UNIVERSITY OF CALIFORNIA,
IRVINE

Towards Effective Visual Learning for Data-Centric Machine Vision

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computational Science

by

Pu Li

2023

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Professor Xiaobai Liu, for his unwavering support and invaluable guidance throughout my PhD study. His immense knowledge and experience have reshaped my understanding of academic research. He teaches me how to identify interesting research problems and approach them with a systematic methodology. He trains me to write papers in a organized, concise, and compelling manner, effectively conveying my research to the academic community. He makes me grow into an independent researcher.

Also, I would like to thank Professor Marie Roch for her insightful suggestions in my research. Her patient guidance introduces me to the field of marine mammal call extraction. She always rigorously examines my results and offers inspiring suggestions with her expertise. Her constructive feedback offers immense help in improving the quality of my work.

Additionally, I would like to express my gratitude to my co-advisor in UCI, Professor Xiaohui Xie, for his invaluable assistance in my research. He consistently reminds me to identify and address critical research problems. His expert insights and suggestions have been essential in advancing my research.

Moreover, I would like to express my deep appreciation to Professor Jérôme Gilles, Professor Wayne Hayes, and Professor Charless Fowlkes for joining my committee. Their expertise and valuable feedback enhances my understanding of my research topic and improves the quality of my work.

Besides, I would like to thank Professor Jose Castillo, Parisa Plant and Jean Macneil for their guidance and instruction throughout the doctoral program. They are always available to answer my questions and provide guidance on important milestones, ensuring that I remain on track with my work.

In addition, I would like to thank Professor Holger Klinck, Professor Erica Fleishman, Professor Eva-Marie Nosal, Professor Douglas Gillespie, Pina Gruden, Peter Conant, Yu Shiu, Kaitlin Palmer, Danielle Cholewiak, Tyler Helble for their indispensable contributions to my research.

Part of the work in this dissertation is supported by grants N000141712867 and N000142112567.

VITA

Pu Li

EDUCATION

Doctor of Philosophy in Computational Science	**2018-Present**
University of California, Irvine	*Irvine, California*
San Diego State University	*San Diego, California*
Bachelor of Science in Biology Sciences	**2011-2015**
Tsinghua University	*Beijing, China*

RESEARCH EXPERIENCE

Graduate Research Assistant	**2018–2023**
San Diego State University	*San Diego, California*

TEACHING EXPERIENCE

Teaching Assistant	**2020–2021**
San Diego State University	*San Diego, California*

REFEREED JOURNAL PUBLICATIONS

• First-authored Papers

Learning via Sampling: Building Deep Whistle Models From Raw Data **Under Review**
Nature Machine Intelligence

Policy-driven Auto-Augmentation with Distillment Rewards for Scene Text Recognition **Under Review**
IEEE Transactions on Image Processing

Using Deep Learning to Track Time×Frequency Whistle Contours of Toothed Whales without Human-annotated Training Data **Accepted**
The Journal of the Acoustical Society of America

Learning Stage-wise GANs for Whistle Extraction in Time-Frequency Spectrograms **2023**
IEEE Transactions on Multimedia

• Co-authored Papers

An Iterative Semi-Supervised Approach with Pixel-wise Contrastive Loss for Road Extraction **Under Review**
IEEE Transactions on Multimedia

Silbido Profundo: An open source package for the use of deep learning to detect odontocete whistles **2022**
The Journal of the Acoustical Society of America

REFEREED CONFERENCE PUBLICATIONS

• First-authored Papers

Learning Sample-Specific Policies for Sequential Image Augmentation **2021**
Proceedings of the 29th ACM International Conference on Multimedia

Learning Knowledge-Rich Sequential Model for Planar Homography Estimation in Aerial Video **2020**
International Conference on Pattern Recognition (ICPR)

Learning Deep Models From Synthetic Data for Extracting Dolphin Whistle Contours **2020**
International Joint Conference on Neural Networks (IJCNN)

SOFTWARE

DeepWhistle `https://github.com/Paul-LiPu/DeepWhistle`
An deep learning algorithm for odontocetes whistle extraction.

Silbido Profundo `https://github.com/MarineBioAcousticsRC/silbido`
MATLAB software that integrate DeepWhistle detector.

DeepVideoHomography `https://github.com/Paul-LiPu/DeepVideoHomography`
An deep learning algorithm for homography estimation in aerial videos.

CompositeGAN `https://github.com/Paul-LiPu/CompositeGAN_WhistleAugment`
An composite-GAN algorithm that augments whistle extraction data.

RLAugment `https://github.com/Paul-LiPu/rl_autoaug`
An reinforcement-learning algorithm for sequential image augmentation.

ABSTRACT OF THE DISSERTATION

Towards Effective Visual Learning for Data-Centric Machine Vision

By

Pu Li

Doctor of Philosophy in Computational Science

University of California, Irvine, 2023

Associate Professor Xiaobai Liu , Chair

Over the past decade, Artificial Neural Networks (ANNs) have emerged as a prevalent approach for solving a wide range of artificial intelligence tasks, including image classification, protein structure prediction, and natural language processing. These networks simulate the structure and function of biological neural networks, consisting of interconnected nodes and learnable parameters that enable them to achieve competitive performance when trained on large-scale datasets. However, manual annotation of large-scale datasets is both expensive and time-consuming, which limits the performance and real-world application of ANNs.

This dissertation proposes a series of models and algorithms for automatically establishing new training datasets or expanding existing ones without requiring additional human annotation effort. These approaches significantly improve the efficiency of visual learning while reducing the cost of annotation. In contrast to model-centric approaches that focus on improving model architectures and training strategies, the proposed methods are data-centric, with the primary goal of generating or selecting data that can benefit model training. These approaches are applied to tasks that involve recognizing specific targets within visual representations, such as images and time-frequency spectrograms of audio data.

The proposed data-centric approaches introduce three novel learning schemes to the existing literature. The first scheme involves automatic generation of training data for visual models.

To achieve this, multiple data generation methods are developed, including heuristics and learning-based generative models. The former superimposes target signals onto background scenes using manually designed rules, while the latter uses a stage-wise generative adversarial network to generate realistic data by sequentially producing backgrounds and foreground targets from random signals.

The second scheme aims to train models from an augmented dataset. Although there are various transformations that may augment training data, finding the optimal augmentation strategy from a large augmentation hyperparameter space can be challenging. To overcome this, the proposed policy-driven framework uses a reinforcement-learning model to predict the optimal sequence of transformations to be applied to each data sample.

The third scheme focuses on learning visual models from raw data and pseudo-labels, without any human annotations. To select high-quality data and reduce errors in the pseudo-labels, the proposed probability-sampling algorithm combines the assessement results of correctness, complexity, and diversity of data samples and pseudo-labels. Selected samples are then expanded with "contrastive samples" that have similar target signals but different background scenes, enabling the application of contrastive loss to provide additional guidance in model training.

The proposed methods were evaluated on multiple tasks, including whistle extraction, image classification, and scene text recognition. Extensive experiments showed that these methods achieved state-of-the-art performance on public benchmarks.

Chapter 1

Introduction

This chapter provides an overview of the dissertation. We begin by introducing the concept of data-centric visual learning in Section 1.1, which is the core focus of this work. In Section 1.1, we introduce the concept of data-centric visual learning and its significance in this work. In Section 1.2, we describe the application scenarios for the proposed data-centric visual learning methods, including three specific tasks (whistle extraction, image classification, and scene text recognition) and the corresponding datasets. We then provide an overview of the methodology applied to our target tasks and datasets in Section 1.3. We discuss the key techniques developed to enhance learning efficiency in a data-centric way, such as data generation, data augmentation, and pseudo-label selection. Finally, in Section 1.4, we outline the structure of the remaining chapters in this dissertation.

1.1 Data-centric Effective Visual Learning

Machine learning is a popular method for addressing computer vision challenges by creating mathematical models that are optimized on a set of training data [142]. These optimized models can then be used for a variety of tasks, such as image classification, object tracking, and image segmentation. Artificial neural networks (ANN) [81] have become a dominant machine learning method for achieving exceptional performance on computer vision tasks. ANN can integrate multiple layers of linear and non-linear transformations, enabling the learning of complex functions [102]. However, optimizing the vast number of parameters in such models usually requires extensive, expensive human-labeled datasets [9]. To reduce the cost of developing ANN, it is critical to investigate annotation-efficient learning approaches that can achieve satisfactory performance on target tasks with reduced annotation costs.

One of the main challenges when training ANNs with small-scale datasets is the risk of overfitting [42], which occurs when a model is too complex and captures the noise in the training data, rather than the underlying patterns that generalize to unseen data. To address this challenge, a variety of methods have been proposed to enrich the dataset and the knowledge learned by the model. For instance, active learning [31] selects the most informative samples for annotation from a large pool of unlabeled data, while data augmentation [181] generates new training samples by applying transformations to existing data, thereby increasing the dataset's diversity. Additionally, semi-supervised and unsupervised learning [200] leverage unlabeled data to improve model performance. Other approaches include task-independent pretraining [24] and transfer learning [214], which leverage knowledge learned from large-scale datasets to enhance the model's performance on a target task.

Data-centric methods seek to improve the quality of the training data by incorporating more challenging samples or augmenting the data to increase sample variance. Conversely, model-centric methods concentrate on refining the model architecture or training techniques to better utilize the available data. While model-centric methods can deliver substantial performance gains, they may be constrained by the quality and diversity of the dataset. For example, if a dataset only contains images of cars seen from the front, the model may struggle to recognize cars viewed from other angles. In contrast, data-centric methods can overcome this limitation by incorporating more diverse and relevant data. Additionally, these methods can help guide the collection and annotation of additional data, which leads to more efficient dataset improvements in terms of cost.

This dissertation mainly presents a series of methods for achieving annotation-efficient learning on visual data from a data-centric perspective. In the following sections, we will outline our application scenarios and describe our methodologies.

1.2 Application Tasks and Datasets

We apply the proposed data-centric effective visual learning methods to three distinct applications: whistle extraction, image classification, and scene text recognition. Whistle extraction focuses on the audio data domain and uses time-frequency spectrograms as the visual representation of audio data. On the other hand, image classification and scene text recognition are visual tasks that utilize image data.

1.2.1 Task: Whistle Extraction

Marine mammals, such as whales and dolphins, produce whistles that contain important information for scientific research, including species identification [54], and tracking animals' movement, behavior, density, and abundance [150, 30, 84]. Whistle analysis typically involves examining the time-frequency spectrogram of audio recordings, where whistles appear as contour-shaped signals, as shown in Figure 1.1. Analysts typically draw a polyline over each whistle as the trace of the whistle contour [15]. The shape and position of these polylines are used as the basis for whistle analysis. Recently, the increasing amount of audio data makes it challenging for scientists to manually mark the whistles in the spectrogram [171]. To address this challenge, researchers have developed whistle extraction algorithms that can automatically localize whistle signals in the time-frequency spectrogram. An example output of a whistle extraction algorithm is shown in the bottom of Figure 1.1, where the recognized whistles are colored differently.

We use the acoustic data from the Detection, Classification, Localization, and Density Estimation (DCLDE) workshop 2011 [32] for model training and evaluation. This dataset contains 393 recordings collected for five species of odontocetes: *Tursiops truncatus*, *Delphinus capensis*, *Delphinus delphis*, *Peponocephala electra*, and *Stenella longirostris*. The data is gathered using two types of hydrophones: ITC 1042 (Intl. Transducer Corp., Santa Barbara, CA) and HS 150 (Sonar Research and Development Ltd., Beverly, UK) hydrophones. The hydrophones are deployed by the R/V David Starr Jordan, mounted on the stationary platform R/P FLIP (Fisher and Spiess 1963), and launched from small boats at depths ranging from 10 to 30 meters. The acoustic signals are sampled at 192 kHz with 16 or 24 bit quantization.

To train our whistle extraction model, we utilized both a labeled dataset and an unlabeled dataset. The labeled dataset consisted of 30 recordings that were not used for evaluation in a previous study, and were annotated by analysts for whistles. These recordings spanned a total duration of 127 minutes and included 12,539 annotated whistles. In addition, we used a subset of recordings from the DCLDE workshop 2011 as our evaluation dataset, consisting of 12 recordings for four species of odontocetes, with a total duration of around 43 minutes and 6,011 annotated whistles. We did not include the recordings of short-beaked common dolphin (*Delphinus delphis*) due to annotation errors. Moreover, we also utilized an unlabeled dataset, consisting of 348 recordings for the same five species of odontocetes and spanning a total duration of around 29 hours. These recordings were not annotated by analysts and were used to further train our model in an unsupervised manner.

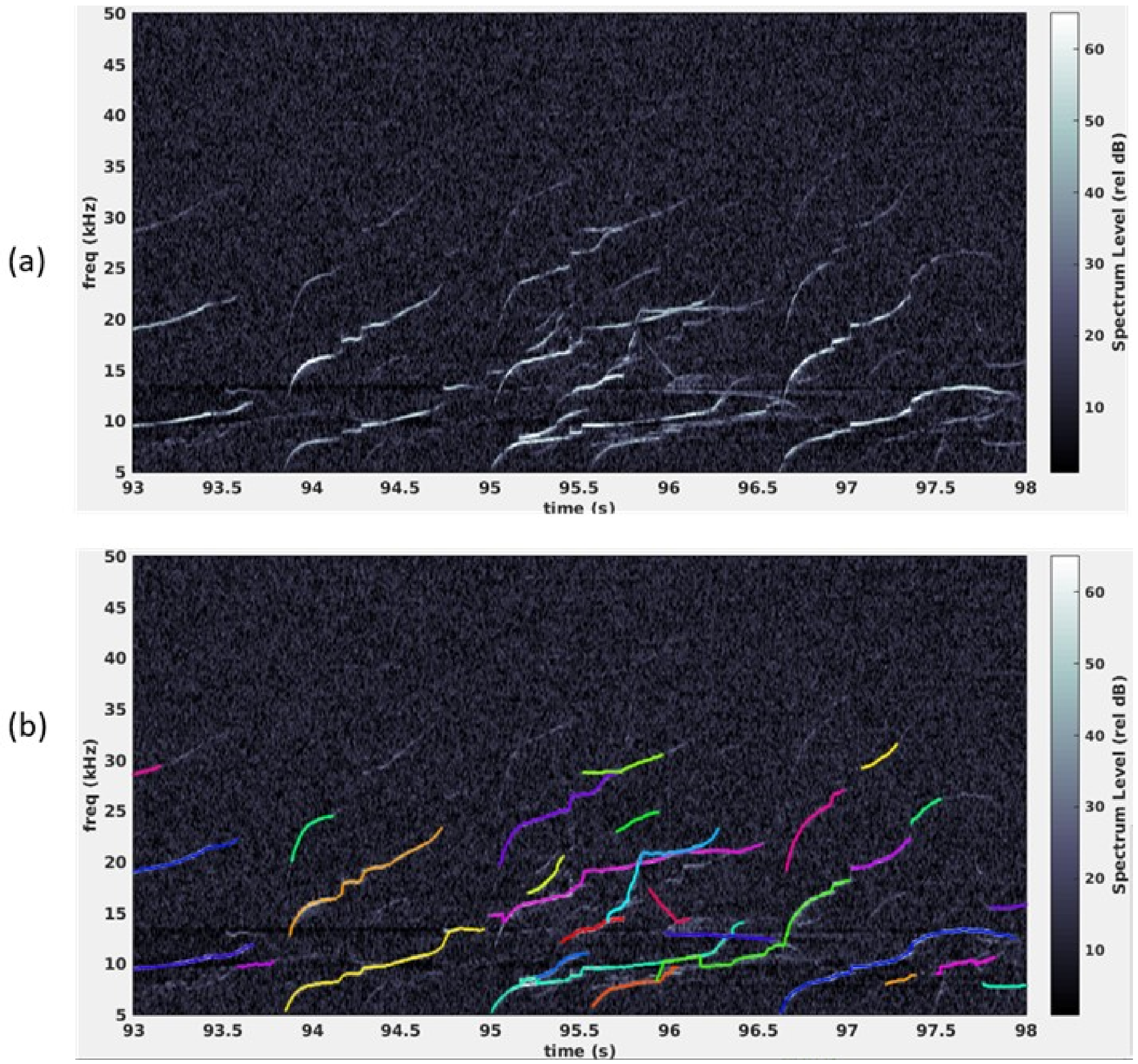


Figure 1.1: Upper: Time-frequency spectrogram with whistles produced by common dolphins (Delphinus delphis). Lower: Extracted whistle contours (colored).

1.2.2 Task: Multi-class Image Classification

Multi-class Image classification aims to assign semantic labels to images based on their visual content [124]. We focus on the single-label problem where an input image only corresponds to one semantic label. An example dataset commonly used for image classification is CIFAR10 [99], which contains images of various objects and animals. Some sample images from CIFAR10 are shown in Figure 1.2.

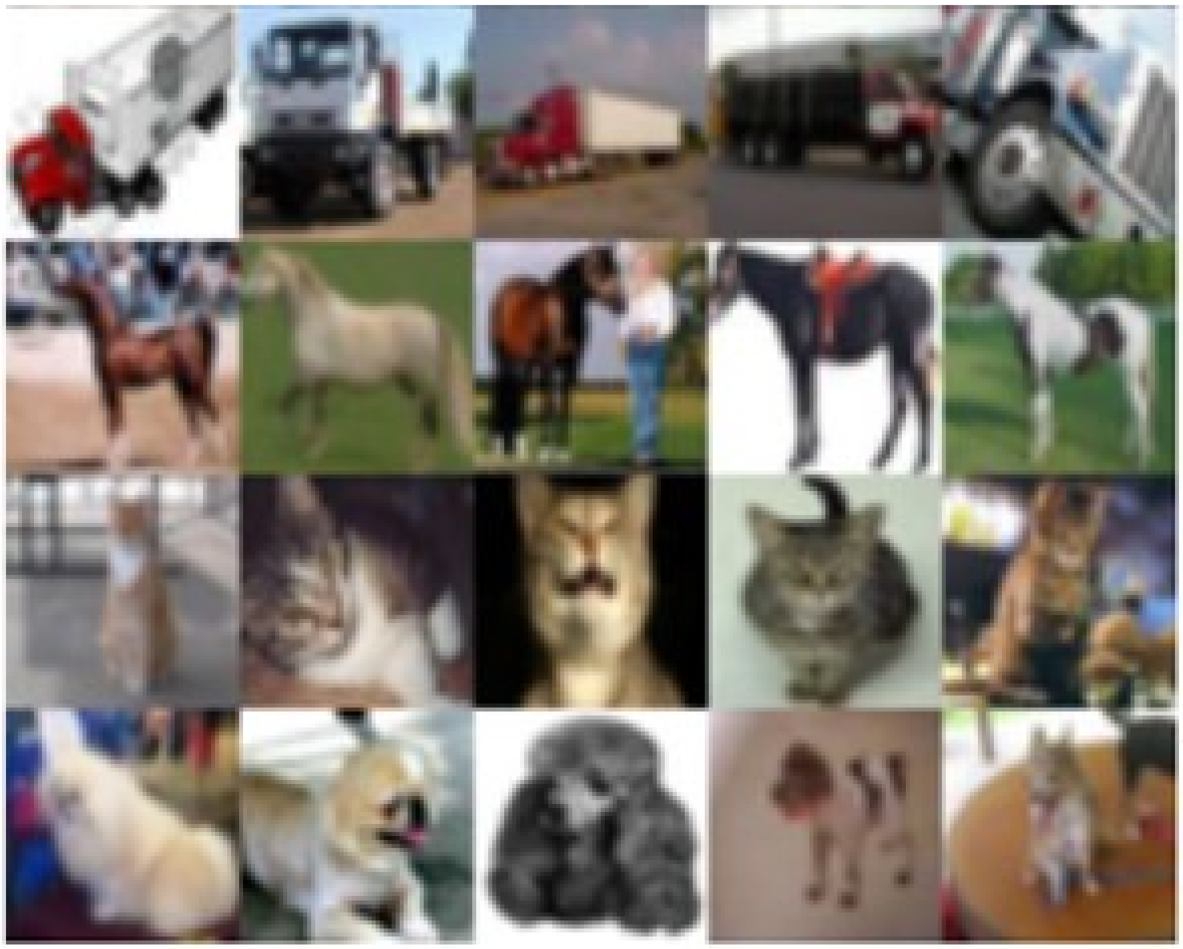


Figure 1.2: Examples of image for classification. Row 1-4 are labeled as truck, horse, cat and dog, respectively.

We apply the proposed algorithms over three public image datasets. The first one is an image dataset for garbage classification [110]. It includes 6234 images from 8 categories (5 recyclable categories: Cardboard, Glass, Metal, Paper, and Plastic; 3 non-recyclable categories: Wet Trash, Food, Bottle). We split the dataset into three-fold: a training set (2774 images), a validation set (310 images), and a testing set (3150 images). All images are resized to have a longer side of 500 pixels. We show some samples from this dataset in Figure. 1.3. The testing images are collected in different settings from the training images (lighting conditions, camera angles, etc.) from the training images, which makes this dataset a challenging task for state-of-the-art classification models.

