Our segment-level measure is based on the *segment mapping score*, which computes a one-to-one mapping between system-generated and ground-truth segments. The measure can be formally defined as follows. First, a complete bipartite graph $G = (V_S \cup V_G, E)$ is constructed for each spectrogram. Each element in V_S represents a system-generated segment, while each element in V_G represents a ground-truth segment. Let n_S and n_G be the numbers of segments in the system-generated and ground-truth masks, respectively. Then, G is created such that $|V_S| = |V_G| = n_t = \max(n_S, n_G)$. If $n_S < n_V$, then dummy nodes are appended to V_S so that $|V_S| = n_t$, and

vice versa for V_G. An edge $e_{ij} \in E$ is weighted by

$$w(e_{ij}) = \begin{cases} J(v_i, v_j), & \text{if } v_i \text{ and } v_j \text{ are not dummy nodes} \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

where $J(v_i, v_j)$ is the overlap ratio calculated with Jaccard index as defined in (4.3). Given a complete bipartite graph G with the above construction, we find a maximum matching $M \subseteq E$ between V_S and V_G. The *segment mapping score* μ of a segmentation mask is then calculated as the average edge weight across all matched pairs. Formally,

$$\mu = \frac{1}{n_t} \sum_{e_{ij} \in M} w(e_{ij}) \tag{5.2}$$

Consider the case between two segmentation results as described in Example 1. Using this segment-level measure, the first segmentation that splits one ground-truth segment into two will achieve a lower score compared to the second segmentation. This correctly reflects our preference.

In addition to the *segment mapping score*, we also measure the precision and recall at the segment level, where a system-generated segment is considered a match with a ground-truth segment if its mapping score is greater than a threshold θ_μ.

5.2 Experiment Setup

In this section, we provide the detailed setup of our experiments, including the datasets used and parameters specific to the proposed method as well as the baseline methods.

5.2.1 Datasets

For evaluation, we applied our proposed method to a set of 200 manually-annotated recordings, each 10-second long. The recordings were acquired from H. J. Andrews Experimental Forest in Oregon, USA using omni-directional microphones placed in natural environments. The set includes different levels of segmentation difficulty, i.e., from the relatively clean to noisy recordings, where bird song segments overlap both in time and frequency domains.

5.2.2 Parameters

We first apply the segmentation method in [20] to compute the probability map for each input spectrogram. Then, as explained in Section 4.1, a set of values need to be predetermined for generating the segmentation hierarchy using fixed thresholds. Based on empirical observations, thresholds under 0.20 causes severe under-segmentation where multiple segments are merged into one large blob, while thresholds above 0.70 produces over-segmentation where segments are often split and eroded, often changing their shapes. So, we set the number of thresholds $k = 10$ and the values start from 0.20 to 0.65, with a 0.05 increment. Each candidate segment is then represented with a 50-dimensional feature vector $\mathbf{x}$, i.e., $K = 50$ for K-means clustering mentioned in Section 4.2. Finally, to build the quality predictor using Support Vector Regression, nine regularization parameters $\mathcal{C} = 10^{-4}, 10^{-3}, \ldots, 10^{3}, 10^{4}$ are considered.

5.2.3 Features

In Section 4.2, we define template-based features to represent each segment. For evaluation, we experiment with different features by appending different features such as the segment's average energy and average probability of each segment to the 50-dimensional template-based features. Three segments of rain are also appended as templates. In addition, the 38-dimensional features proposed in [5] are also used to replace the template-based features.

5.2.4 Baselines

As mentioned in the beginning of this section, we compare our method with energy-based thresholding approach and Neal's method presented in [20]. The energy-based thresholding simply applies a single global threshold to every input spectrogram. Meanwhile, Neal's method applies a single global threshold to each probability map corresponding to an input spectrogram. Since the optimal thresholds for both baseline methods are unknown, we considered five values for each method. In particular, we use 0.070 to 0.090 with a 0.005 increment for energy-based thresholding, and 0.20 to 0.60 with a 0.10 increment for Neal's method.

5.2.5 Post-processing

To further eliminate noise, a post-processing step is performed to filter out noise segments from the final solution. Formally, for every $s_i \in Q^*$, where Q^* is the final set of segments selected by the proposed method, if $|s_i| < \theta_{filter}$, then s_i is removed from Q^*. In our experiments, we use $\theta_{filter} = 345$.