The other two datasets are the public image datasets: CIFAR-10, CIFAR-100 [99]. We use the standard training/testing splits. CIFAR-10 consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. The dataset is divided into 50,000 training images and 10,000 testing images. It is widely used as a benchmark for image classification tasks. Similar to CIFAR-10, CIFAR-100 includes 100 classes, each containing 600 images, with a total of 60,000 color images of size 32x32, which is also divided into 50,000 training images and 10,000 testing images.

1.2.3 Task: Scene Text Recognition

Scene text recognition (STR) refers to the task of recognizing character sequences in natural scenes, where the captured text may vary significantly in appearance due to different fonts, sizes, orientations, and backgrounds [178]. The input to an STR model consists of an image containing text, and the output is a predicted sequence of characters. Notably, in contrast to traditional optical character recognition (OCR) systems that require character segmentation and recognition [18], many STR models directly recognize text in images without explicitly localizing individual characters , e.g., [178, 113, 73]. As shown in Figure 1.4, the sequential relationship between the characters is essential for the prediction, even though the character locations are not predicted.

To evaluate the effectiveness of our proposed method, we plan to apply it to both synthetic

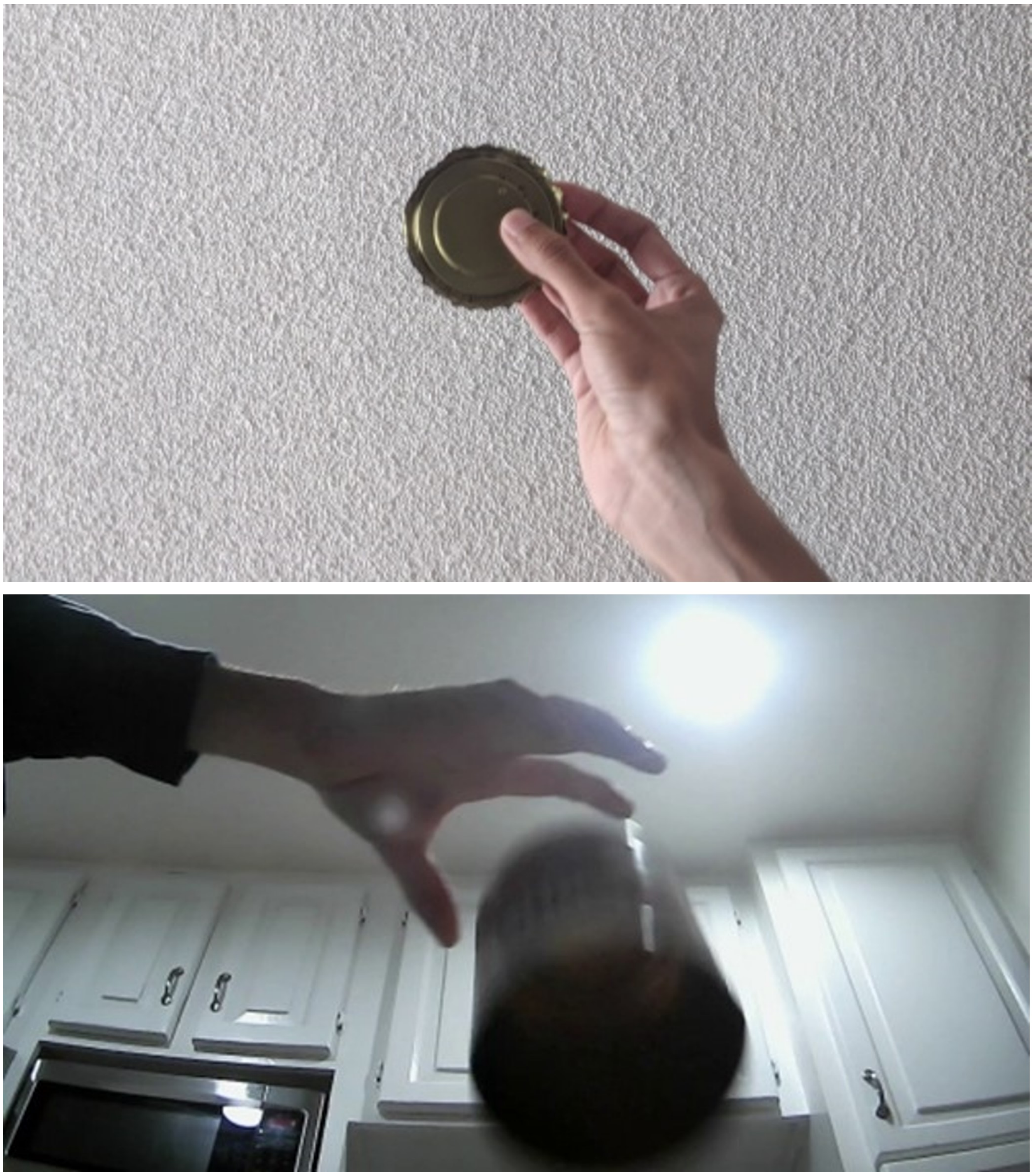


Figure 1.3: Sample images from the Garbage dataset. Two images of metal from the training (left) and testing set (right) have different appearance due to varying lighting conditions.

and real-world STR datasets, which are publicly available benchmarks for this task. These datasets present a variety of challenges such as varying text orientations, complex backgrounds, and different writing styles, making STR a challenging yet crucial task in computer vision research.

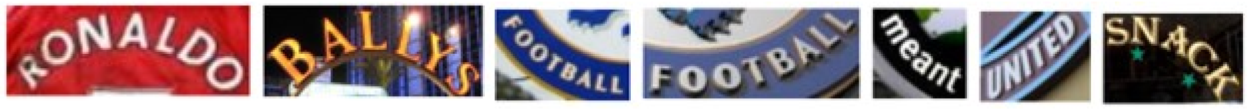


Figure 1.4: From left to right, the texts in those natural scenes are RONALDO, BALLYS, FOOTBALL, FOOTBALL, meant, UNITED, and SNACK.

Synthetic datasets For STR model pretraining, we utilized two large-scale synthetic datasets: MJSynth (**MJ**) [80] and SynthText (**ST**) [64]. The former contains 8.9 million synthetic text images, generated by blending words, borders, and shadows onto real-world images. The data generation algorithm includes perturbations such as projective distortion and noise to enhance the synthetic text. On the other hand, the latter dataset includes 5.5 million synthetic text images, introduced primarily for scene text detection where text is blended onto natural images. Text regions are cropped from the images for training and evaluating scene text recognition models.

Real-world datasets In addition to the synthetic datasets, we used seven small-scale real-world datasets for training our STR model: IIIT5K-Words (IIIT) [141], Street View Text (SVT) [205], ICDAR2003 (IC03) [127], ICDAR2013 (IC13) [90], ICDAR2015 (IC15) [89], SVT Perspective (SVTP) [159], and CUTE80 (CT) [169]. IIIT contains 2,000 training text images and 3,000 evaluation images collected through Google image search. SVT includes cropped text from Google Street View, with 257 training text images and 647 evaluation images. IC03 refers to the dataset used in the ICDAR 2003 Robust Reading competition, containing 1,156 and 1,110 images for training and evaluation, respectively. We use only the images without non-alphanumeric characters, resulting in 867 evaluation images. IC13 extends IC03 and has 848 images for training and 1,095 images for testing, but we pruned the non-alphanumeric character images, leaving 1,015 evaluation images. IC15, introduced in the 2015 Robust Reading competition, provides 4,468 training images and 2,077 evaluation images captured by moving persons wearing Google Glasses, which may lead to blurry and perspective-distorted text images. SVTP is also collected from Google Street Views but contains images with perspective distortions, with 645 evaluation images. Finally, CT contains 288 text images for evaluation, featuring curved text in natural scenes.

1.3 Methodology Overview

We adopt three data-centric approaches to reduce the burden of manual annotation. Firstly, we seek to generate new data from existing data using either a heuristic generation method or a learning-based generative model. Secondly, we propose a learning-based augmentation strategy for applying transformations to both the samples and their corresponding labels. Thirdly, we develop methods to learn from raw data and automatically generated pseudo-labels. With these approaches, we are able to create additional training data without requiring manual annotation and effectively use them for model training. The details of our proposed methods are explained in the following subsections.

1.3.1 Learning from Generated Data

We consider a scenario where our objective is to recognize target signals present on background scenes. To generate novel data, we explore the combination of various target signals and background scenes. We present both heuristic methods and learning-based methods for acquiring target signals, acquiring background signals, and compositing them to generate novel data.

Heuristic-based Method

We utilize heuristic methods to generate novel samples by combining target signals and background signals. Target signals may be obtained from either annotated or synthetic data, while background signals may be acquired from raw data. For instance, to add a car to an image of a street view, a car image can be extracted by either cutting it out from a street-view image or rendering a car image using computer graphics models. The background street-view can be acquired from images containing no cars.

To improve the integration of the target signal into the background, a function can be applied to modify the target signal. In the example of adding a car to a street-view image, the car can be moved, rotated, and resized to fit the geometry of the street scene. Additionally, to achieve a smooth transition from the background to the car, the content around the edges of the car can be blurred. Heuristic weighting parameters can also be applied to the background and target signal before addition. These parameters may also increase the variance of the target signal.

We employ these heuristic-based methods as a baseline for generating additional data.

Learning-based Generative Model

The aforementioned heuristic method may face limitations in generating novel and realistic target signals. For instance, if a car is generated by cropping it from an existing image, the view and shape of the car may not be altered. On the other hand, if a car is synthesized using computer graphics models, it may not appear realistic when added to a real street view image.

To mitigate the limitations of heuristic methods and increase the diversity of generated data, we employ generative adversarial networks to generate realistic novel signals. While

GANs have shown success in synthesizing visually appealing samples and augmenting existing data [49, 133, 12, 76], there are still challenges for using GANs to synthesize high-quality augmented data, especially for pixel-wise regression tasks such as semantic segmentation.

One limitation of GANs is the mode collapse problem [235], where the generated samples have lower variance than the real samples. Another limitation is the potential for generating samples with artifacts or failure regions [154], which can impede the training of pixel-wise regression tasks. To address this issue, a sample selection method may be required to choose high-quality samples from the GAN-generated samples [146]. Furthermore, the training of GANs can be unstable, resulting in generated samples having a different distribution compared to real samples [1].

Therefore, GAN-based data augmentation usually requires improving the quality of generated samples. A common solution is to use real samples or computer graphics models in the generator network. In [3], the GAN learned to generate samples conditioned on real samples and random numbers. Similarly, Mu et al. [145] transferred synthetic images built by computer graphics models to realistic images, and the augmented samples improved models in estimation of gazes, hand poses, and animal poses. Another way to improve training of the GAN is to use supervision from target tasks. Waheed et al. [203] added an auxiliary classification head on the discriminator of GAN and used the classification loss to guide discriminator and generator learning. Recently, stage-wise GANs were proposed to augment data for pixel-wise regression tasks. Pandey et al. [156] employed a two-stage GAN augmentation of cell nuclei segmentation data. Their framework generates a cell nuclei segmentation mask in the first stage and images of nuclei in the second stage.

Our proposed method is closely related to [156], and we further separate the learning of object appearance and the segmentation mask. This separation can be extended to other semantic segmentation scenarios. For example, when generating a scene containing road and cars, our framework may first generate the appearance of the road and car independent of the segmentation mask, then generate an image of the scene according to the segmentation mask and the appearance of the objects (road and cars). In this way, our framework explores the distribution of object appearance and provides variance in the appearance of objects in the generated image of the scene. Another improvement is that we employ the knowledge from segmentation networks to regularize bin-wise correspondence between generated samples and labels.

Background Signal and Target Signal Synthesis We assume that the background signals or target signals follow an implicit distribution which can be modeled generative adversarial network (GAN) [57]. Specifically, we train a generator network to map multivariate Gaussian variables to desired samples. At the same time, we train a discriminator network to provide feedback on how well the generative samples match the desired distribution.

The training of the generator network and discriminator network is a zero-sum game where these two networks are alternatively updated [57]. After the generator network generates samples, the discriminator network takes both generated samples and real samples as input. The training objective of discriminator network is to distinguish between the generated and real samples by maximizing the distance between their outputs. The output of discriminator network can be a value [5, 56], which summarize the whole image, or vector [79, 244], where each element represents a patch on the image. The distance metric can be euclidean distance [244], KL divergence [56], Wasserstein distance [5], etc. On the other hand, the generator's objective is to generate samples that are increasingly indistinguishable from the real data. To achieve this goal, the generator network adjusts its parameters to minimize the distance between the generated and real samples, as quantified by the discriminator network.

Composite Target Signal and Background Signal We employ unpaired domain transfer algorithms to composite target signal and background signal. Domain transfer is the task to transform data from one domain to another, e.g., converting an image of horse to an image of zebra. Unpaired domain transfer algorithms use no paired data between the source domain and target domain for model training [244]. Instead, they use two sets of generator and discriminator networks. One set transfers samples from the source domain to the target domain, while the other set performs the opposite. The discriminator networks are trained to distinguish generated samples from real samples, as we discussed above. In contrast, the generator networks are trained to minimize the cycle loss in addition to enlarging the discriminator loss. We can transfer one sample to another domain by using one generator network, and then transfer it back via the other generator network. The cycle loss is defined as the distance between the original sample and the sample generated after this cycle.

Our proposed approach involves using a stage-wise generative adversarial network to decompose the novel data generation into three phases: (i) generating the background signals, (ii) generating the target signal, and (iii) adding the target signal to the background. By breaking down the process in this way, we are able to explore the composition among different signals and backgrounds. Moreover, each stage involves a simpler task when compared to directly generating the required data. This simplifies and stabilizes the training of the generative model. We applied our proposed stage-wise GAN framework to augment whistle extraction datasets of varying sizes and observed consistent and significant improvements. While GAN frameworks have been previously used for spectrogram generation and data augmentation in audio recognition tasks [204], to the best of our knowledge, this is the first work to apply GAN-based augmentation specifically to audio spectrogram segmentation data.

1.3.2 Learning from Augmented Data

When augmenting data by transforming existing data, it is essential to determine the optimal data augmentation strategy, i.e., a set of transformations that best improves the model performance. However, due to the vast number of possible transformations, traditional search methods such as grid search may become computationally infeasible. There are many transformation types, e.g., translation and rotation. Each type may have different magnitudes or parameters to be determined, such as the number of pixels and direction for image translation, and can be applied with different probabilities. For example, if there are 10 transformation types, each with 10 magnitudes and 10 levels of probabilities, a single training sample can be transformed in 10^3 ways. With n samples in one dataset, the number of ways to transform the dataset becomes 10^{3n}. Moreover, when multiple transformations are combined, the number of potential augmentation strategies increases exponentially, making the search for an optimal strategy more challenging.

In recent years, researchers have proposed searching for the optimal data augmentation policy on a dataset level. One such approach, introduced by Cubuk et al. [34], defines an augmentation strategy as a set of sub-policies, each consisting of two consecutive image transformations that can be applied to each sample in the dataset. The performance of a candidate policy is evaluated by how effectively it improves a proxy task, such as classification performance on a hold-out dataset. The optimal policy is obtained by searching in the hyper-parameter space, including transformation types, operation magnitudes, and the possibility of transformations. This search is made feasible through the use of reinforcement learning models [34], Bayesian optimization [115], population-based training [74], reduced hyper-parameter space [35], etc. These search-based augmentation methods has been successfully applied in multiple image relevant tasks, including image classification [34], and object detection [246].

The search-based augmentation methods mentioned above aim to find an optimal augmentation policy that can be uniformly applied to the entire dataset. Although such a dataset-scale policy may work well for the majority of the training images, it may not be optimal for some images, whose optimal augmentation policies may differ from the others. Therefore, it is crucial to optimize the optimal transformation or transformation combinations for each sample independently.

There have been attempts to explore augmentation strategies based on image content as well [48, 162]. Fawzi et al. [48] search for perspective transformations that maximize the classification loss, but the algorithm operates on perspective transformation matrix and may not be easily extended to other types of transformations. Ratner et al. [162] use generative adversarial training to generate a sequence of transformations for each sample, but the length of the sequence is fixed, making it impossible to apply different numbers of transformations to each sample. Additionally, the objective of generative adversarial training encourages the transformed images to follow the distribution of the original data, which may exclude severe transformations and difficult samples for the classifier.

We formulate all possible transformation as a sequence of atomic transformations over each training sample. Denoting the transformation sequence as f_1, f_2, ..., f_t, the transformed version of the original sample x_0 at step t is given by $x_t = f_t \circ f_{t-1} \ldots \circ x_0$, where $\circ$ denotes the transformation operator. Each transformation f_i is defined by its transformation type and magnitude. The transformed sample at a step of the sequence serves as the input to the next transformation, enabling us to cast the search for the optimal transformation sequence as a sequential decision process.

We tackle the sequential decision problem with reinforcement learning methods, e.g., deep Q-learning [201]. An imaginary agent will take actions based on the current observed state to acquire the maximum cumulative rewards in the future. In our case, each action is drawn from an action set including all transformation operations. The agent's action is modeled by a map called policy. In Q-learning, for example, the agent will select action which have the largest Q-value, where Q-value is the expected cumulative reward after taking a certain action at the current state. A policy model can be trained on observation data consisting of states, actions, and rewards. We will briefly describe the states, actions, and rewards for the agent in our data augmentation strategy learning.

Action The action space includes all possible transformations that the agent can use to augment each individual sample-label pair. A transformation may have two attributes: transformation type and magnitude. We also introduce a special action, namely the stop action, which, once chosen, terminates the sequence of transformations. Note that both transformations and early stops are determined by the policy network based on the training sample. The action space may be discrete or continuous.

State The state of the environment is a function of time, representing the information available to the agent at a certain time. In our method, the state is defined as a combination of the training sample and the previous actions taken. The previous actions are encoded as one-hot vectors without including the stop action, and the initial action is a zero vector. The state, used as input to the policy model, provides essential information for the agent's decision. For example, an agent may not translate the image further to the left if the object for classification is near the left boundary, and it may avoid translating the image to the right if it had already moved the image to the left in the previous step.

Reward An action leads to a change of state and an immediate reward. The agent's goal is to maximize the cumulative reward, which is the weighted sum of the rewards obtained in all following states. Depending on the type of augmented sample we aim to generate, there may be different ways to define the reward of an action. For instance, we may want to create samples that are more difficult for a classifier to recognize correctly. In this case, the reward of a transformation can be defined as the difference between the classifier's predictions before and after applying the transformation.

Objective function Various deep reinforcement learning frameworks can be applied to train the policy network depending on whether our action space is discrete or continuous. In the case of a discrete action space, we can use double Q-learning [201] with the temporal difference objective. If our action space is continuous, we may use the Proximal Policy Optimization algorithm (PPO) [174] to train both the actor network, which predicts actions, and the critic network, which predicts the reward.

A reinforcement learning-based framework enables us to find an optimal sample-specific augmentation strategy on a vast search space, which allows for more effective augmentations for each sample and improving the efficiency of the augmentation process in terms of the number of augmented samples. Additionally, since the augmentation policy is based on sample content, the early stop action help avoid the generation of corrupted augmented samples. We applied the proposed framework to two different tasks: image classification and scene text recognition. To encourage the generation of more challenging augmented samples in image classification, we gave a higher reward when the transformation caused a greater drop in classification confidence. For scene text recognition, we trained two STR models on different datasets and assigned a higher reward if one transformation led to a smaller difference between their predictions. This approach aimed to reduce dataset bias in the augmented samples. Using the proposed policy-learning framework and these reward functions, we were able to improve the performance and augmentation efficiency over existing augmentation approaches for both tasks.

1.3.3 Learning from Raw Data and Pseudo-label

Our goal is to create or enhance datasets without requiring human effort in annotation by utilizing raw data and pseudo-labels. We begin by an existing method for generating pseudo-labels, which can be achieved using either a non-learning-based technique, such as the Canny detector [16] for edge detection in images, or a learning-based approach, turning the learning process into a semi-supervised one [105]. Though generating significant amounts of data with pseudo-labels is cost-effective, many of these pseudo-labels may contain errors.

To address the issue of label noise, three categories of approaches have been proposed. The first category aims to identify and select a subset of samples that have high-quality pseudo-labels. Choi et al. [27] proposed a density-based approach that clusters data and assumed that samples in high-density clusters were likely to have correct pseudo-labels. Nishi et al. [149] computed the loss values on augmented samples and considered samples with lower loss values to have higher quality pseudo-labels. Zhou et al. [243] proposed using loss dynamics and the consistency of model outputs among different augmentations for pseudo-label evaluation. Another approach [225] selected pseudo-labels based on prediction discrepancy among a sequence of model checkpoints. Although these methods have introduced different metrics for selecting pseudo-labels, there is currently no unified framework to combine multiple metrics.

The second category of approaches aims to leverage pseudo-labels through various designs of loss functions or network optimization processes. Castell et al. [19] introduced the SuperLoss, which automatically downweight the contribution of samples with a large loss. In the work of [153], two types of pseudo-labels were utilized, and a noise-aware loss function was designed to reduce the weight of the loss on pixels where the pseudo-labels disagreed with each other. Tan et al. [192] applied a contrastive loss and a structural similarity loss to reduce the impact of noisy labels. Xia et al. [219] categorized model parameters into two types: one that plays a key role in learning clean labels and the other that tends to fit noisy labels. Different update rules were applied to these two types of parameters to reduce the effect of noisy labels. While these proposed loss functions are not combined with the pseudo-label and sample selection process, we introduce a triplet loss function to fully explore the similarities between selected samples.

The third category of approaches involves refining pseudo-labels using specific cues. One example is the method proposed by Zhang et al. [239], in which they periodically generate object re-identification pseudo-labels using data clustering and use the consensus of all generations to refine the pseudo-labels. Li et al. [112] combine noisy labels and model output using a weighted sum as the refined pseudo-label. Tanaka et al. [193] alternatively update network parameters and labels in each training epoch, with the labels being updated using the network output.

We propose a probability sampling framework that unifies multiple metrics, including correctness, complexity, and diversity, to select samples and pseudo-labels of higher quality. Additionally, we design a seed-and-expansion selection process that can be readily combined with contrastive loss.

Pseudo-label Data Sampling

The proposed probability sampling method begins with a set of numerical measures used to evaluate the quality of the pseudo-labels. Specifically, we assess the sample and pseudo-label from three perspectives: correctness of the pseudo-label, complexity of the target signal, and diversity of the target signal. We will elaborate on the idea of quality assessment in the following paragraphs.

Measure I: Correctness We introduce a metric to assess the correctness of the pseudo-labels. As human annotations are unavailable, we cannot directly measure the distance between the pseudo-labels and the true labels. Instead, we train a model using all available pseudo-labels and measure the correctness by the similarity between the output of this network and the pseudo-labels. When the network prediction is closer to the pseudo-label, the pseudo-label is less likely to be an outlier in the training data and is thus more likely to be correct. Different similarity metrics can be employed depending on the application. For instance, Kullback–Leibler (KL) divergence [87] may be used for image classification data, while intersection over union (IoU) [168] may be used for object detection data.

Measure II: Complexity We define complexity as the level of how complex the target signals are. The appropriate measurement of complexity may vary depending on the application. For instance, in an object detection task, the number or size of the objects may serve as an indicator. To ensure that challenging samples are included in the training dataset, we prioritize the selection of samples with more complex target signals in the pseudo-labels. However, such pseudo-labels may also contain more false positives. To address this issue, we consider combining the complexity and correctness measures. For example, instead of solely relying on one of the scores, we can use the product of the correctness score and complexity score.

Measure III: Diversity We introduce the diversity measure to evaluate the variability of the selected samples. Diversity is a commonly used standard in active learning [175]. Given a set of selected samples, we measure the diversity of a candidate sample based on its dissimilarity to other selected samples. The dissimilarity between two samples is computed using a specific distance metric on the features of the pseudo-labels and the samples. For example, in the case of semantic segmentation, the Euclidean distance between the Histogram of Oriented Gradients (HOG) features [38] of pseudo-labels can be used as the distance metric. We can measure the dissimilarity between one sample and a set of samples using the minimum dissimilarity between this sample and any one sample of this set. By ensuring the selection of samples with higher diversity, the model is exposed to various target signals during training, which helps improve its generalization ability.

The three measures presented above each characterize a different aspect of a candidate sample and should be effectively combined to optimize the selection of pseudo-labels. However, finding a subset of data that has the best diversity may require exploring all possible combinations of data samples, which is an NP-hard problem. Combining the diversity measure with the other two measures may further increase the difficulty in finding the best selection. Therefore, we propose a nuanced probability sampling method for combining these measures.

The proposed probability sampling method is based on Metropolis-Hastings (MH) algorithm [26] which simulates a Markov process moving towards a target state. In our case, we aim to select a fixed number of samples from the dataset and the target state is the optimal selection. The Markov process start with a state where we randomly select the expected number of samples. We then generate action proposals that replace one selected sample with one unused sample. Each proposal can be accepted or rejected. If the proposal is accepted, the new state is kept and the next action proposal is based on the updated state. Otherwise, the new state is discarded, and the next action proposal works on the current state. For each proposal, we generate a uniform random number between 0 and 1, and the acceptance of the proposal is determined by whether the random number is smaller than the proposal's acceptance ratio. The acceptance ratio indicates how probable the new proposed state is with respect to the current state. We design the acceptance ratio as a function of the three measures described above, allowing us to prioritize samples with higher correctness and complexity and more diversity in the selection. By generating proposals and changing the selected samples for a certain number of times, we can effectively optimize the selection and arrive at a set of samples that have high correctness, complexity, and diversity.

Loss Function

We select samples that are suitable for contrastive loss [21]. Specifically, we consider a pair of samples as a positive pair if they have similar target signals, while negative pairs have different target signals. We assume that such samples are available in the raw dataset. For example, if we capture a video using a camera moving around a car, different frames will contain the same target car with different background scenes. We can use the contrastive learning loss to regularize the training, which encourages the positive pairs of samples to have similar model output and negative pairs to have dissimilar model output.

1.4 Dissertation Outline

The remainder of the dissertation is organized as follows:

- **Chapter 2.** We implement the heuristic methods to generate extra training data for whistle extraction. We develop a deep-learning-based method for extracting whistles of toothed whales (Odontoceti) in hydrophone recordings. The audio signals are transformed into time-frequency spectrograms, and a deep neural network is trained to predict time-frequency bins that are associated with whistles. To decrease the need for extensive manual annotation, we synthesize training samples by combining background environments and whistle-like signals. Furthermore, we apply a recall-guided data sampling method during model training, which selects more difficult samples in the later phases of training. Without whistle annotation, our proposed method is able to surpass the non-learning-based alternative on a subset of the DCLDE 2011 dataset. This chapter was published in 2020 International Joint Conference on Neural Networks (IJCNN) [109].

- **Chapter 3.** We implement the framework of stage-wise generative adversarial networks (GANs) to increase the variation of generated samples. The proposed GAN compiles new whistle data suitable for deep model training via three stages: generation of background noise in the spectrogram, generation of whistle contours, and generation of whistle signals. By separating the generation of different components in the samples, our framework composes visually promising whistle data and labels even when few expert-annotated data are available. For annotated datasets with different sizes, the proposed data augmentation framework leads to a consistent improvement in performance of the whistle extraction model. This chapter was published on IEEE Transactions on Multimedia [111].

- **Chapter 4.** We implement the proposed policy-learning framework for sample-specific image augmentation in image classification datasets. The policy network generates a sequence of image transformations over a training image, with the aim of pursuing a higher reward by selecting samples that are difficult to correctly recognize by the classifier. We iteratively train the classifier and the policy network, which further augment the dataset without redundancy. We have applied this approach to both public image classification benchmarks and a newly collected image dataset for material recognition. Comparing to alternative state-of-the-art methods, our policy-driven approach achieves comparable or improved classification performance while using significantly fewer augmented images. This chapter was published on Proceedings of the 29th ACM International Conference on Multimedia [110].

- **Chapter 5.** We extend the policy-driven sequential image augmentation approach to a sequence-prediction task: scene text recognition. The main technical contribution of this work is the auto-augmentation scheme, which partitions the training images into two subsets and aims to minimize the divergence between their distributions. We introduce a distillment reward function to quantify the cross-subset divergence and guide the learning of the augmentation agent. The proposed learned augmentation policy outperforms alternative methods in terms of scene text recognition performance. This chapter was submitted to IEEE Transactions on Image Processing.

- **Chapter 6.** We propose an enhanced loss function to address errors in pseudo-labels. We observe that pseudo-labels usually miss labeling some whistles, which results in a trained model having a low recall in predicting whistle signals. Therefore, we assign a higher weight to the time-frequency bins that contain whistles according to the pseudo-labels. This encourages the model to achieve a higher recall when predicting the presence of whistles. We evaluate the proposed technique on two sets of pseudo-labels generated using different algorithms, and both experiments show that the proposed loss function significantly improves the performance of whistle extraction. In the best-case scenario, pseudo-labels can result in performance comparable to that achieved through thousands of human-annotated whistles. This chapter was accepted by the Journal of the Acoustical Society of America.

- **Chapter 7.** We implement the probability-sampling-based pseudo-label selection framework on whistle extraction data. The first step of our approach is to apply an off-the-shelf unsupervised detector to identify whistles in spectrograms. Next, we employ a sampling framework that selects the most informative samples based on several metrics. These metrics include heuristic approximations of the correctness of the pseudo-labels, the complexity of the whistle signals, and the diversity of the selected samples. To combine these metrics, we introduce a probabilistic model and use a Metropolis-Hastings algorithm to select samples with optimal correctness, complexity, and diversity. Moreover, we expand the selection by incorporating samples with similar whistle contour shapes but varying background noise, which are suitable for contrastive learning. Our proposed method consistently improves the performance of whistle extraction models across varying datasets. This chapter was submitted to Nature Machine Intelligence.

- **Chapter 8.** This chapter concludes the results of this dissertation and includes discussion of future directions.

Chapter 2

Heuristic Whistle-extraction Data Generation

2.1 Introduction

Background Many animals produce calls that may contain information on species' identity (e.g., [54]), movement (e.g., [150]), individual identity (e.g., [30]), behavior (e.g., [83]), or provide clues into density and abundance (e.g., [84]). Passive acoustic recorders collect audio data that can be analyzed to yield such information. The cost of data collection has fallen due to innovation driven by demand for consumer electronics, but processing the extract detailed animal acoustic signals from continuous acoustic recordings.

Our objective was to develop automated methods to extract whistles from an environment with many confounding sound sources. To illustrate, we applied our method to an input spectrogram (Fig. 2.1 upper panel) to produce whistle contours (middle panel). Contours represent the time-frequency traces of odontocete whistles. Ecological, behavioral, and communication questions, such as individual identity and population density, can be answered with acoustic recordings of animals. In many cases, doing so relies on detailed information about individual animal call attributes including duration, frequency range, and frequency modulation. Therefore, in many contexts, it is insufficient to simply know that animals are present and calling.

Contour detection can be challenging due to spatial and temporal differences in ocean soundscapes, including varying sea state; precipitation; animal behavior; whistles that overlap in time and frequency; non-target biotic sounds, such as dolphin echolocation clicks or the impulsive signals of snapping shrimp; and abiotic sounds, such as those made by vessels.

In next section, we review existing methods for extracting whistle contours and related problems. Among these methods, learning-based approaches can model the statistical properties of whistle contours directly and often perform better than other methods. However, it is expensive and time-consuming to manually annotate whistle contours in time-frequency spectrograms, and doing so requires expertise in bioacoustics. Moreover, training a modern machine-learning model, e.g., a deep neural network, often requires many more annotated samples than classical models. Therefore, we investigated learning-based methods that can leverage small numbers of human annotations, or no annotations at all, to predict whistles.

We aimed to generate a pointwise, two-dimensional confidence map to represent whistle contours in time$\times$frequency spectrograms of varying durations. Each element of the map represents the likelihood that a corresponding time$\times$frequency point in a spectrogram belongs to a whistle contour. Similar to edges or boundaries in natural images, short segments of whistle contours often have primitive (simple) shapes, such as a crossing between two whistles or a frequency-modulated sweep. For example, in Fig. 2.1 middle panel, the light blue and purple whistles overlap between 95.5 and 96 s.

Classical computer vision methods, e.g., primal sketch [63] or textons [245], employ a set of primitive shapes to parse natural images or videos. Rather than directly learning a dictionary

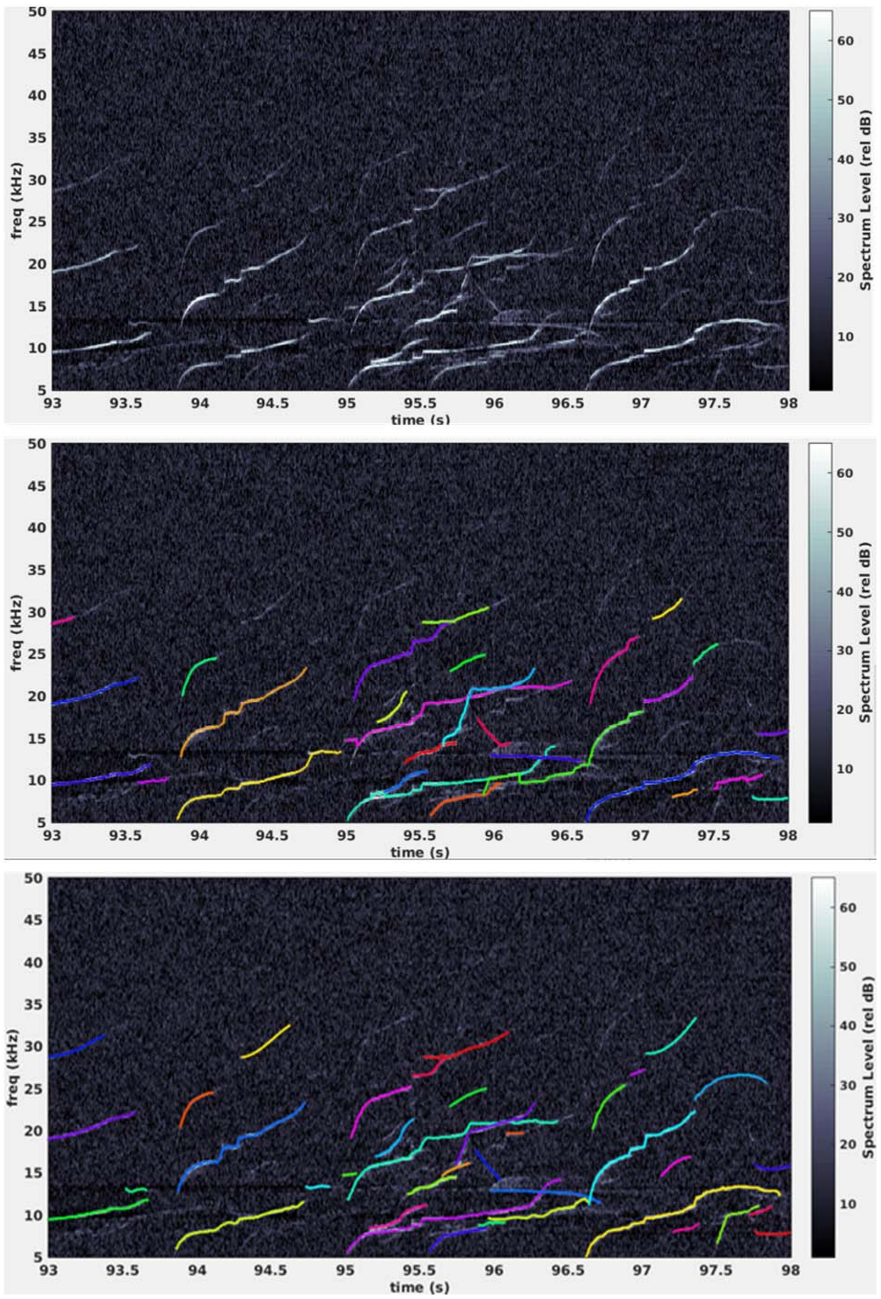


Figure 2.1: Whistle Contour Extraction. (a) Time-frequency spectrogram with whistles produced by common dolphins. Horizontal: time; Vertical: Frequency. (b) Whistle contours (colored) detected by the proposed method. (c) System performance in F-Score versus proportions of training samples.

of such primitive shapes, we employed a deep convolutional network to implicitly learn the whistle information from small patches of natural or synthetic spectrograms and a label set that indicated where whistles occurred within each patch. During testing, we partitioned the spectrogram into small patches and assembled the confidence map predictions into a larger map, which we then processed with an existing, graphbased whistle extractor to measure the performance of our system and provide an equivalent comparison with other methods (Fig. 2.2) [171].

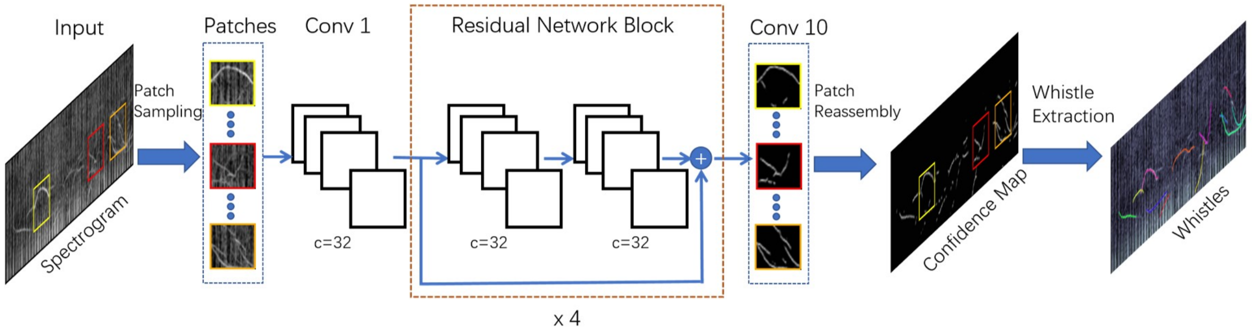


Figure 2.2: Flowchart of the proposed method. The graph search method [171] is applied to the confidence map for whistle extraction. There are four Residual blocks (brown) before output layer Conv10.

The performance of the proposed method largely depends on the quality and quantity of patch-shape pairs used to train the deep network. We found in our experiments that the performance of our method decreased substantially as the number of training samples decreased. To overcome this, we developed a comprehensive learning-from-synthesis approach to train the network from synthetic data. We sampled small time-frequency patches from spectrograms of ocean sound without whistles. Into these, we blended primitive, whistlelike shapes, permitting the learning of whistle characteristics in diverse sound environments. We examined multiple methods that collect primitive shapes from different sources, including whistle annotations in spectrograms and edge annotations in natural images. We also used the edge detections of classical computer vision methods (e.g., Canny's edge detector [16]) to synthesize data, which allowed us to extract whistles without using any human annotations. We used these automatically generated spectrogram patches and their shapes to train the deep network. Empirical studies of these data-generation methods suggested that our method is capable of achieving high accuracy while substantially reducing the amount of human effort required.

2.2 Relations to Previous Works

2.2.1 Whistle Contour Extraction

Previous researchers used different methods to extract delphinid whistle contours. Trajectory-search methods seek peaks in the spectral energy of short consecutive segments and stitch neighboring peaks together on the basis of a trajectory estimate ([54, 171, 137]). Other work has examined local context, performing ridge regressions [95] or building on ridge regression maps by using energy minimization algorithms to find contours followed by a final classification to reduce excessive numbers of false positives [176].

Probabilistic frameworks are an alternative approach to extracting whistle contours. Examples of this approach include hypothesis tests of spectrogram region distributions [36], Bayesian inference [66], Kalman filters [132], particle filters [171, 215], and multi-target tracking [59].

Neural networks have been used to extract tonal information in human speech and music tasks [68, 11], and usually perform better than probabilistic methods when the number of annotations is sufficient. The same methods may be applicable to problems in marine acoustics. However, the latter application has not been tested, in part because there are orders of magnitude fewer annotated data for marine mammal tonal calls than human speech and music. These techniques are promising and have the potential to significantly improve the efficacy of passive acoustic monitoring and animal conservation efforts.

2.2.2 Deep Models

Convolutional neural networks are effective for pixel-wise classification in computer vision methods, such as image contour detection [222, 177, 223, 121] and semantic segmentation

[123]. These neural networks implicitly leverage the context around each pixel to predict labels. Our aim is analogous: to predict the location of whistle time-frequency nodes on the basis of the local intensity and shape of the whistle contours in spectrograms.

2.2.3 Dictionary Learning

Dictionary learning [189] is a strategy for discovering a set of vectors, or atoms, that can be used to reconstruct elements of a data set. The well-known k-means algorithm [190] is a simple example of such learning, with the k mean vectors serving as the atoms of the dictionary. Dictionary learning has been applied to tasks including image super-resolution [224], face recognition [136], image denoising [131], and visual tracking of objects [136]. Traditional methods usually build a dictionary of bases (atoms) and then use the dictionary to reconstruct the input image. The dictionary often is explicitly specified through hand-crafting or discriminative training. In this work, we built an overcomplete dictionary of primitive whistle shapes to represent time-frequency spectrograms. During testing, we sampled multiple small-size timefrequency patches from the input spectrogram and employed a deep neural network to retrieve the primitive whistle shape of each patch.