5.3 Experiment Results with Fixed and Adaptive Thresholds

In this section, we present the results of using fixed and adaptive thresholds for generating segmentation hierarchy. As discussed in Section 4.1, the fixed thresholds are predetermined and do not consider the values in each probability map. Meanwhile, the adaptive thresholds are computed based on the probabilities present on the map. The results of using the fixed set and adaptive set are presented in Table 5.1 and Table 5.2, respectively.

From Table 5.1 and 5.2, it can be seen that using fixed thresholds produces better results compared to the adaptive. The highest segment mapping score using fixed thresholds is 0.328, which is achieved by using the 50-dimensional template-based feature vector to represent each segment. Meanwhile, using adaptive thresholds only results in 0.298 segment mapping score, which is obtained by appending average energy to the 50-dimensional template-based feature vector. With adaptive thresholds, the values are all within the range of probabilities in the map. This often causes the thresholds to be very low. On the positive side, relatively faint segments (i.e., those with low energy) can be detected. But, this also creates more and larger noise segments, because the lowest threshold is not high enough to filter them out. In such cases, trees can consist of only noise segments, forcing the selection algorithm to extract the noise. This leads to higher numbers of extracted segments, consequently bringing down the mapping scores.

5.4 Experiment Results with Baselines

As can be seen in the previous section, using fixed thresholds produces better results compared to the adaptive thresholds. So, in this section, we use the results of using fixed thresholds with 50-dimensional template-based feature vector to compare with energy-based thresholding and Neal's method. The final segmentation results are shown in

Table 5.1: Evaluation of using a fixed set of thresholds for building segmentation hierarchy and using different features.

	Pixel Level		Segment Level							
	TPR	FPR	Mapping Score	$\theta_\mu = 0.3$		$\theta_\mu = 0.4$		$\theta_\mu = 0.5$		$\theta_\mu = 0.6$	
				Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Proposed Method	0.782	0.029	**0.328**	0.557	0.651	0.485	0.562	0.35	0.401	0.201	0.238
Proposed method with 50 templates and Average Energy	0.755	0.026	0.317	0.539	0.616	0.471	0.530	0.352	0.397	0.2	0.234
Proposed method with 50 templates and Average Probability	0.682	0.020	0.305	0.548	0.543	0.452	0.443	0.360	0.353	0.240	0.239
Proposed method with 50 templates and Average Energy and Probability	0.729	0.023	0.327	0.553	0.615	0.472	0.519	0.354	0.389	0.217	0.241
Proposed method with 50 templates and 3 rain templates	0.782	0.029	0.327	0.556	0.651	0.484	0.562	0.350	0.401	0.201	0.238
Proposed method with 38-D features	0.709	0.031	0.223	0.503	0.341	0.44	0.294	0.351	0.238	0.269	0.175

Table 5.2: Evaluation of using adaptive thresholds for building segmentation hierarchy and using different features.

	Pixel Level		Segment Level							
				$\theta_\mu = 0.3$		$\theta_\mu = 0.4$		$\theta_\mu = 0.5$		$\theta_\mu = 0.6$	
	TPR	FPR	Mapping Score	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Proposed method with 50 templates	0.704	0.032	0.296	0.484	0.722	0.382	0.562	0.263	0.382	0.161	0.223
Proposed method with 50 templates and Average Energy	0.702	0.033	**0.298**	0.482	0.709	0.387	0.561	0.266	0.387	0.164	0.233
Proposed method with 50 templates and Average Probability	0.663	0.029	0.292	0.469	0.653	0.385	0.523	0.274	0.365	0.169	0.235
Proposed method with 50 templates Average Energy and Probability	0.63	0.026	0.282	0.455	0.601	0.38	0.487	0.272	0.342	0.17	0.215
Proposed method with 50 templates and 3 rain templates	0.715	0.032	0.297	0.488	0.717	0.387	0.559	0.261	0.377	0.158	0.221
Proposed method with 38-D features	0.941	0.076	0.261	0.487	0.604	0.35	0.427	0.201	0.247	0.082	0.104

Table 5.3, which includes the TPR and FPR for pixel-level evaluation, and segment mapping score (μ) with precision and recall for segment-level evaluation.

As can be seen from the results, the energy-based thresholding performs rather poorly for noisy recordings in both pixel and segment levels. Meanwhile, the pixel-level quality of Neal's method is very sensitive to the global threshold. The value 0.4 as recommended by [20] produces a good balance between TPR and FPR, and achieves the top segment mapping score among the results of Neal's method. At the pixel-level, our method is comparable with Neal's method, since both methods essentially use the same pixel-level predictor. However, our method performs significantly better at the segment level, regardless of the threshold used by Neal's method. This shows that by learning from the annotations provided by human at both pixel and segment levels, our approach produces segments with better quality.