2.2.4 Learning from Synthesis

The power of deep convolutional neural networks to detect and classify objects is limited when there are insufficient quantities of annotated data. Data synthesis, or the use of algorithms to develop artificial training examples, can help to fill this gap. Data synthesis differs from data augmentation, which is a transformation of an existing instance. Data synthesis has increased performance in the fields of text localization [64] and instance detection [45]. In this work, we examined multiple approaches for generating time-frequency whistle patches to augment or entirely replace training data. We also studied a network learning algorithm that allowed us to adaptively synthesize the most important samples over training steps.

2.3 Our Approach

2.3.1 Overview

Our method for extracting whistle contours in timefrequency spectrograms has three major steps: (i) sample patches from the input spectrogram, (ii) feed each patch into a neural network to obtain its corresponding confidence map, (iii) aggregate patch confidence maps to obtain an overall contour confidence map the same size as the original input spectrogram (Fig. 2.2). We used graph-search [171] to extract discrete whistle contours from the aggregated confidence map. The network learned a representation of the contour patches. Effective training of such a network usually requires hundreds of thousands of annotated examples. The quality of annotations largely relies on expert knowledge of the sounds that the target taxonomic group produces, and obtaining annotated data is time and labor intensive. Limiting or eliminating the need for humans to annotate spectrograms while maintaining performance levels would be a major advance in learning-based whistle detection.

2.3.2 Background: Graph Search Method

We first introduce the graph search algorithm [171] which organizes time-frequency peaks into a graph. At each time step, peaks either start new graphs or are used to extend terminating nodes of existing graphs when the new peak is a reasonable extension to a graph as predicted by a low-order polynomial fit. Graphs can bridge gaps in detected peaks up to a user-established interval, reducing the impact of missed peak detections. When a peak can be added to a pair of graphs, the graphs are merged. Whe a graph has not been extended within a specified time, it is considered closed, and the internal nodes with multiple inputs and outputs represent positions where whistles have crossed. Forward and backward predicting polynomial fits are used to determine which branches of the graph should be placed together to form a whistle track.

2.3.3 Data and Signal Processing

We used a subset of the annotated delphinid data from the Detection, Classification, Localization, and Density Estimation (DCLDE) 2011 workshop [32]. We standardized the recordings (192 kHz sample rate, 16 or 24 bit quantization) to 16 bit quantization. We restricted our analysis to two taxa: common dolphins (*Delphius capensis* and *D. delphinus*) and bottlenose dolphins (*Tursiops truncatus*). Common dolphins were the more challenging taxon due to their tendency to aggregate in large numbers with many simultaneous whistles, resulting in numerous whistle crossings. We selected 20 audio sequences that had time-frequency contour annotations created by analysts with the aid of an interactive toolkit [171]. We used 14 of these recordings for training and the remaining ones for test. This resulted in a total of 7,161 whistle contours in the training sequences and 911 whistle contours in test sequences.

We created log-magnitude spectrograms from discrete Fourier transforms of 8 ms Hamming windowed frames (125 Hz resolution) advanced every 2 ms. We restricted the dynamic range of the resulting spectrogram to a floor of 0 and a ceiling of 6 (an intensity range of 0 to 120 dB rel., uncalibrated and unadjusted for frequency bin width) on the basis of empirical results. We normalized spectrogram values to the interval [0, 1]. We then frequency-limited these data between 5 and 50 kHz (361 frequency bins), the range of most delphinid whistles and their harmonics.

We partitioned spectrograms into 100 ms by 6.25 kHz (50 x 50 time-frequency bins) patches.

In the training data, patches were formed by tiling the spectrogram into patch-sized regions that overlapped by 50%. Patches were split into whistle-positive and whistle-negative groups. There were 74,112 positive patches and we randomly selected an equal number of negative patches. We used these data, or random samples thereof, to train the deep contour model.

We also examined whether the model could learn from alternative data sources. To do so, we examined three different sources: frequency contour information provided by the DCLDE 2011 annotation data (no spectrograms), a set of analyst-annotated image edge annotations from the Berkeley Segmentation Dataset (BSDS 500) [4], and a set of edges generated automatically from the BSDS 500 images by a Canny edge detector [44]. Regardless of the source, we injected the annotations into whistle-absent spectrogram patches.

We used a similar processing chain to process data for testing, with additional steps as outlined below.

2.3.4 Deep Representation of Whistle Contours

In the experiment without data synthesis, we used analyst ground-truthed whistle (WGT) annotations of real data to determine the model's capacity to learn to represent small sections of whistles when training data are relatively abundant. Training data consisted of the 148,224 positive and negative spectrogram patches and corresponding 50 x 50 whistle confidence labels, which we set to 1 in each timefrequency node that contained a whistle and to zero otherwise. We trained a fully convolutional deep network [223] with stochastic gradient descent from these data.

When using the trained network to extract whistles, we partitioned the spectrograms of the test dataset into nonoverlapping patches. The network produced confidence maps that predicted the location of whistle energy in the input spectrogram patch. We reassembled these confidence maps into a spectrogram-like structure that we passed to a peak processing algorithm [171] which produced a set of hypothesized whistles. The choice of post-processing algorithm was one of convenience, and we expect that other peak-based assembly algorithms will yield similar improvements when peaks are selected in a more reliable manner.

Similar deep models have been used to detect edges or boundaries in images, and were much more accurate than traditional edge detection methods when applied to standard benchmarks (e.g., BSDS 500 [4]). The learning in such deep models, however, requires thousands of human-annotated image-contour pairs.

By restricting our analysis to small spectrogram patches and eliminating large segments of the spectrogram in which whistles were absent, we substantially reduced class imbalance. The convolutional filters were no larger than 10 ms by 375 Hz, and the network had a receptive field of 46 ms by 2.875 kHz. Whistle fragments in the resulting receptive field had relatively simple shapes. Longer whistle fragments would have had a more complex distribution that is likely to present greater challenges to model in a larger receptive field. Small patches and receptive fields allowed us to exploit multiple methods for generating a sufficient volume of synthetic training data to learn to predict confidence maps on the basis of little to no whistle data.

2.3.5 Synthesizing Training Data

We developed methods to generate synthetic training data to minimize human effort and increase the volume of training data. We injected primitive shapes (Fig. 2.3) into 100 ms by 6.25 kHz spectrogram patches of whistle-absent recordings, which provided realistic examples of ocean ambient sounds (Fig. 2.4). In these experiments, we selected these examples from our whistle-negative patches, but one could use this method to synthesize data for a novel recording environment.

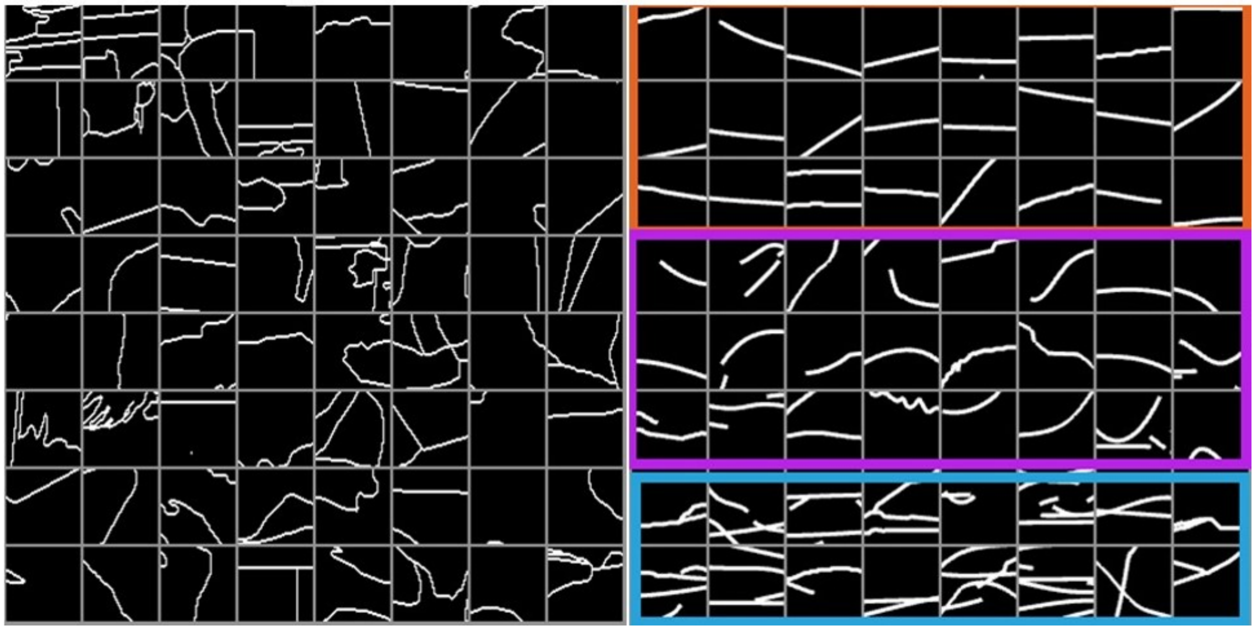


Figure 2.3: Sample of shapes from which the implicit dictionary was derived. Left: shapes in natural images, extracted from the boundary maps in the Berkeley Segmentation Dataset 500. Right: shapes derived from whistle annotations in time-frequency spectrograms; linear, curved, and crossed whistles are highlighted by orange, magenta, and blue boxes, respectively.

We explored two methods for data generation. First, we used contours collected from the computer vision domain to create synthetic data. These were either analyst-annotated boundaries of images in the BSDS 500 data [4] or edge segments that we detected automatically with Ding et. al.'s modifications to the Canny edge detector [44]. These contours contain many shapes similar to very short segments of delphinid whistles, although scenes containing anthropogenic landmarks (e.g., buildings) can have edges with sharp corners or vertical excursions that are not typically present in delphinid whistles. Second, we used samples of the humananalyst annotated whistle contours in the DCLDE 2011 data. We created synthetic data, injecting these accurately captured whistle shapes into spectrograms of ocean sound data.

We created binary label masks (Y) for each synthetic example (1 indicated whistle presence) and convolved them with a random Gaussian filter (G) to blur each binary mask. We aligned the blurred binary contour mask with a whistleabsent spectrogram patch to generate whistle-

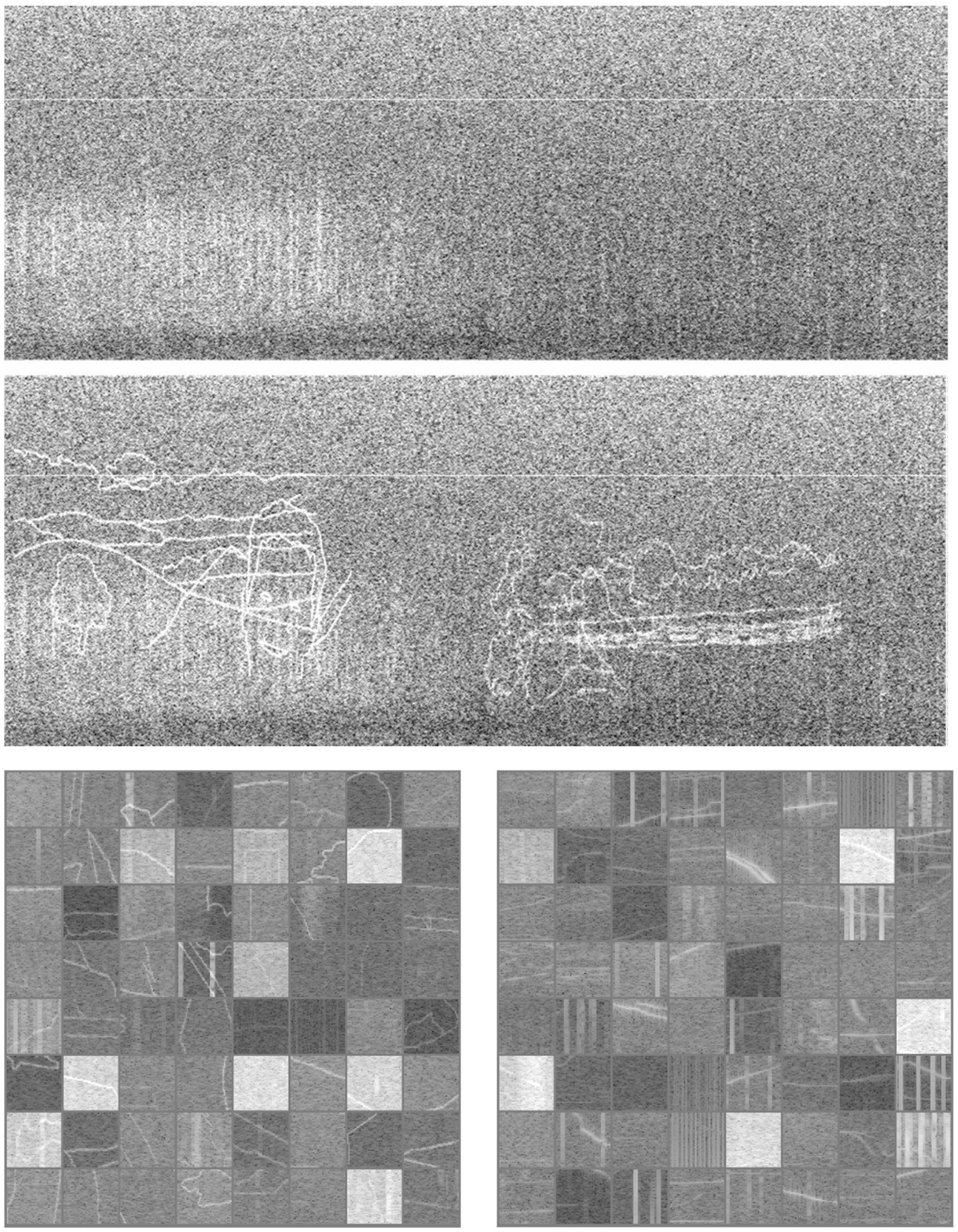


Figure 2.4: Example of data synthesis. Top: background spectrogram. Middle: spectrogram overlaid with natural image edges. Bottom: synthetic patches from BSDS 500 labels (left) and DCLDE 2011 whistle contours (right).

present training samples. We obtained the synthesized sample X' via

$$X' = X + \alpha(Y * G) \tag{2.1}$$

where X is a background spectrogram patch, α is a random intensity parameter drawn from a uniform distribution over [0.03, 0.23], and $*$ is the convolution operation. This results in signal strengths varying over 24 dB. The signal-to-noise ratio (SNR) of these signals was dependent on the noise levels in the spectrogram patches into which they were blended and the randomized signal intensity, and these values were representative of typical SNR values in real data. Our sample synthesis method reduced or, in some cases, eliminated the need for humans to annotate whistle contours (Fig. 2.4).

2.3.6 Simultaneous Learning and Sample Synthesis

Traditional learning-from-synthesis methods usually generate many samples to ensure effective learning. However, it is unclear how to determine the minimum number of synthesized samples for a particular application. Moreover, this strategy assumes that all examples are equally important and generates a homogeneous set of synthesis samples. Providing equal weight to examples that are easy and difficult to learn is inefficient, causing the network to expend more time considering examples that it already predicts well. Biasing the synthesis process to produce samples in proportion to their difficulty should increase the effectiveness of the learning process.

We developed an integrated approach that adaptively generates samples while training the network. This process focuses on those spectrograms or contour segments that were not well modeled by the network. We based our two-stage learning algorithm on the standard mini-batch gradient descent method. In the first stage, we restricted training contours and sound patches to a small subset (6.25%) of the available data, from which we generated synthetic whistle positive examples. We also included an equal number of randomly selected negative examples. We used these positive and negative samples to train an initial network. As we describe in the experiment section, use of this small amount of data with standard training techniques (no synthesis) yielded confidence maps with low recall (Fig. 2.5).

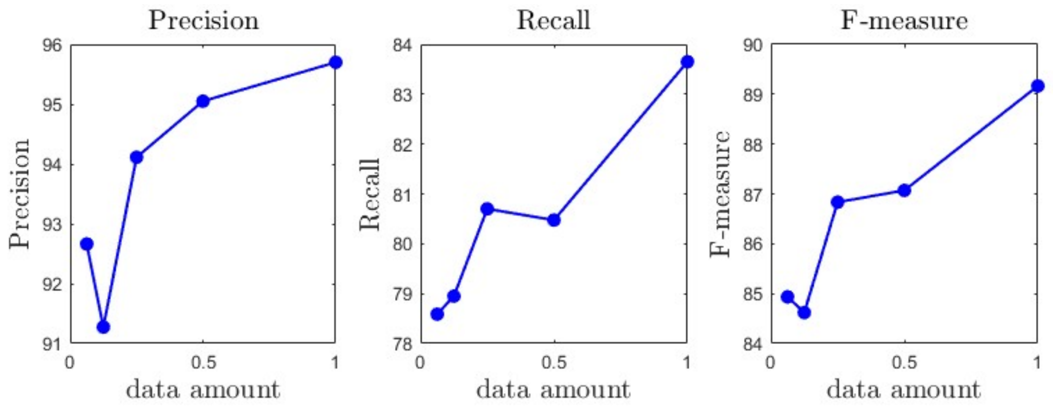


Figure 2.5: Confidence map metrics that were based on 6.25, 12.50, 25.0, 50.00, and 100% of the 148,224 training patches used in WGT.

The second stage was an iterative process with multiple training epochs. At the beginning of each epoch, we used the network weights from the previously completed epoch to produce confidence maps of the validation data. Validation data consisted of positive and negative spectrograms containing the non-synthesized data that were available to the algorithm (the 6.25% subset of the annotations). We computed the recall rate of the confidence maps, which allowed us to identify which positive examples were more difficult for the current network to classify. We assessed recall for a sample as

$$\mathrm{R}(z) = \frac{\sum\sum(I_{\geq 0.5}(C_z) \cdot Y) + \varepsilon}{\sum\sum(Y) + \varepsilon} \tag{2.2}$$

where C_z is the confidence map predicted from z, I is an indicator function thresholded at 0.5 that binarizes C_z, Y is the label mask, $\cdot$ is the element-wise (Hadamard) multiplication operator, and is a small value to prevent division by zero. The double summation is over the rows and columns of the confidence map. We chose to use recall rates instead of precision rates to retain as many contour candidates as possible during the early stage of the learning process.

We next used probabilistic selection to generate a new set of training data with roughly equal numbers of positive and negative examples. Let $p(z)$ be the probability that training sample z is selected. Let $R(z,t)$ be the recall for sample in the t^{th} epoch. Then the probability of selecting a negative N or positive P sample z at epoch t is

$$p(z) = \begin{cases} 0.5\frac{1}{|N|} & z \in N \\ \\ 0.5\frac{1-\mathrm{R}(z,t-1))}{\sum\limits_{\zeta \in P}(1-\mathrm{R}(\zeta,t-1)))} & z \in P \end{cases} \tag{2.3}$$

which biases selection of positive examples towards those with lower recall. We used the whistle annotations associated with selected positive examples to synthesize new examples. This enabled the network to observe the difficult pattern in varied contexts. This adaptive learning process can synthesize difficult samples over training epochs and improves performance.

In summary, the key components of our learning algorithm include sampling difficult examples of primitive whistle shapes from $p(z)$, synthesizing new samples for the selected shapes, and updating the network parameters on the basis of the new synthesized samples. In this way, network learning and sample synthesis are alternated until convergence.

2.3.7 Implementation

Network Design. Our network has 10 convolutional layers and each hidden layer has 32 channels. We added batch normalization layers behind all convolutional layers on residual branches. We used parametric rectified linear unit (Prelu) layers [72] as nonlinear layers behind the first batch normalization layers on the residual branches. The first and last convolutional layers had a filter size of 5 and zero padding size of 2; the remaining convolutional layers had a filter size of 3 and zero padding size of 1. The eight layers in between formed four residual blocks [71]. All layers had a stride size of 1. This simple architecture had a first-layer convolutional filter of size 50 ms by 3.125 kHz region in the time-frequency domain. Subsequent layers had smaller 25 ms by 0.625 Hz filters. We did not use pooling in this architecture, which makes predictions on individual time-frequency nodes.

Extracting Whistles From Confidence Maps. In peak processing whistle extraction methods, extraneous sound sources in the spectrograms frequently produce false positive peaks. When there is enough structure in the extraneous above-ambient peaks or the distance between peaks is short, the extraneous peaks can lead to false-positive whistle contours. Similarly, a high number of missed peaks can lead to false negatives. A more-reliable method of identifying peaks on the basis of contextual cues would dramatically improve the robustness of such algorithms. We replaced the spectrogram input with a confidence map of the probability that each time-frequency node is part of a whistle contour.

We employed an existing graph search algorithm [171] that organizes time-frequency peaks into a graph to test the reliability of our system's peak estimation and its capacity to improve peak tracking algorithms by exploiting contextual cues in the deep network. We applied this post-processing step to the test data after the confidence map patches were reassembled into a spectrogram-like structure and whistle energy was identified with a threshold τ of 0.5.

At each time step, whistle energy in the confidence map either initiates a new graph or extends terminating nodes of existing graphs. Extension is chosen when the new peak is along a reasonable trajectory predicted by a low-order polynomial fit of the graph path near a terminating node. Graphs can bridge gaps in detected peaks up to a userestablished time interval, reducing the impact of missed peak detections. When a peak is added to a pair of graphs, the graphs are merged. Graphs that have not been extended within a specified time are considered closed. Internal nodes of closed graphs with multiple inputs and outputs represent potential whistle crossing points. We used an analysis of forward- and backward-predicting polynomial fits to determine which branches of the graph should be placed together in the hypothesized whistle contours.

2.4 Experiments

2.4.1 Evaluation Protocol

Metrics. To assess the quality of the confidence maps before whistle contour extraction, we employed the BSDS 500 benchmark tools and protocol [4] to calculate precision and recall. We thinned our ground-truthed confidence maps to 1pixel wide and compared them with a predicted confidence map binarized by 30 thresholds between (0, 1). We applied all default parameters in the evaluation tools.

We used three metrics to evaluate and compare the quality of discrete whistle contours predicted by the complete contour extraction system: (i) recall, the percentage of validated whistle contours that were detected, (ii) precision, the percentage of detections that were correct, and (iii) F1-score, or the harmonic average of precision and recall. We determined success or failure of whistle extraction by examining the set of expected analyst detections as described in [171]. Briefly, for each analyst-annotated whistle contour, we checked whether any of the detections overlapped in time. If so, we examined whether each overlapping detection matched the analyst annotation. We considered that a match occurred if the average deviation in frequency between the two contours was < 350 Hz and the analyst detections were ≥ 150 ms in duration, with a signal to noise ratio ≥ 10 dB over at least a third of the whistle. When detection of a whistle was not expected (too short or low intensity), we discarded the matched detections, and they did not contribute to the metrics. We counted unmatched detections as false positives. Due to the multiple decision criteria, we evaluated the algorithm at the single operating point used in [171] and DCLDE 2011.

Dataset. We evaluate the proposed methods using the DCLDE2011 dataset [32], including 20 audio sequences with whistle contour annotations. We use 14 sequences for training and the other 6 sequences for testing. Whistle contours are manually annotated with the aid of an interactive toolkit [171]. There are a total of 7161 whistle contours in training sequences and 911 whistle contours in testing sequences. We also use the image edge annotations in BSD500 [134] for data synthesis and evaluate how these natural image contours can be used to improve whistle contour methods. Audio sequences are transferred into log-magnitude spectrograms which are analogous to two-dimensional images. All spectrograms extracted from training sequences are cut into 50-by-50 pixels patches. This leads to a total of 74112 positive patches and 74112 negative patches, used for training the proposed deep contour model.

Experiments. We implemented five variants of our whistle contour extraction method to quantify the contributions of the deep contour model, data synthesis, and recall-guided learning. All but one of these variants used human or machine-detected annotations to synthesize training samples. We compared the variants to the baseline method [171]. We used a fully convolutional network, trained on the complete set of ground-truthed whistle data (WGT), to assess the learning capacity of the deep contour model. The remaining variants trained the network exclusively with synthesized data. EdgeGT used all analyst-annotated edges from the BSDS 500 data [4]. The training data in EdgeCanny were not analyst annotations, but automatically generated edges from the BSDS 500 images. The remaining variants were trained with data synthesized from delphinid whistle annotations. µWGT employed a small amount of whistle contour time-frequency information from the 2011 DCLDE data (we did not use audio data containing real whistles) to synthesize spectrogram-contour samples. For μWGT, we simulated limited annotated data by randomly selecting 6.25% (4,632) of the patches in WGT. The analyst contours associated with the selected patches were used to synthesize samples. μWGT-RG extended μWGT with the our recall-guided learning-with-synthesis algorithm. We used the confidence maps generated by the above networks as inputs to call the graph-search method [171] to generate whistle contours (Fig. 2.2).

We used Pytorch [157] for all our experiments. We trained all network variants with the

Adam optimizer [98] with 600,000 iterations. We used Charbonnier loss [20] for network gradient calculation:

$$Loss(\hat{y}, y) = \sqrt{||\hat{y} - y||_2^2 + \varepsilon}. \tag{2.4}$$

where $\cdot$ is the squared L2 norm, $\hat{y}$ is the network output, y is the ground-truthed data, and ε is $1x10^{-3}$. We used an initial learning rate of 0.001, and decayed the learning rate by a factor of 0.1 every 250k iterations and initialized the networks with the Kaiming Normalization [72]. The μWGT-RG network was initialized to the final μWGT network.

2.4.2 Analyses and Results

Number of Training Samples. We first evaluated how the quantity of annotated whistle contours affected the prediction of confidence maps. System performance increased as the volume of training data increased (Fig. 2.5). Training an effective deep contour model required a large number of annotated samples. The largest performance increase occurred as the selected proportion of training data approached 25%, suggesting that the data have some redundant patterns. Our evaluation suggested that it is valuable to synthesize many training samples.

Prediction Execution Time. We tested and ran our algorithms on a workstation with an NVIDIA GTX 1080Ti GPU. Training required about 8 hours, with some variation among model variants. Testing 1 s of data required about 10 ms for network prediction plus 300 ms for the graph search. Our method can extract whistle contours approximately 3 times faster than real-time.

Quantitative Results. We evaluated our system against portions of the DCLDE 2011 data [32] that contained 911 test whistles meeting the SNR and duration selection criteria described above. There were 354 bottlenose dolphin and 557 common dolphin whistles.

For confidence map evaluation (Fig. 2.6), WGT, the method designed to test model capacity with larger amounts of training data, dominated the precision-recall curves of the data synthesis methods. The synthesis methods based on delphinid whistle contours, μWGT and μWGT-GT, generally performed better than EdgeGT and EdgeCanny, which relied on image edges. EdgeGT and EdgeCanny, however, had higher precision than the other synthesis methods at lower recall.

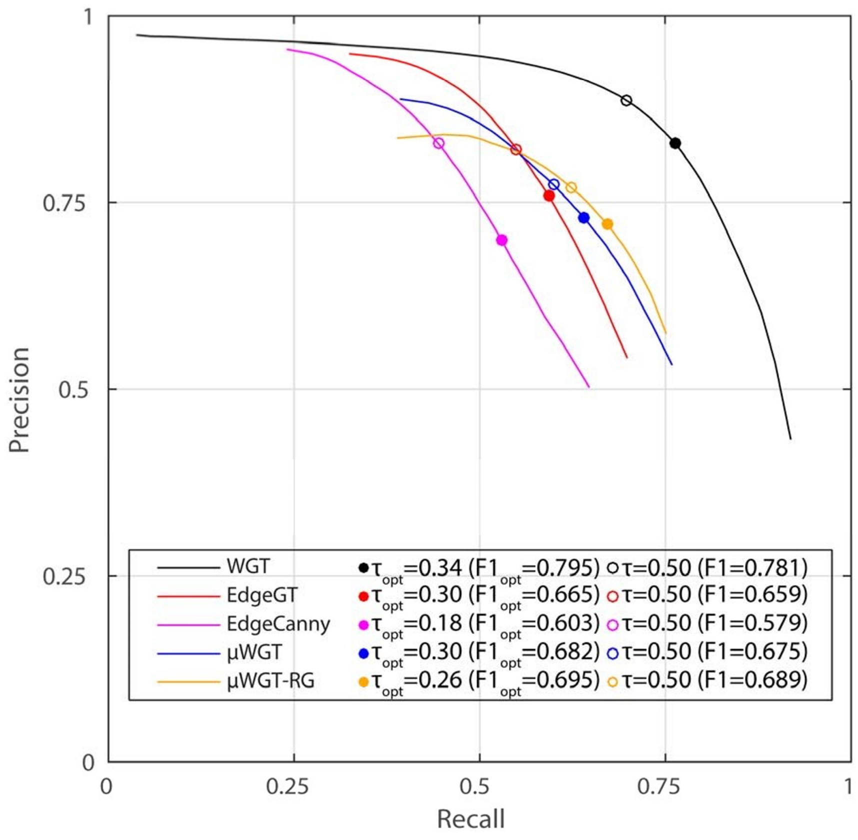


Figure 2.6: Performance (precision and recall curve) of the five deep models predicting time-frequency nodes containing whistle energy. Circles indicate operating points (at threshold τ) along the curve. Filled circles: optimal F1 performance. Open circles: operating point used when extracting whistles with the post-processing peak assembly algorithm.

For the two-stage whistle extraction system, we compared our previous work [171] to systems that used the same time$\times$frequency peak processing rules, but relied on peak predictions from our confidence maps (threshold 0.5, Table I). The scores of the graph search algorithm were lower than those reported in [171] because we focused on a difficult subset of the data. The average F1-score of the fully trained model, WGT, was 0.250 greater than that of the graph search method without the use of confidence maps.

All variants of the models that used synthesized data outperformed the baseline version of the graph search method. The F1-score results of EdgeCanny suggested that networks can outperform existing techniques by a considerable margin of 0.088 ($\sim$14% improvement) without the use of analyst annotations. The use of image edges is particularly useful when no whistle contours are available. The results of μWGT and μWGT-RG demonstrated that synthetic whistles based on real whistles improved the F1 scores, and that recall-guided learning could produce further improvements.

Qualitative Results. Fig. 2.7 represents a particularly challenging acoustic scene for whistle extraction systems. These data contained delphinid whistles and signals from a shipboard echosounder. Analyst annotations (top row) are compared with the baseline graph search (middle row) and the same graph search with peaks identified by WGT generated confidence maps (bottom row). This sequence, like many others in the DCLDE 2011 data [32], included changes in sound regime, anthropogenic signals, and other types of acoustic clutter. Our deep learning method of identifying whistle energy within a spectrogram had greater precision and recall than standard peak selection mechanisms such as the one used in the graph algorithm. The original graph-based method produced extraneous detections due to false positives in the peak prediction and missed a number of the whistles that the deep whistle contour detector predicted.

Discussion. Although whistle contours have complex patterns, the local structure of most whistles is relatively simple. The inner layers of the network may have implicitly captured the representative atoms of a dictionary of whistle segments and reconstructed confidence maps from them. Because there was no attempt to minimize the number of implicit representations, we expect that an implicit representation of the dictionary will be overcomplete.

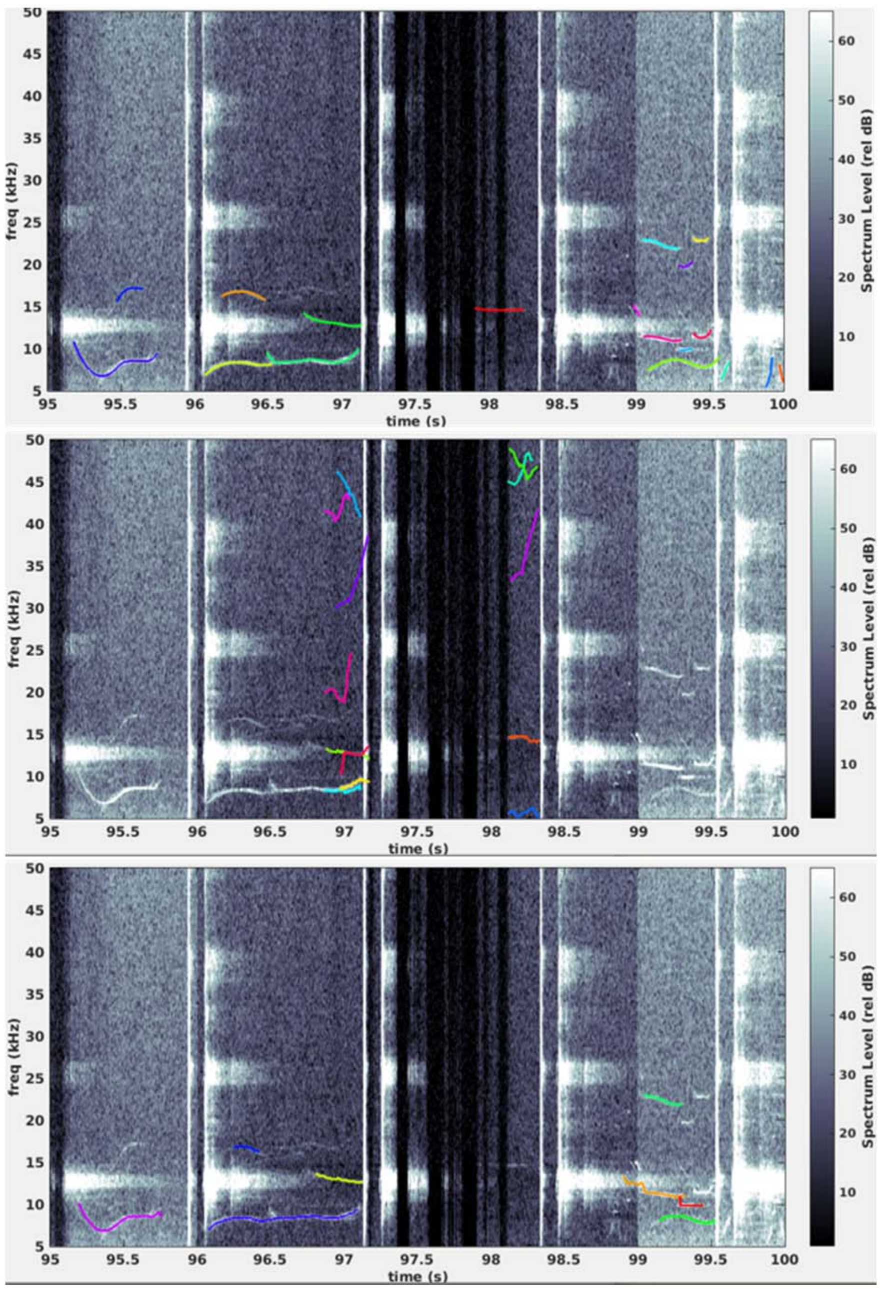


Figure 2.7: Whistles, annotations, and predictions in the presence of an echosounder (repeating signal with broad bandwidth). Whistles are overlaid with randomly colored annotations. Annotations were produced by experienced human analysts (top row), predictions from the graph search method [171] (middle row), and predictions from the WGT method, which replaces the graph search signal processing with confidence maps (bottom row).

The success of such a dictionary-based method depends on the quality of the retrieval and reconstruction procedure. Alternatively, the method simply may be learning to leverage the local context to improve the prediction of when timefrequency nodes are part of a larger whistle structure. Regardless, it is clear that the network learns to find whistle elements while ignoring distractors such as echolocation clicks (Fig. 2.2; clicks appear as vertical lines in the input spectrogram and are absent in the confidence map).

911 ground truth whistles			
Contour extractor	Precision	Recall	F1-score
Graph-Search	0.634	0.633	0.634
WGT	0.956	0.822	0.884
Synthesis-based experiments			
EdgeGT	0.913	0.638	0.751
EdgeCanny	0.900	0.603	0.722
μWGT	0.900	0.673	0.770
μWGT-RG	0.924	0.693	0.792

Table 2.1: Quantitative results of whistle contour extraction. Testing sequences for both long-beaked common dolphin and bottlenose dolphin are used. GT_N: number of ground truth whistles in testing files. DWC-I uses all the annotated spectrograms and other variants use only synthesized spectrograms. Bold: best results while not using annotated spectrograms. See texts for more details.

2.5 Conclusion

Although whistle contours have complex patterns, the local structure of most whistles is relatively simple. The inner layers of the network may have implicitly captured the representative atoms of a dictionary of whistle segments and reconstructed confidence maps from them. Because there was no attempt to minimize the number of implicit representations, we expect that an implicit representation of the dictionary will be overcomplete. The success of such a dictionary-based method depends on the quality of the retrieval and reconstruction procedure. Alternatively, the method simply may be learning to leverage the local context to improve the prediction of when timefrequency nodes are part of a larger whistle structure. Regardless, it is clear that the network learns to find whistle elements while ignoring distractors such as echolocation clicks (Fig. 2.2; clicks appear as vertical lines in the input spectrogram and are absent in the confidence map).

Chapter 3

Learning-based Whistle-extraction Data Generation

3.1 Introduction

3.1.1 Background

Spectrograms in the time $\times$ frequency domain can show signal structure and are frequently used in audio analysis [160] . Patterns in spectrograms are used for sound event classification [164], bird song recognition [88], music genre classification [103], automatic music transcription [170], speech emotion recognition [238], and other tasks. Many acoustic signals have frequency-modulated (FM) components that are visible in spectrograms. Examples include human speech [230], human singing [202], cries of newborns [139], vocal melodies [126], and whale calls [171]. In this work, we concentrate on whistles, the characteristic FM tonal calls of toothed whales.

Whale calls are used to study species identity [54, 86], individual identity [82], behavior [195]

[187], communication and social activities [196], and density and abundance [84]. Because whistles appear in spectrograms as characteristic contour shapes (Fig. 3.1 top left), experts can manually recognize animals' occurrence and label whistles as polylines on spectrograms. Whistle extraction algorithms [171, 109, 132, 22, 60, 208] aim to automate this process and identify each whistle as a polyline. Such extraction is challenging because of the high spatial and temporal variation of ocean sounds. Signals to be analyzed can be affected by recording device characteristics, sea state and propagation conditions, animal behavior, vocalizations from non-target species, and anthropogenic sounds, such as shipping and sonar.

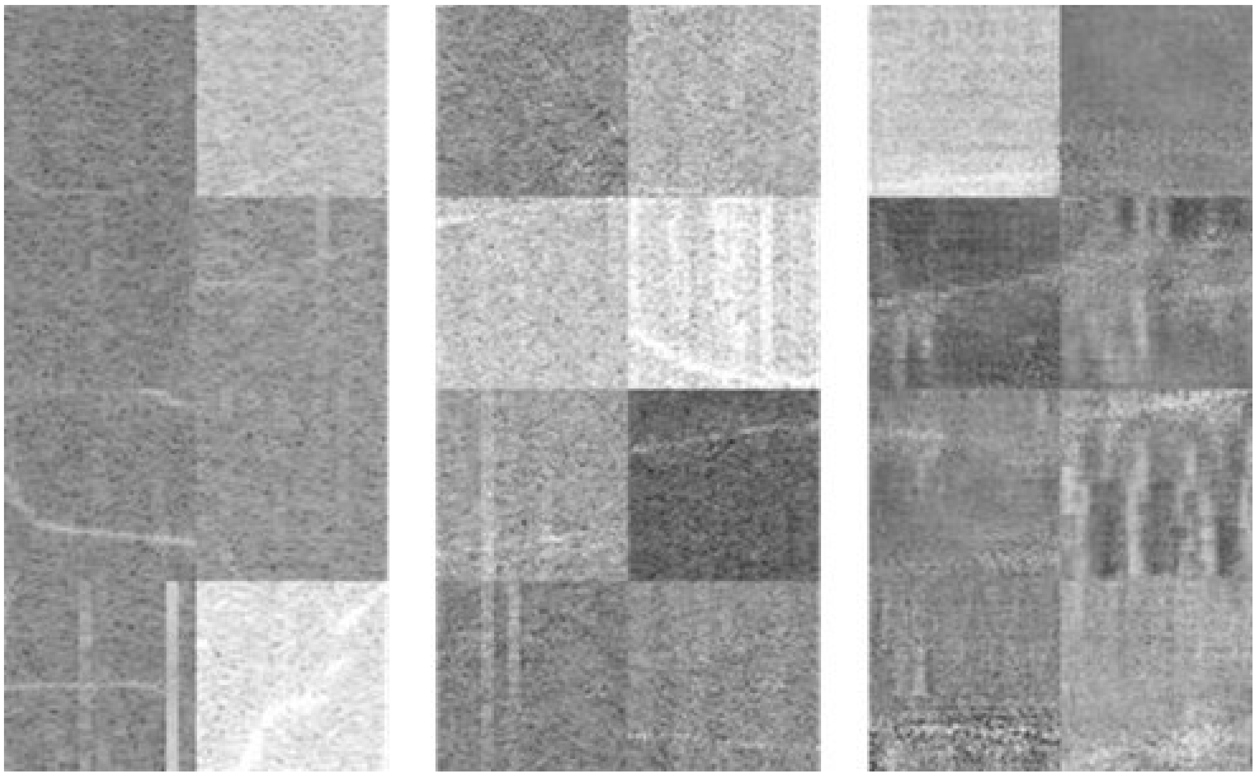


Figure 3.1: Examples of spectrogram patches of (i) real samples (left); (ii) samples generated by our stage-wise GAN; (iii) samples generated by a single GAN. Multiple 64×64 patches are concatenated for better visualization.

Traditional methods (e.g., graph search [171]) first extract the spectral peaks, i.e., bins with local maximum energy on spectrogram, and then track the trace of whistle signals on the spectrogram by polynomial fitting of peaks [171] or probabilistic modeling [60]. Recently, [109] adapted convolutional neural networks (CNNs) to extract whistles and achieved improved performance. Instead of using spectral peaks, [109] predicts the confidence associated with the probability that a whistle signal appears in each time $\times$ frequency bin, which is similar to semantic segmentation in computer vision. [19] then uses graph search [171] to connect bins that are likely to contain a signal. By learning from a large set of annotated samples, the whistle extraction model can recognize noise and whistle patterns, and improve on graph search and probabilistic model results by a large margin. However, the performance of learning-based methods may degrade significantly as the amount of annotated data decreases, and large datasets are not always available because whistle annotation is expensive and time-consuming. This motivates us to explore ways to synthesize whistle data cheaply with existing data by applying learning-based data generation methods.