In Figure 5.1, we show a qualitative example where our proposed method can produce better segmentation results compared to Neal's method. As can be observed from the figure, the proposed method properly extract bird song segments while Neal's method splits several segments. A potential disadvantage of using the proposed method is that we have to select at least a segment from a segmentation tree, even if the tree contains only noise segments. In contrast, Neal's method may not have to extract such noise segment if its probability is less than the fixed threshold. Also, with the proposed method taking segments from multiple thresholds, more noise segments are included in the segmentation hierarchy, which may further contaminate the final result.

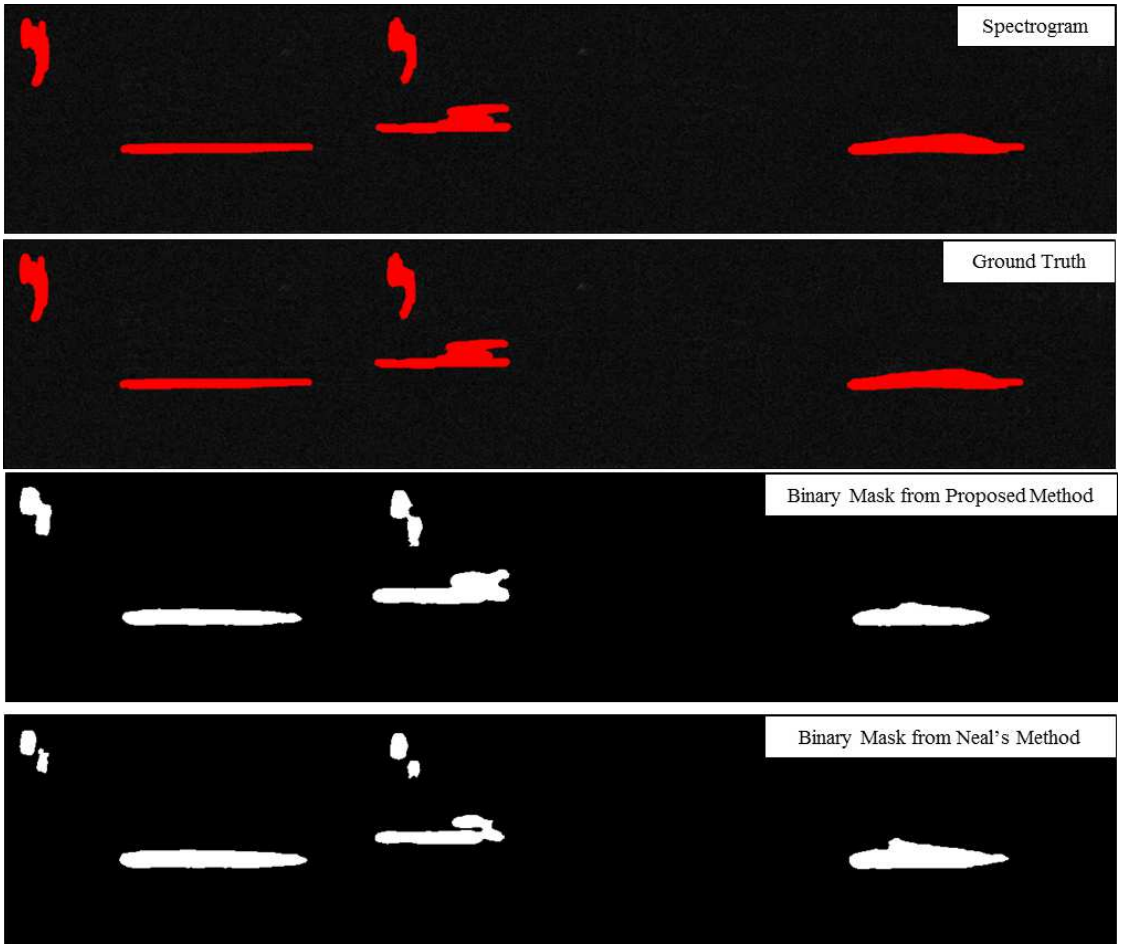

Figure 5.1: A qualitative example where the proposed framework produces better segmentation compared to Neal's method.