3.1.2 Objectives

The primary focus of this work was to develop methods that improve whistle extraction models when data are limited, thereby reducing the amount of data annotation required to recognize whistles. Therefore, our experiments mainly addressed situations with few data, and we sought to mitigate the effect of overfitting and improve the model's transferability for recognizing tonal signals. Although there are many ways to reduce overfitting, e.g., semi-supervised learning [85] and regularization [110], we focused on data augmentation methods for two reasons. First, we seek a method that can be applied to all datasets of frequency-modulated signal, including those containing no unannotated data. Semi-supervised learning may not be applicable in this scenario. Second, we are interested in characterizing the distribution of whistle data and exploring the effect of novel data on extraction of tonal signals. Regularization terms may not provide insight in this context. We note that our data augmentation method may be combined with a semi-supervised framework or loss function regularization to further improve the system performance. Though it is interesting to have these techniques involved, it is beyond the scope of this work.

Common audio or image data augmentation methods usually transform existing data to acquire new data, e.g., by adding Gaussian noise [71], and the augmented samples may implicitly act as a regularizer for the training of deep models [39]. But the distribution of the augmented samples may not be similar to that of the original data; e.g., generated whistle data may have abnormal contour shapes or unrealistic background noise. Previous work [109] generated novel samples by adding whistle contours to negative samples (background noise that contains no whistle signals), which simulated the situation where the same whistles occur in different ocean environments. However, the generated data did not include novel whistle shapes or background noise patterns, which restricted the variance in the data.

In this work, our goal is to generate novel pairs of whistle data and labels. Although changes in noise affect vocalizations of many taxa [14], including toothed whales [7], we make the simplifying assumption that background noise is independent of whistle contours (contour-shape segmentation of whistles, which indicates the location of whistles on spectrogram and the whistles' frequency modulation). On the basis of this assumption, we decouple the synthesis of background noise and whistle contours. The generated whistle contours are used as labels for the model in [109]. Next, we add generated whistle signals with the desired contour shape to the spectrogram of background noise; i.e., we generate corresponding whistle data for the whistle contour.

We design our whistle generation algorithm as a series of three generative adversarial network (GAN) modules. The first GAN learns the ocean noise environment; it maps random numbers that have a Gaussian distribution to spectrograms representing background noise. The next GAN learns to map random inputs to spectrograms with whistle-like FM sweeps. The third GAN combines the outputs of the first two GAN modules, synthetic background noises and whistles, to obtain a synthetic whistle spectrogram. The generated whistle should follow the whistle contour's shape in the input. We employ an unpaired domain transfer framework, CycleGAN [244], to learn how synthetic noise and whistles can be merged into a synthetic spectrogram. While the original CycleGAN can generate slightly misaligned whistle signals from the desired contours, we exploit the whistle extraction network learned from annotated data to enforce the bin-wise consistency between generated whistles and input contours.

Another challenge is that GANs may not learn well with limited data. This may lead to corrupted synthesis, especially of the whistle contours. We observe that corrupted data have less confident predictions: the predicted probability is neither close to 0 nor close to 1, and thus the entropy is high. Accordingly, we introduce a method to prune such low-quality generated samples. Furthermore, because imperfect learning by GANs with few data may lead to discrepancies between the distributions of real data and generated data, we employ auxiliary batch normalization (ABN) layers [221] which separate the statistics of real and generated data to reduce the possible harmful effect of training with generated data.

3.1.3 Contributions

We made three contributions. First, we proposed the stage-wise composite GANs to generate novel whistle extraction data, including spectrograms and corresponding whistle contour labels. Our experiments showed that the proposed stage-wise GAN surpassed the vanilla GANs with respect to the visual quality of the generated data (Fig. 3.1 middle and right). Second, we designed a comprehensive strategy to use GAN-generated samples to improve whistle-extraction models. We set criteria to remove corrupted data and we redesigned the whistle extraction network by adding ABN layers to optimize the training with generated data. Third, we applied our proposed data augmentation methods to varied amounts of whistle extraction data and observed consistent and significant improvements. Although GAN frameworks have been used for spectrogram generation and data augmentation in audio recognition tasks [204], to our knowledge, this is the first work to apply GAN-based augmentation to audio spectrogram segmentation data.

3.2 Related Works

3.2.1 Whistle Contour Extraction

There are three main classes of methods for extracting whale frequency-modulated whistles. The first is models that predict the probability of whistle peaks conditioned on past observations. Examples of this class include tests of hypothesized spectrogram region distributions [36], Bayesian inference [66], Kalman filters [132], and Monte-Carlo density filters [171, 215, 59, 60]. The second class, trajectory-search methods, seeks energy peaks along the frequency dimension and connects those peaks along the time dimension on the basis of trajectory estimation [54, 171, 137]. Improved trajectory-search methods reduce excessive numbers of false positives by applying ridge regression to local contexts [95] or energy minimization algorithms to ridge regression maps [176].

In recent years, the third class, deep learning methods, has been applied to process tonal information. Early works included extraction of information from human speech [68] and music [11]. Deep neural networks were also applied to toothed whale whistles [86, 118], but the goal of these works was to classify a time segment to species or call type rather than to extract detailed time $\times$ frequency information. In [109], we proposed a deep neural network to extract time $\times$ frequency contours of individual whistles. We apply our proposed data augmentation system to the training of whistle extraction model developed in [109].

3.2.2 Generative Adversarial Networks

Generative adversarial networks (GANs) are a category of generative models. GANs are widely used for artificial image generation, e.g., face manipulation [148], compression noise removal [51], and generating images of people [125]. We adapted the methods from these computer vision tasks to generate realistic spectrograms that served as our novel training data. The landmark work on GANs [57] proposed one generator network that synthesizes samples ($G : X \rightarrow Y$, where X is a random vector and Y is a generated sample) and one discriminator network that learns to distinguish between generated samples and real samples. These networks are coupled in a zero-sum game with each network trying to outperform the other. Following [57], researchers have improved the network architecture of GANs [161] and objective functions [62] to stabilize the training of GANs. Those GANs implicitly learn the distribution of real samples, and novel data can be sampled from the distribution. We employ this type of GAN to generate novel spectrogram noises and whistle contours.

Another type of GAN tackles the image-to-image translation problem, aiming to learn a mapping ($F : x \rightarrow y$, where $x \in X$, $y \in Y$) between a source domain X and a target domain Y, e.g., transfer a horse in the image to a zebra. CycleGAN [244] extends this idea by leaning two mappings ($F : x \rightarrow y$, $G : y \rightarrow x$, where $x \in X$, $y \in Y$) without the need for pairwise correspondence between the elements of X and Y. This idea can be adapted to our task to generate spectrograms containing whistles, where X is the domain containing pairs of desired whistle contours and spectrograms with background noise, and Y consists of spectrograms with whistles and noise. Recent work improved the idea of [244] by adding a spatial attention mechanism [46] and image quality assessment term [23].

3.2.3 GAN-based Augmentation

GANs provide an option to generate novel data by learning the distribution of existing data and sampling data from the distribution, which is a valuable addition to the common augmentation techniques that are based on data transformation. Vanilla GAN models, which map random numbers to generated samples, have been used for data augmentation. [49] trained a GAN model to augment computed tomography (CT) images of livers for the classification of lesions. [133] applied a conditional GAN to augment samples from given categories and restore the balance of imbalanced image classification data. [12] applied progressively grown GANs (PGGANs) to a brain segmentation task, and the generator learned to synthesize the generated sample and corresponding segmentation labels. Domain transfer GANs have also been used for data augmentation. [76] applied CycleGAN to day-to-night image translation, which helped to improve the object detection model.

Despite the success of GANs in synthesizing visually appealing samples and augmenting existing data, there are still limitations of GANs for synthesizing high-quality augmented data, especially for pixel-wise regression tasks such as semantic segmentation. First, GANs usually suffer from mode collapses [235]: the generated samples may have lower variance than the real samples. Second, GANs may generate samples with artifacts or failure regions [154], which may especially hamper the training of pixel-wise regression tasks. A sample selection method may be required to choose high-quality samples from the GAN-generated samples [146]. Third, the training of GANs can be unstable, which results in different distributions of generated samples and real samples [1].

Therefore, GAN-based data augmentation usually requires improving the quality of generated samples. A common solution is to use real samples or computer graphics models in the generator network. In [3], the GAN learned to generate samples conditioned on real samples and random numbers. Similarly, Mu et al. [145] transferred synthetic images built by computer graphics models to realistic images, and the augmented samples improved models in estimation of gazes, hand poses, and animal poses. Another way to improve training of the GAN is to use supervision from target tasks. Waheed et al. [203] added an auxiliary classification head on the discriminator of GAN and used the classification loss to guide discriminator and generator learning.

Recently, stage-wise GANs were proposed to augment data for pixel-wise regression tasks. Pandey et al. [156] employed a two-stage GAN augmentation of cell nuclei segmentation data.

Their framework generates a cell nuclei segmentation mask in the first stage and images of nuclei in the second stage. Our proposed method is closely related to [156], and we further separate the learning of object appearance and the segmentation mask. This separation can be extended to other semantic segmentation scenarios. For example, when generating a scene containing road and cars, our framework may first generate the appearance of the road and car independent of the segmentation mask, then generate an image of the scene according to the segmentation mask and the appearance of the objects (road and cars). In this way, our framework explores the distribution of object appearance and provides variance in the appearance of objects in the generated image of the scene. Another improvement is that we employ the knowledge from segmentation networks to regularize bin-wise correspondence between generated samples and labels.

3.3 Methods

The objective of this work is to develop a data augmentation approach to generate novel data for whistle extraction. We treat the cropped patches from the time-frequency spectrograms as data samples, and we employ stage-wise GANs, which we call WAS-GANs (**W**histle **A**ugmentation **S**tage-wise **G**enerative **A**dversarial **N**etworks), to generate both negative samples (noise only) and positive samples (whistles in the presence of noise). Our techniques can be extended to other acoustic tasks or computer vision tasks, e.g., sound classification and semantic segmentation.

Fig. 3.2 illustrates the three stages of our sample generation approach. In Stage 1, a Wasserstein GAN with gradient penalty (WGAN-gp) [62] learns to produce the negative samples containing background noises. In Stage 2, we train another WGAN-gp model with the real whistle contour annotations to generate whistle contour segmentation masks. In Stage 3, we use a CycleGAN [244] to generate positive samples. The whistle signals are added to the negative samples obtained in Stage 1 according to contour shapes defined in Stage 2. The positive samples and segmentation masks are used as the whistle extraction data and labels, respectively. Both generated negative samples and positive samples are used to train the whistle extraction model, and the resulted whistle extraction performance is used to assess our GAN-based augmentation.

3.3.1 GAN-based Negative Sample Synthesis

We assume that the underwater background noise (negative samples) follows an implicit distribution. The generator learns the mapping between a multivariate Gaussian distribution and the distribution of negative samples. While many GAN models can learn this mapping, we chose WGAN-gp because its training is relatively stable [62]. The model includes a generator network, G, and a discriminator network, D. Network G maps a multivariate Gaussian random variable to generate negative samples. Network D estimates the Wasserstein distance between real samples and generated background noise (negative) samples. We denote P_r as the distribution of real data x; P_g as the distribution of generated data implicitly defined by $\widetilde{x} = G(z)$, where z is a random vector following the standard multivariate Gaussian distribution; and $\hat{x}$ as a randomly weighted sum of x and $\widetilde{x}$. The loss function for the discriminator network is defined as

$$L = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D(\tilde{x})\right] - \mathbb{E}_{x\sim\mathbb{P}_r}\left[D(x)\right] + \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[(||\nabla_{\hat{x}}D(\hat{x})||_2 - 1)^2\right] \quad (3.1)$$

where $\nabla_{\hat{x}}D(\hat{x})$ is the gradient of discriminator D's output on $\hat{x}$. This loss function encourages the discriminator to maximize the estimated Wasserstein distance between real and generated samples. The gradient penalty term $\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[(||\nabla_{\hat{x}}D(\hat{x})||_2 - 1)^2\right]$ enforces a soft version of Lipschitz constraint on the discriminator network. The loss function for the

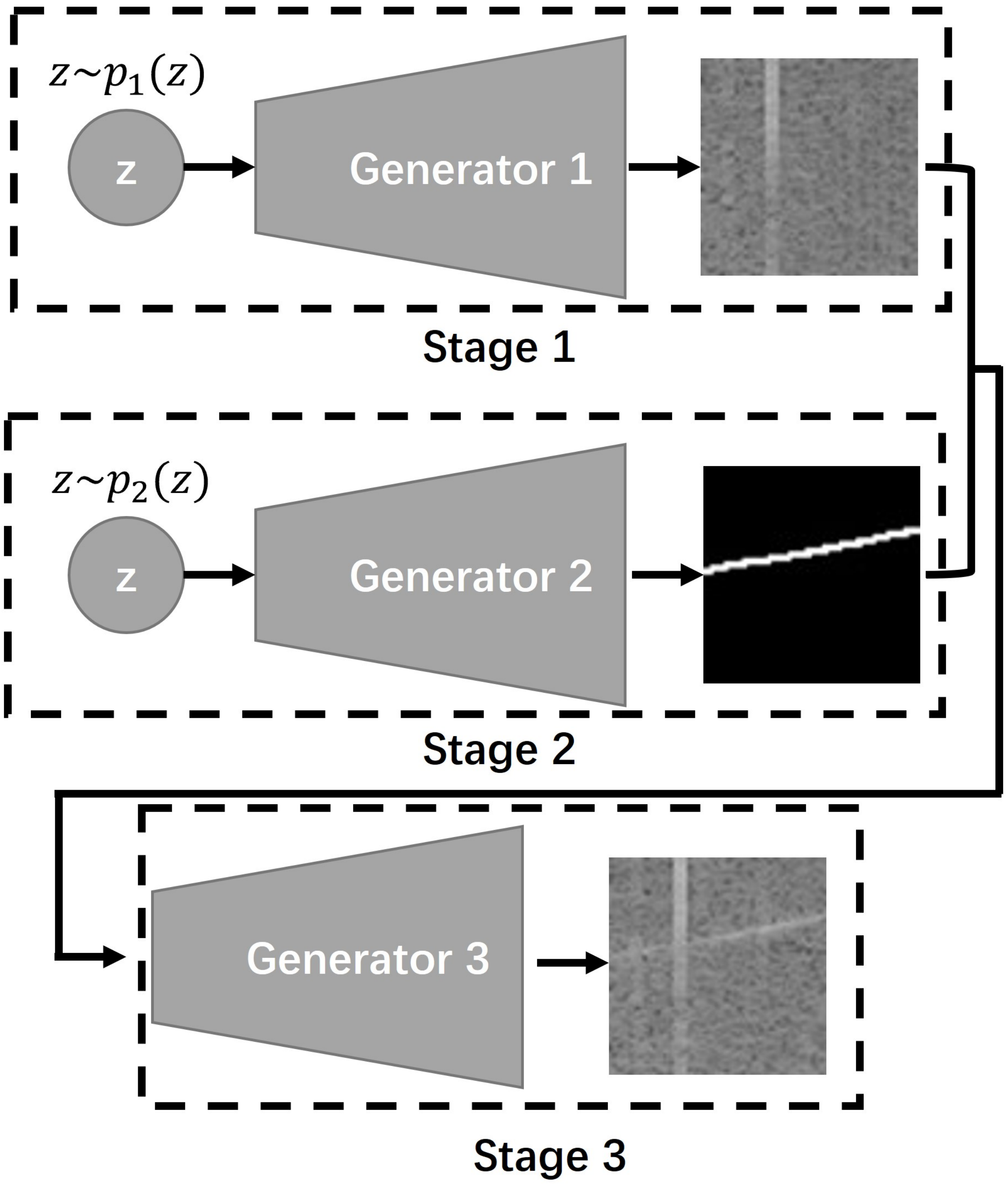


Figure 3.2: Sketch of the proposed stage-wise GAN frameworks. The first two generators produce a spectrogram patch of background noise and a spectrogram patch of foreground whistle contour, respectively. These patches serve as inputs for the third generator.

generator network is

$$L_G = \mathbb{E}_z \left[-D\left(G(z)\right) \right] \tag{3.2}$$

which encourages the generator to generate samples that have a small estimated Wasserstein distance from the real samples, i.e., to follow a distribution similar to that of the real data.

We split synthesis of positive samples (spectrograms containing whistles) into two stages: generation of whistle contours and injection of the whistle into synthetic background noise. In the first stage, we employ the same networks and loss functions as in Section 3.3.1, given the assumption that the shape of whistle contours is independent of the underwater environments.

In the second stage, we aim to generate positive samples according to the synthetic background noise and whistle contours. We treat this as an unpaired domain transfer task, which can be solved effectively by CycleGAN [244]. Our source domain, A, consists of pairs of negative samples and whistle contours, and the target domain B includes positive samples. We adopt the CycleGAN from [244] for our experiments, but any improved model readily can be used in our framework.

There are two sets of generator and discriminator networks in CycleGAN. G_A denotes the generator network that transfers samples from domain A to domain B, i.e., generates whistle with the desired shape on the background noise spectrogram. D_A denotes the discriminator network that distinguishes between real and generated spectrograms in domain B. G_B is the network that transfers samples from domain B to domain A, effectively separating the whistle contour from the background noise. Because we assume that the whistle contour and background noise are independent, we do not use a single D_B network for the joint distribution of whistle contours and background noise. Instead, we use two D_B networks for the marginal distributions, one to discriminate negative samples and one to discriminate whistle contours.

Instead of directly generating positive samples by G_A, we let G_A predict a residual term (whistle signals without background noises) to be added to the negative samples. By denoting a negative sample as I_N, a whistle contour as I_W, and the generated positive sample as I'_P, the process can be described as

$$I'_P = I_N + \gamma G_A(I_N,\ I_W) \tag{3.3}$$

where γ is a factor that controls the signal strength and accounts for variability in the received signal level. This parameter can simulate the variation in signal strength caused by variation in signal source strength or the distance between the animal and recording devices.

To enforce the bin-wise correspondence between generated positive samples and whistle contours, i.e., to avoid misalignment between generated whistle extraction data and labels, we use the whistle extraction models, which are trained on the same set of real samples as CycleGAN, to design a regularization term for G_A training. We call this term a loss function for the pixel-wise consistency, and represent it as

$$L_{consistence} = ||f(I'_P) - I_W||_1 \tag{3.4}$$

where f denotes the whistle extraction model and $f(x)$ is the model's output, a confidence map indicating the presence of whistle energy in each bin of the spectrogram, with an input x. This loss encourages the whistle signals to appear at the same position as the desired whistle contour.

To guarantee that the generated positive samples have the same background magnitude as the input negative samples, we also include the identity loss,

$$L_{identity} = ||G_A(I_N, 0)||_1 + ||G_B(I_N) - (I_N, 0)||_1 \tag{3.5}$$

where 0 indicates an empty whistle contour input, i.e., we do not want the CycleGAN to generate any whistles. We denote $(I_N, 0)$ as the concatenated I_N and empty whistle segmentation map. G_A should produce residuals of zero when there are no input whistle contours. We also use adversarial loss, L_{D_A}, L_{D_B}, L_{G_A}, L_{G_B}, and cycle consistence loss (L_{cyc}) from CycleGAN

$$L_{D_A} = (D_A(I_P) - 1)^2 + (D_A(I'_P))^2 \tag{3.6}$$

$$L_{D_B} = (D_B(I_N,\ I_W) - 1)^2 + (D_B(G_B(I_P)))^2 \tag{3.7}$$

$$L_{G_A} = (D_A(I'_P)\ -\ 1)^2 \tag{3.8}$$

$$L_{G_B} = (D_B(G_B(I_P)) -\ 1)^2 \tag{3.9}$$

$$L_{cyc} = ||G_B(I'_P) - (I_N,\ I_W)\,||_1 + ||G_A(G_B\,(I_P)) - I_P||_1 \tag{3.10}$$

where I_P refers to real positive samples. We simplify the notation of two D_B networks in one D_B function in the above equation. The full objective for generators is

$$L_G = L_{G_A} + L_{G_B}\ + \lambda_0 L_{cyc} + \lambda_1 L_{consistence} + \lambda_2 L_{identity} \tag{3.11}$$

where λ_0, λ_1, and λ_2 control the relative importance of the corresponding loss items. The full objective of the discriminator is

$$L_D = L_{D_A} + L_{D_B} \tag{3.12}$$

Ideally, D_A, D_B will assign 1 to real samples and assign 0 to generated samples with this training objective. G_A, G_B will try to fool the discriminators and generate realistic samples.