Table 5.3: Evaluation of energy-based thresholding, Neal's method, and the proposed method.

	Pixel Level		Segment Level							
	TPR	FPR	Mapping Score	$\theta_\mu = 0.3$		$\theta_\mu = 0.4$		$\theta_\mu = 0.5$		$\theta_\mu = 0.6$	
				Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Energy-based ($\theta = 0.070$)	0.747	0.037	0.136	0.199	0.369	0.125	0.208	0.071	0.106	0.022	0.038
Energy-based ($\theta = 0.075$)	0.599	0.015	**0.167**	0.226	0.29	0.147	0.182	0.09	0.11	0.04	0.045
Energy-based ($\theta = 0.080$)	0.53	0.01	0.16	0.217	0.252	0.148	0.167	0.093	0.105	0.042	0.047
Energy-based ($\theta = 0.085$)	0.471	0.007	0.154	0.224	0.238	0.145	0.155	0.094	0.099	0.041	0.046
Energy-based ($\theta = 0.090$)	0.416	0.005	0.139	0.224	0.213	0.146	0.140	0.086	0.084	0.038	0.041
Neal's method ($\theta = 0.20$)	0.889	0.051	0.211	0.363	0.485	0.23	0.294	0.11	0.137	0.041	0.05
Neal's method ($\theta = 0.30$)	0.829	0.035	0.226	0.39	0.496	0.252	0.311	0.138	0.165	0.047	0.055
Neal's method ($\theta = 0.40$)	0.761	0.024	**0.226**	0.393	0.439	0.273	0.299	0.154	0.162	0.069	0.072
Neal's method ($\theta = 0.50$)	0.693	0.017	0.217	0.394	0.398	0.273	0.26	0.165	0.163	0.074	0.070
Neal's method ($\theta = 0.60$)	0.619	0.012	0.203	0.387	0.336	0.271	0.225	0.167	0.142	0.077	0.07
Proposed Method	0.782	0.029	**0.328**	0.557	0.651	0.485	0.562	0.35	0.401	0.201	0.238

Chapter 6: Conclusion and Future Work

In this thesis, we developed a supervised hierarchical segmentation method to extract bird song segments from noisy recordings. Given a spectrogram and its corresponding probability map, a set of thresholds are applied to the map to generate a hierarchy of candidate segments. A regression model is then trained to predict the quality of each segment, and a bottom-up approach is used to select good-quality segments. The novel contributions of our work are:

1. We introduced a hierarchical segmentation framework to allow different bird song segments, either from the same or different spectrograms, to be extracted with different thresholds. This method is suitable for in-field recordings, where bird song signals may have different energy levels.

2. We introduced a novel supervised approach for predicting the quality of segments as a whole using not only pixel-level, but also segment-level information provided by the positive examples.

3. We introduced an optimal bottom-up approach for extracting good-quality segments from the hierarchy.

4. We proposed a novel measure to evaluate segmentation results at the segment level, which better reflects the overall segmentation quality.

Our method is most suitable for processing bird song recordings that are collected from the natural environment with various levels of noise and signal strength. The method, however, is sensitive to the quality of positive examples provided by human annotator. Inconsistent annotation may cause our method to learn imperfectly at the segment level, thus hurting subsequent extraction steps.

Future efforts can be directed toward several aspects. In terms of generating segmentation hierarchy, more robust methods can be developed to determine the set of thresholds, since the final selection algorithm strongly depends on the quality of candidate segments. For instance, the selection process can only select from the segments

formed in the hierarchy. On a related account, extracting bird song segments that (partially) overlap in both time and frequency domains is also a challenge. With the current method, segments overlapping in both domains are merged into one large blob. A more sophisticated method beyond simply applying thresholds may be employed to detect such overlapping segments. In addition, the proposed framework can also be improved by developing algorithms that can actively learn from user feedback regarding the segmentation result. Such feedback can be used to correct the quality predictor and/or the selection process itself.

Bibliography

[1] Ryan Prescott Adams and David J. C. MacKay. Bayesian online changepoint detection, 2007.

[2] S. E. Anderson, A. S. Dave, and D. Margoliash. Template-based automatic recognition of birdsong syllables from continuous recordings. *J. Acoust. Soc. Am.*, 100(2 Pt 1):1209–19, Aug 1996.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):898–916, May 2011.