3.3.2 Whistle Extraction Model

We use the whistle extraction model from [109] as our baseline. This model, which is similar to a selective edge detection model, produces a confidence map of the whistle signals. Although the generated samples are visually similar to real samples (Fig. 3.3), the distributions of the real and generated whistle contour may differ due to the imperfect training of GAN when data are limited. This discrepancy decreases the accuracy of our whistle extraction model when we use the generated samples for data augmentation. Therefore, we use ABN layers [221]; i.e., we use auxiliary BatchNorm (BN) layers for forwarding generated samples and normal BN layers for real samples. We share the same convolutional layers for real and generated samples. By denoting the input sample as x, the whistle signal label as y, and the whistle extraction model as f, the loss without ABN can be described as

$$L = ||y - f(x))||_2 \tag{3.13}$$

The loss with ABN is

$$L = \frac{1}{(1+\lambda)}(||(y_{real} - f(x_{real}))||_2 + \lambda||y_{fake} - f_{abn}(x_{fake}))||_2) \tag{3.14}$$

where x_{real}, y_{real} are the real samples and labels, respectively, and x_{fake}, y_{fake} are the generated samples and labels, respectively. λ is a factor to adjust the weights of real data and generated data in loss calculation. $f_{abn}(x)$ denotes the output of the whistle extraction model for input x when the auxiliary BN layer is used in forwarding. We empirically find that ABN layers improve the whistle extraction performance when the distributions of the generated and real samples may be different.

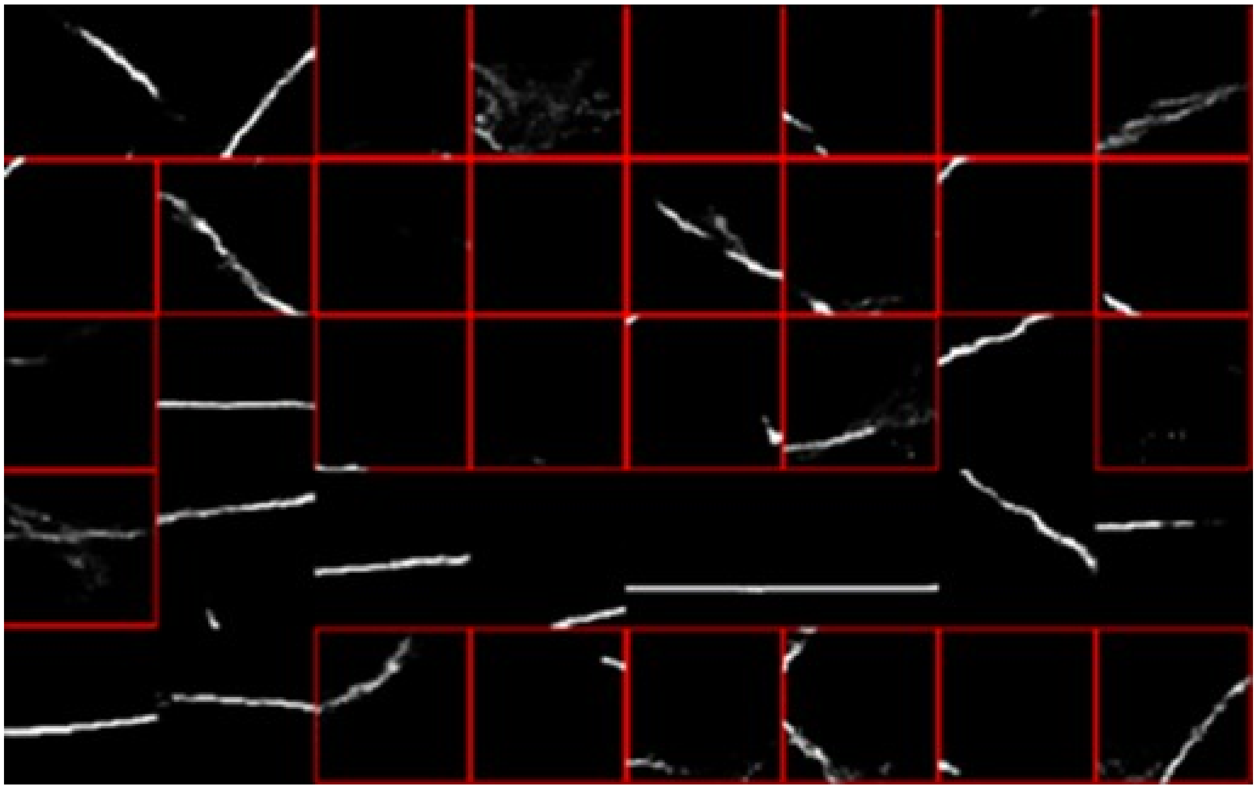


Figure 3.3: Illustration of whistle contour selection. Low-quality generated patches are highlighted by red bounding boxes. Multiple 64×64 patches are concatenated.

3.3.3 GAN-based Positive Sample Synthesis

The quality of GAN-synthesized samples is affected by the number of real samples available for training. The generator may synthesize poor-quality samples when the number of real examples used in GAN training is low. Fig. 3.3 provides examples of synthetic whistle contours when 2500 real positive samples are used for GAN training, including whistle contours that are of poor quality. Therefore, we designed two heuristic conditions for selecting high-quality generated samples. Denoting the value of an individual bin in the whistle contour patch as p, we select the sample for training the whistle extraction model when

$$\sum -plogp \ < T_e \tag{3.15}$$

and

$$\sum \delta(p - T_c) > T_p \tag{3.16}$$

where

$$\delta(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{3.17}$$

T_e is a threshold for the sum of the pixel entropy, so the first condition removes generated whistles with diffuse medium-intensity signals (high entropy). The second condition chooses samples in which more than T_p bins have intensity above T_c, allowing samples with short whistle fragments to be removed.

3.4 Data and Implementation

3.4.1 Datasets

We used the whistle extraction data from the 2011 workshop on detection, classification, localization, and density estimation of marine mammals (DCLDE 2011, available on the MobySound Archive [32]). These data contain recordings of calls made by five toothed whale species: long-beaked common dolphins (*Delphinus capensis*), short-beaked common dolphins (*Delphinus delphis*), bottlenose dolphins (*Tursiops truncatus*), melon-headed whales (*Peponocephala electra*), and spinner dolphins (*Stenella longirostris*). Whistle contours were

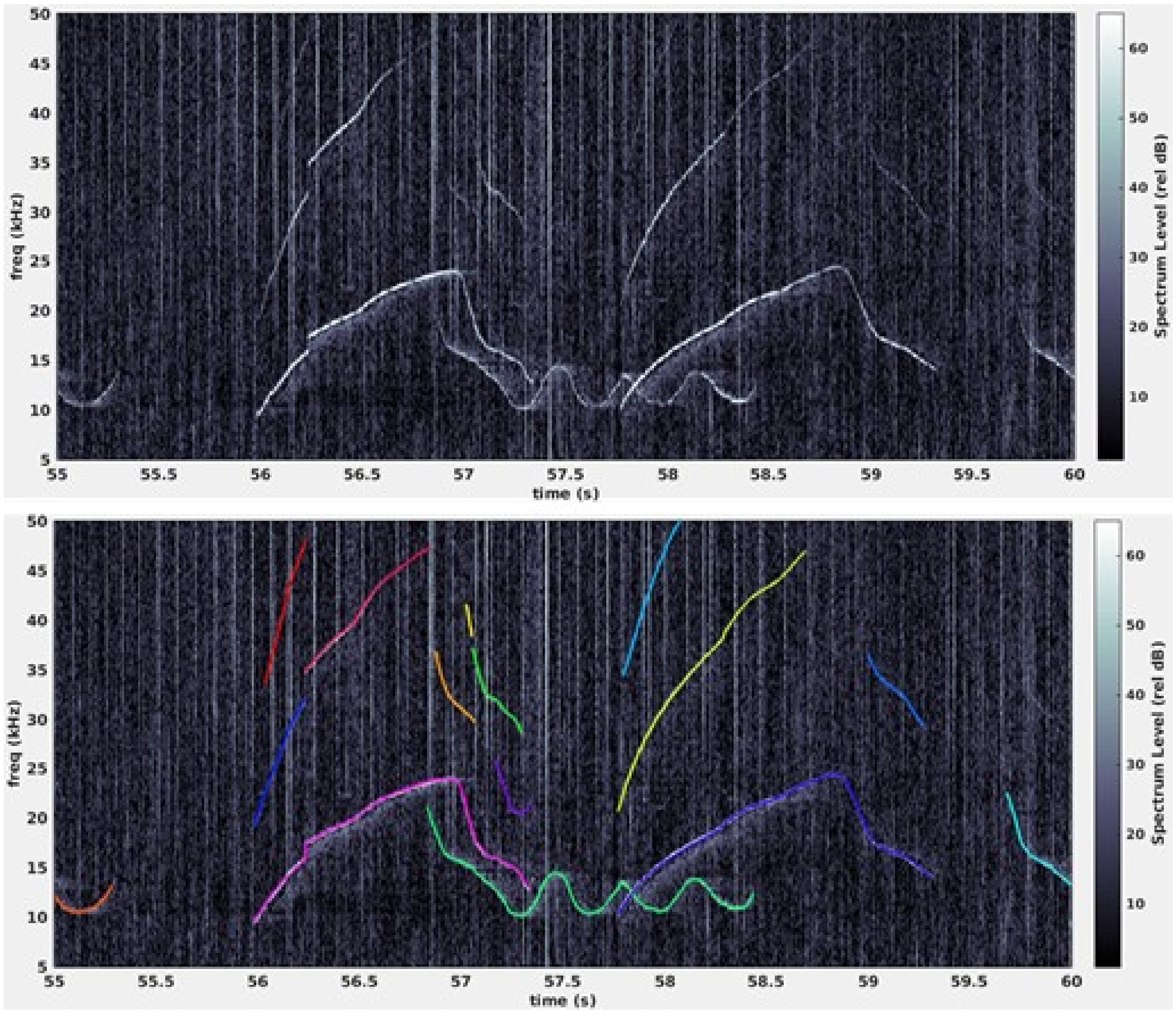


Figure 3.4: Illustration of whistle extraction. (Top) spectrogram visualized by *Silbido* [171]; (Bottom) extracted whistles, where each whistle is highlighted with a different color.

annotated by trained analysts across the 5-50 kHz bandwidth as described in [171]. We use 30 recordings from the 5 species to train and 12 recordings from 4 species to test. Short-beaked common dolphins are removed from evaluation because some of the files had annotation errors. The training data consisted of approximately 127 min of recordings with 12,539 annotated whistles. The test data (∼43 min of acoustic data) contained 6,011 annotated whistles.

We computed log-magnitude spectrograms for the whistle extraction model and the GAN-based data synthesis. We employed series of discrete Fourier transforms in spectrogram computation. 8 ms Hamming-windowed frames (125 Hz bandwidth) were computed every 2 ms, and we empirically restricted the dynamic range of the log_{10} magnitude spectrogram to the range [0, 6] (an intensity range of 0 to 120 dB rel.), i.e., we transformed the values <0 to 0, and those >6 to 6. We divided the spectrogram values by 6 which made them within [0, 1], and discarded the spectrogram values outside of the annotation frequency range of 5-50 kHz (361 frequency bins), which covers the frequency range of most delphinid whistles and their harmonics.

For network training, we partitioned the spectrogram into 64×64 patches, where each patch covered a time interval of 128 ms and frequency interval of 8 kHz. For the training data, we selected the positive patches with a sliding window with a 25 pixel step size across portions of spectrograms containing whistles, which led to 115,968 positive patches available for training. We randomly selected the same number of negative patches, which only contain noise, and combined them with positive patches as our training data (referred to as the full dataset). Most of our experiments used a subset of the full data (referred to as a reduced dataset). We describe the details of generating the reduced dataset in Section 3.5.1.

3.4.2 Networks and Algorithms

Whistle extraction network

We used the same network architecture as [109]. The model has 10 convolutional layers, including 1 input layer, 4 residual blocks (each block contains two convolution layers), and 1 output layer. The input layer and output layer use kernel size 5 and padding size of 2, and other layers use a kernel size of 3 and a padding size of 1. All hidden layers have 32 channels. The model input is a one-channel spectrogram and the output is a confidence map of whistle occurrence. The size of the output confidence map is the same as that of the input spectrogram.

We trained the whistle extraction model with an Adam optimizer (initial learning rate=1×10^{-3}, betas = [0.9, 0.999], weight decay=5×10^{-4}) for 1×10^6 and 3×10^5 iterations on the full dataset and reduced datasets, respectively. The learning rate was multiplied by 0.1 every 4×10^5 and 1×10^5 iterations for the full and reduced datasets, respectively. We set the batch size to 64, and we used 64 real samples and 64 generated samples in each iteration for data augmentation experiments. We used λ=1 in the loss function of Eq. 3.14 for our experiments with generated data, which make the generated samples have the same contribution of loss as real examples.

WGAN

We used the same WGAN architecture for the generation of whistle contours and negative samples. The generator network uses a fully-connected layer to output feature maps of size (512,4,4) from a 128-dimensional standard Gaussian distribution. Four groups of convolutional layers and pixel shuffle layers are used to gradually enlarge the feature map to 64×64. A Tanh layer is used to output the 64×64 patch. The discriminator network takes the generated samples and real samples as input, and outputs the Wasserstein distance estimation. It contains 4 convolutional layers with a stride of 2 and a fully connected layer. The networks are optimized by Adam optimizers (initial learning rate = 1×10^{-4}, betas = [0.5, 0.9], batch size = 64) for 3×10^4 and 5×10^4 iterations on the reduced and full datasets, respectively. In each WGAN training iteration, the discriminator is optimized for 5 steps while the generator is optimized for 1 step, where the network parameters are updated by applying the optimizer to one mini-batch of data in each step. For sample selection, we used T_e=70, T_c=0.5, T_p=64.

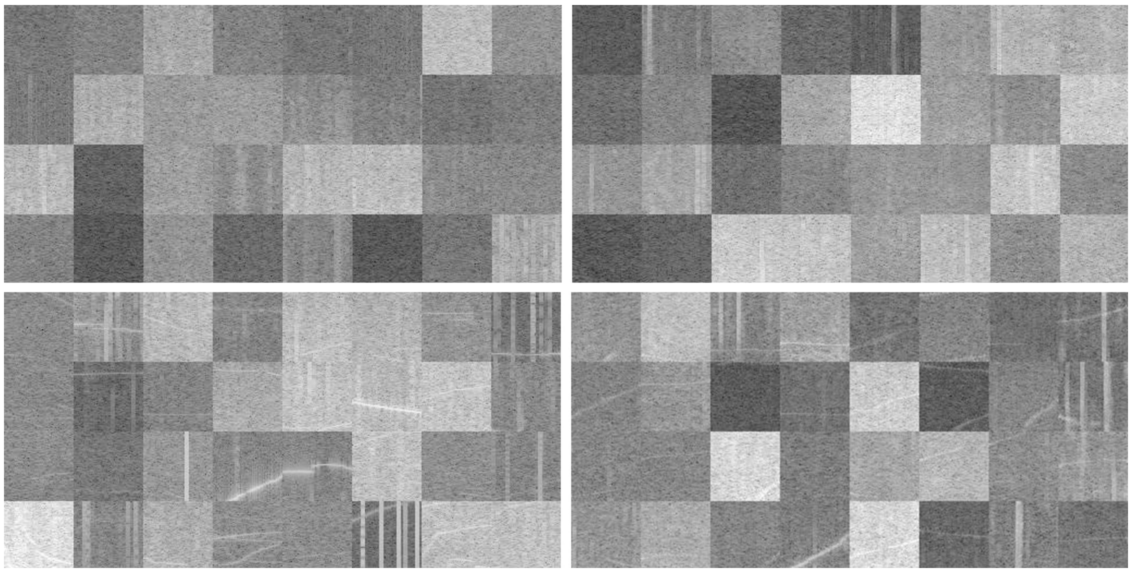


Figure 3.5: Real background noise samples (upper left); Our GAN generated background noise samples (upper right); Real whistle samples (bottom left); Our GAN generated whistle samples (bottom right). Multiple 64 $\times$ 64 patches are concatenated in each category for better visualization of the data variance.

CycleGAN

The GAN model that we used to add whistles on synthetic noise employs the CycleGAN architecture of [244]. The generators follow the U-Net [172] architecture, which has 6 U-Net blocks with a basic width of 64. InstanceNorm layers are used in the U-Net blocks. The discriminator is a fully convolutional network with 3 convolutional layers. We trained the generators and discriminators with Adam optimizers (learning rate = 2×10^{-4}, betas = [0.5, 0.999], batch size = 64) for 25,120 iterations (160 epochs for 10,000 real positive samples) for the reduced dataset and 50 epochs for the full dataset. We set λ_0=10, λ_1=0.5, and λ_2=0.5 for Eq. 3.11. We apply a random γ following a unified distribution between (0.5, 1.5) in Eq. 3.3.

Graph Search

We adapted the graph search [171] algorithm to the outputs of the whistle extraction network to predict individual whistles. This algorithm maintains sets of graphs, the nodes of which indicate the trace of predicted whistle contours. Multiple crossing whistles can be represented by a single graph. At each time step, local maximum points (peaks) on the confidence map are selected along the frequency dimension, and peaks with confidence greater than 0.5 are retained as candidate points. For each candidate point, the algorithm either initiates a new graph or extends terminating nodes of existing graphs. Extensions are made when the new node is along a reasonable trajectory predicted by a low-order polynomial fit of the graph path near a terminating node. Graphs that have not been extended within a specified time are considered closed. Closed graphs are removed from the current graph set. When a graph is of a shorter duration than a settable minimum whistle duration, it is discarded. Otherwise individual whistles are extracted from the graph on the basis of an analysis of graph vertices.

3.4.3 Metrics

Evaluation of Confidence Maps

We first assessed the quality of the whistle-energy confidence maps predicted by the whistle-extraction model. To do this, we utilized the BSDS 500 benchmark tools and protocol [4] to calculate the highest dataset-scale F1-score across various thresholds (referred to as the "Optimal Dataset Scale," or ODS). We thinned each ground-truth whistle to a width of one pixel and compared them to predicted confidence maps that were binarized using 50 evenly distributed thresholds between 0 and 1. All default parameters within the benchmark tool were used in our evaluation.

Evaluation of Whistle Extraction

We used *Silbido* [171] to evaluate the quality of whistle extraction after the graph search was applied to the confidence map. This library calculates recall, the percentage of validated whistle contours that were detected; and precision, the percentage of detections that were correct. Then we calculated the precision, recall, and F1-score on testing files of each species and averaged them among all species. We determined the success or failure of whistle extraction results by examining the set of expected analyst annotations as described in [171]. We checked whether any of the detections overlapped with the analyst-annotated whistle contour in time. If so, we examined whether each overlapping detection matched the analyst's annotation. When the average deviation in frequency between the detected contour and annotation was < 350 Hz and the analyst detections had lengths ≥ 150 ms, with a signal-to-noise ratio ≥ 10 dB over at least 30% of the whistle, we classified the overlapping detections as matched detections. When an annotated whistle did not meet the above criteria (too short or low intensity), we discarded any matching detections, and they did not contribute to the metrics. We classified unmatched detections as false positives.

Table 3.1: Performance of whistle extraction

N[a]	Mean ODS		Mean F1-score		Mean Precision		Mean Recall	
	w/o GAN[b]	w GAN	w/o GAN	w GAN	w/o GAN	w GAN	w/o GAN	w GAN
1000	69.80±2.41[c]	71.33±2.58	79.74±2.94	80.73±1.89	85.01±3.54	76.86±3.99	84.82±3.44	78.32±3.58
2500	75.37±1.50	77.78±0.89	83.04±1.10	84.73±0.90	85.88±1.92	86.38±1.77	81.29±2.26	83.72±1.32
10000	78.64±0.67	79.38±0.38	84.70±1.11	85.21±0.80	87.55±1.52	87.13±1.86	82.67±1.52	83.85±1.13
all	80.85±0.23	81.23±0.10	87.42±0.44	87.88±0.14	89.27±0.20	89.60±0.31	86.04±0.67	86.63±0.36

[a] We denote the number of real positive samples for whistle extraction model and GAN training as N; "all" indicate that the full dataset is used.

[b] w GAN and w/o GAN indicate the performance of whistle extraction model with or without our GAN generated samples, respectively. The whistle extraction model is the same as [109].