[4] R. Bardeli, D. Wolff, F. Kurth, M. Koch, K.-H. Tauchert, and K.-H. Frommolt. Detecting bird sounds in a complex acoustic environment and application to bioacoustic monitoring. *Pattern Recognition Letters*, 31(12):1524 – 1534, 2010. Pattern Recognition of Non-Speech Audio.

[5] 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, 2012.

[6] Chih-Hsun Chou, Chang-Hsing Lee, and Hui-Wen Ni. Bird species recognition by comparing the hmms of the syllables. In *Innovative Computing, Information and Control, 2007. ICICIC '07. Second International Conference on*, pages 143–143, Sept 2007.

[7] Chih-Hsun Chou, Pang-Hsin Liu, and Bingjing Cai. On the studies of syllable segmentation and improving mfccs for automatic birdsong recognition. In *Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE*, pages 745–750, Dec 2008.

[8] Chih-Hsun Chou, Pang-Hsin Liu, and Bingjing Cai. On the studies of syllable segmentation and improving mfccs for automatic birdsong recognition. In *Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE*, pages 745–750, Dec 2008.

[9] Harris Drucker, Chris J.C. Burges, Linda Kaufman, Chris J. C, Burges* Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines, 1996.

[10] S. Fagerlund and U.K. Laine. New parametric representations of bird sounds for automatic classification. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 8247–8251, May 2014.

[11] A Harma. Automatic identification of bird species based on sinusoidal modeling of syllables. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 5, pages V–545–8 vol.5, April 2003.

[12] Chang hsing Lee, Yeuan kuen Lee, and Ren zhuang Huang. Automatic Recognition of Bird Songs Using Cepstral Coefficients. *Journal of Information Technology and Applications*, 1(1):17–23, May 2006.

[13] Yan Huang, I. Pollak, M.N. Do, and C.A. Bouman. Optimal tilings and best basis search in large dictionaries. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 1, pages 327–331 Vol.1, Nov 2003.

[14] P. Jancovic, M. Kokuer, and M. Russell. Bird species recognition from field recordings using hmm-based modelling of frequency tracks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 8252–8256, May 2014.

[15] J. A. Kogan and D. Margoliash. Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden markov models: A comparative study. *J. Acoust. Soc. Am.*, 103(4):2185–96, Apr 1998.

[16] Paola Laiolo. The emerging significance of bioacoustics in animal species conservation. *Biological Conservation*, 143(7):1635 – 1645, 2010. Conservation planning within emerging global climate and economic realities.

[17] Ting Liu, Cory Jones, Mojtaba Seyedhosseini, and Tolga Tasdizen. A modular hierarchical approach to 3d electron microscopy image segmentation. *Journal of Neuroscience Methods*, 226(0):88 – 102, 2014.

[18] David Royal Martin. *An Empirical Approach to Grouping and Segmentation*. PhD thesis, 2002. AAI3082313.

[19] D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(5):530–549, May 2004.

[20] L. Neal, F. Briggs, R. Raich, and X.Z. Fern. Time-frequency segmentation of bird song in noisy acoustic environments. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 2012–2015, May 2011.

[21] NIPS Int. Conf. *Proc. Neural Information Processing Scaled for Bioacoustics, from Neurons to Big Data*, USA, 2013. http://sabiod.org/nips4b.

[22] L.R. Rabiner and M.R. Sambur. An algorithm for determining the endpoints of isolated utterances. *Bell System Technical Journal, The*, 54(2):297–315, Feb 1975.

[23] P. Somervuo, A. Harma, and S. Fagerlund. Parametric representations of bird sounds for automatic species recognition. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(6):2252–2263, Nov 2006.

[24] Ryosuke O. Tachibana, Naoya Oosugi, and Kazuo Okanoya. Semi-automatic classification of birdsong elements using a linear support vector machine. *PLoS ONE*, 9(3):e92584, 03 2014.

[25] T.V. Tjahja, X.Z. Fern, R. Raich, and A.T. Pham. Supervised hierarchical segmentation for bird song recording. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, Apr 2015.

[26] Ni-Chun Wang, R.E. Hudson, Lee Ngee Tan, C.E. Taylor, A. Alwan, and Kung Yao. Bird phrase segmentation by entropy-driven change point detection. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 773–777, May 2013.

[27] Wiley Wang, I. Pollak, Tak-Shing Wong, C.A. Bouman, M.P. Harper, and J.M. Siskind. Hierarchical stochastic image grammars for classification and segmentation. *Image Processing, IEEE Transactions on*, 15(10):3033–3052, Oct 2006.