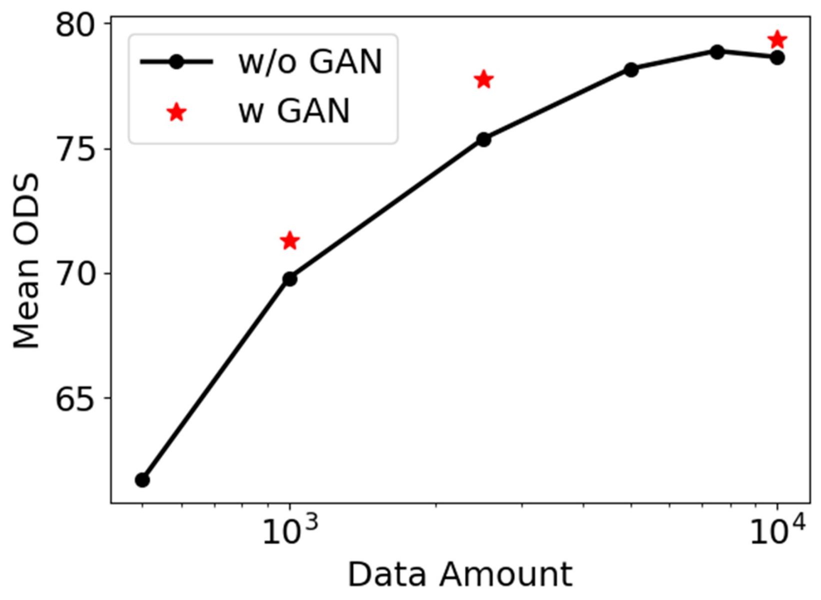
[c] We conduct repeated experiments for each setting, and we report performance average ± standard deviation for each metric. Refer to Section 3.4.3 for more details.

3.5 Experiments and Results

3.5.1 Varied Number of Annotated Samples

We first studied the effect of varying the amount of training data for our whistle extraction network. Because annotation is expensive, a key motivation for data augmentation is to reduce the number of annotations required. Training effective deep-learning models requires a considerable amount of high-quality annotated data [2]. For the whistle extraction task in this work, it remains unclear how the whistle models perform when the amount of annotated data varies. To address this issue, we conducted 6 experiments that selected n positive patches and n negative patches, where n = 500, 1000, 2500, 5000, 7500, or 10000. Random selection of patches was structured to ensure that smaller datasets were subsets of larger ones. We repeated this process five times to obtain 5 datasets for each n. For each dataset, we trained whistle extraction models 5 times, and report average performance.

The experimental results are shown in Fig. 3.6. The black curves show the performance of the confidence map (ODS) and whistle extraction (F1-score) (upper and lower plots, respectively) with respect to the quantity of training data. While the ODS quantifies the performance of the whistle extraction model in detecting the presence and shape of the whistles, the results suggest that, with more training data, the average ODS increases. The increase in whistle extraction F1-score follows the same trend as ODS. Our results show that increasing the amount of annotated data substantially improves the performance of whistle detection. At the same time, as the amount of data increases, the rate of performance improvement decreases, which means that exponentially more data may be needed to increase performance by 1 unit when the initial dataset is larger.



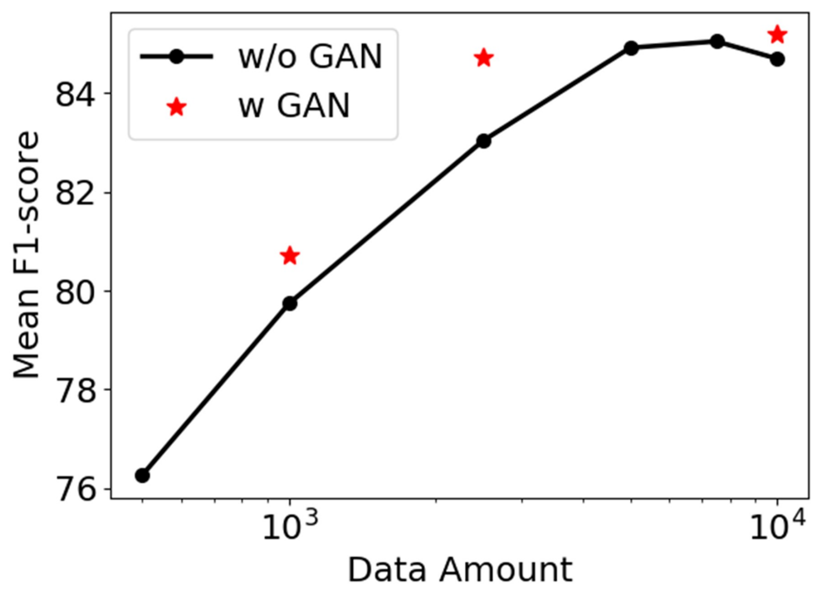


Figure 3.6: Mean spectral peak detection F1-score (upper) or mean whistle extraction F1-score (lower) against the number of real positive samples in the training set. Optimal Dataset Scale (ODS) is an edge detection metric that assesses peak detection. "w/o GAN" and "w GAN" indicates the performance without and with GAN augmentation, respectively.

3.5.2 Data Augmentation

We also studied the effect of varying dataset size on GAN training and data augmentation. In this set of experiments, we applied the proposed augmentation method to augment n = 1000, 2500, and 10000 positive samples and negative samples. In each experiment, we generated 10 $\times$ n samples with our WAS-GAN. All GAN networks were randomly initialized and trained once per dataset. For each augmented dataset, we trained the whistle extraction model with ABN for 5 times.

Fig. 3.5 shows examples of samples generated by our WAS-GAN (n = 2500). By visually comparing the real samples and generated samples, we see that the noise patterns and whistle signal patterns are well simulated by our GAN networks, e.g., the clicks (wide vertical band of high energy across the frequency domain) are simulated well, as are the width and strength of whistle signals.

Table 3.1 reports the experiment's ability to correctly predict time-frequency peaks associated with whistles (mean ODS) and to correctly extract whistles from these predictions (mean F1-score). Consistent performance improvements were obtained for both measures. Our methods obtained gains of 1.53, 2.41, and 0.74 in mean ODS, and 0.99, 1.69, and 0.51 in mean F1-score for the three augmentation experiments when n=1000, 2500, and 10000 training patches, respectively. We also obtained improvements of 0.38 and 0.46 in the mean ODS and mean F1-score, respectively, when we used WAS-GAN on the full dataset. In comparison to experiments using n=10000, we utilized over 100,000 additional annotated samples in our full dataset experiment. These samples were manually labeled as opposed to our GAN augmented samples, and this led to an increase of 2.72 in the whistle extraction F1-score. Without our GAN-generated samples, in order to achieve a 0.46 increase in the F1-score by adding more human-annotated samples to our current dataset, we would have to annotate tens of thousands more samples. The training stability was notably improved (with a reduction in the variance of the F1 metrics) with the addition of the generated data. These improvements highlight the effectiveness of our proposed stage-wise, GAN-based data augmentation method: the use of augmented data improves spectral peak detection results, which in turn also improves whistle contour extraction results.

3.5.3 Ablation Study

We conducted a set of ablation experiments to examine the contributions of different components of the proposed method. We chose datasets with n = 2500 samples for these experiments. The quantitative results are shown in Table 3.3.

Table 3.2: ablation study

Experiments	Mean ODS	Mean F1-score
2500+GAN[a]	77.78	84.73
- residual[b]	-1.43	-1.44
- select	-1.21	-1.44
- ABN	-0.68	-0.98
- ABN, - select	-0.86	-1.97
- residual, - select, - ABN	-2.01	-5.21
vanilla GAN[c]	-0.57	-1.04
Random Addition[d]	-0.36	-0.65
Random Addtion + Gaussian Blur[e]	-0.37	-0.67

[a] GAN augmentation from 2500 real positive samples and 2500 negative samples. We report the whistle extraction performance with our proposed GAN method in this row and the change of performance compared to this row in the following rows.

[b] -XXX means that component XXX is removed. The components include: (i) residual: residual learning; (ii) select: selection of synthetic whistles with entropy and duration criteria; (iii) ABN: auxiliary batch normalization.

[c] We replace stage-wise GANs with a single WGAN-gp [62] for sample synthesis.

[d] We remove the third GAN model (CycleGAN) and directly add the output of the first two GANs with random weights.

[e] We apply random Gaussian blurring to the generated whistle contour before it is added to background noise.

Residual learning

In this ablation experiment, we trained the CycleGAN in stage 3 to directly generate positive samples rather than adding the residual to the negative samples (Eq. 3.3). While we can change the whistle signal magnitude by altering the weight in Eq. 3.3 when the generator outputs residual, the whistle signal's magnitude is determined by the generator model in this setting. In contrast to the proposed WAS-GAN, we observed a decrease of 1.43 in mean ODS and a decrease of 1.44 in mean whistle extraction F1-score when we removed residual learning. This performance drop might be caused by the fact that the GAN needed to output background noise, which might increase the difficulty and instability of learning. Moreover, the variance of generated data decreases when the magnitude of whistle signals cannot be adjusted by the multiplier of the residual.

Patch selection

This ablation experiment removed the quality assurance filter (Eq. 3.15 and Eq. 3.16) for whistles generated by the GAN. As a result, generated whistles similar to those surrounded by the red bounding boxes (Fig. 3.3) were included in the training data. The mean ODS dropped by 1.21 and the mean F1-score decreased by 1.44 after this change. This indicates that low-quality samples may reduce the performance of the whistle extraction network training, and our simple heuristic selection method effectively selects samples for the whistle extraction task.

ABN

Because ABN stores statistics of real samples and generated samples separately, it may better stabilize the training when the generated samples and real samples have different distributions [221]. We evaluated the functionality of applying ABN with and without patch selection to our whistle extraction task; patch selection affects the generated sample distribution. After removal of ABN, the whistle extraction F1-score dropped by 0.98 with patch selection and by 1.97 without patch selection. This suggests that our patch selection method contributes to generating samples that are closer to the actual distribution of whistles. The performance change is consistent with our hypothesis that generated samples and real samples have a different distribution when few data are included in GAN training.

We also observed decreases in ODS of 0.68 with patch selection and 0.86 without patch selection after removal of ABN, which is a less decrease compared to whistle extraction F1-score. While ODS demonstrates the whistle extraction model's performance at the spectrogram bin level, this metric does not always linearly correlate to the whistle extraction performance, because it ignores the signal continuity among bins. We observed that removing ABN frequently resulted in poorer continuity of predicted patches (e.g., Fig. 3.7d and 3.7e, first and third examples) and a greater number of false positives (e.g., Fig. 3.7d and 3.7e, second example). The whistle extraction F1-score also indicates the model's ability to recognize whistle signals under varying noise conditions or suppress false positives in the high-energy region of spectrogram according to the context information (signals in the neighborhood). The generated whistle contour and signals may be less continuous than the real samples, which will train the whistle extraction model to ignore context information and make discontinuous predictions when ABN is removed. The comparison among Fig. 3.7 (c), (d), (e) rightmost column also shows that use of our generated data reduces false positives.

Stage-wise GAN

Instead of decomposing the sample generation into multiple stages, we used a single WGAN-gp with two output channels to generate whistle data, the spectrogram samples and their labels, similar to [12]. To deal with the increased learning difficulty of one WGAN, we

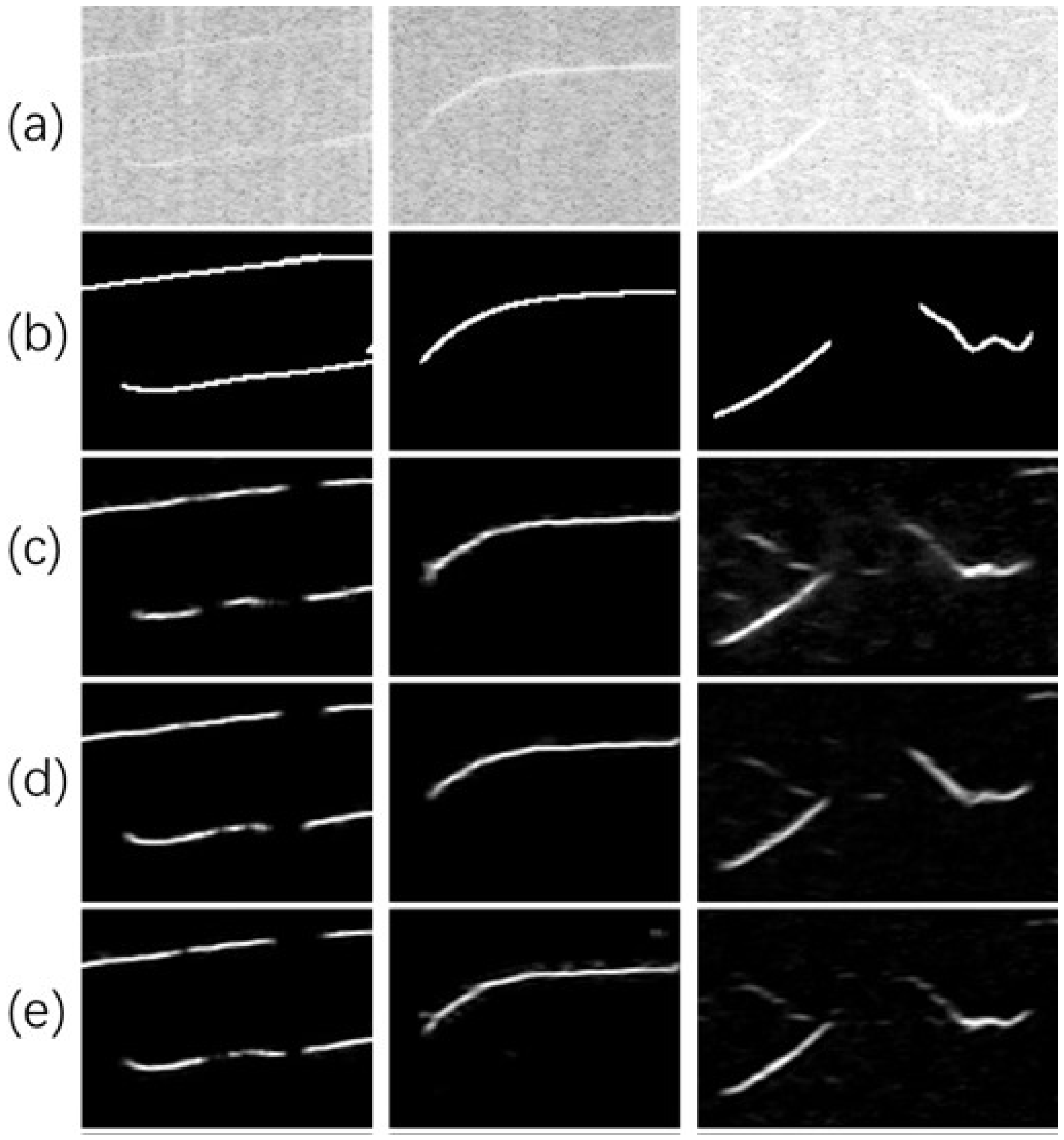


Figure 3.7: Outputs of whistle extraction models. Models with the best whistle extraction F1-score among all parallel experiments in each training setting are visualized. (a) Spectrograms that are used as model input. (b) Ground truth. (c) Output of model trained with 2500 real positive patches and negative patches. (d) Output of model trained with 2500 positive patches and negative patches and GAN synthesized data. (e) Same as d, but the model does not have auxiliary batch normalization (ABN).

increased the WGAN-gp capacity of the generator by using twice the number of hidden layers for each convolutional layer output as that in Section 3.4.2. Examples of samples generated by this model are shown in Fig. 3.8. We saw clear artifacts and unnatural, sudden changes in the magnitude in adjacent bins on the spectrogram. The visual quality of generated samples

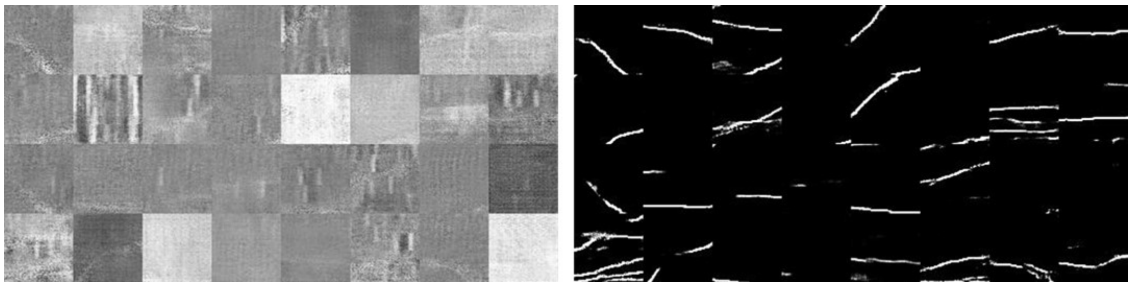


Figure 3.8: Positive samples (left) and corresponding whistle contour (right) generated by vanilla GAN. Multiple 64 $\times$ 64 patches are concatenated.

was substantially worse than those generated by our stage-wise GAN in Fig. 3.5. We also observed a decrease of 1.04 in the whistle extraction F1-score compared to our proposed framework. Data augmentation with the low-quality samples still permitted the performance of the model to surpass that without augmentation for the time-frequency detection task. The negative effect of using corrupted data might be mitigated by the ABN layer.

The third GAN

In this ablation study, we remove the third GAN and instead generate positive sample I'_P by simply adding the generated whistle contour I_W to the generated background noise I_N. Following the work of Li et al. [109], we apply Gaussian blur G with random deviation parameter σ to the whistle contour, and we add the blurred contour to the background noise:

$$I'_P = I_N + \lambda CLIP(I_W + G(Y,\ \sigma)) \tag{3.18}$$

where the clipping function $CLIP(x)$ is

$$CLIP(x) = \begin{cases} 0,\, x \in (-\infty, 0) \\ x,\, x \in [0, 1] \\ 1,\, x \in (1, +\infty) \end{cases} \tag{3.19}$$

We also try a simple version which does not contain Gaussian blur:

$$I'_P = I_N + \lambda I_W \tag{3.20}$$

where λ is a random weighting parameter. We use the same parameter setting as Li et al. [109], where λ and σ are uniform random numbers within the ranges of [0.03, 0.23] and [0.3, 1.3], respectively. As shown in Table 3.3, both methods in Equation 3.19 and Equation 3.20 lead to inferior performance compared to the proposed stage-wise GAN method ("2500+GAN") that uses the third GAN. Considering that we use the same set of background noise and whistle contour shapes, this ablation study indicates that our proposed stage-wise GAN method generates more realistic whistle signals with a appearance which contributes to the improved training of the whistle model.

Table 3.3: Comparison of whistle extraction methods

Method	F1-score	Precision	Recall
Roch et al., 2011 [171]	75.95	81.125	72.275
Gruden et al., 2020 [60]	83.40	76.55	92.45
Gruden et al., 2020 (≥150ms)	74.38	95.85	60.875
Li et al., 2020 [109]	87.42	89.27	86.04
Li et al., 2020 + our GAN	87.88	89.60	86.63

3.5.4 Comparison with Other Whistle Extraction Methods

In addition to our previous work on network-based whistle extraction [109], we have selected two representative and competitive whistle extraction methods for comparison. Both methods identify whistle candidate points by determining if the Signal-to-Noise Ratio (SNR) values are above a threshold on the denoised spectrogram. The Graph-Search method developed by Roch et al. [171] employs graphs consisting of candidate points, which are extended with new points based on how well these new candidate points align with the existing graph through polynomial fitting results. As a point of comparison, Gruden et al. [60] uses a probabilistic approach based on the sequential Monte-Carlo probability hypothesis density (SMCPHD). In addition to the result of all SMCPHD predictions, we also present the results of predictions that are longer than 150ms, as both Graph-Search and our method apply this length criterion for detection.

Our approach outperforms SMCPHD and Graph-Search in the whistle extraction F1-score by 4.48 and 11.93, respectively. Additionally, our GAN-generated samples improve the method in [109] by 0.46 in F1-score, 0.33 in precision, and 0.59 in recall. SMCPHD demonstrates the highest recall but the lowest precision in this comparison, which indicates its aggressive strategy of making more whistle predictions. By removing whistle detections by SMCPHD that are shorter than 150ms, the precision of SMCPHD is improved by 19.3, while the recall is decreased by 31.57. This study suggests that SMCPHD prefers shorter segments of whistles in its predictions. Our GAN-generated samples help the learning-based model achieve a competitive performance advantage on this whistle extraction task, however, it should be noted that optimizing the other algorithms for this specific dataset may diminish these advantages.

3.6 Conclusion and discussion

We present a framework of stage-wise generative adversarial networks to generate training samples for whistle extraction. The data generation process consists of three stages: (i) generate time x frequency spectrogram patches containing background noise (ii) generate whistle contours and automatically discard poor quality contours (iii) fuse whistle signals with the background noise. Each stage is completed by one trained generative adversarial network. Compared to using a single vanilla GAN generating whistle extraction data and labels, our stage-wise GANs can generate samples with fewer artifacts which results in increased whistle extraction performance. We examined our data generation method by a series of experiments employing differing quantities of real and generated data, and note that using the generated data lead to consistent performance gains.

The stage-wise design may mainly contribute to the success of our data generation method. It separates the modeling of different components and the relationship between components, which eases the learning of the GANs in each stage as well as provides a straightforward way to explore different combinations of components. In our case, we generated the background noise separately and we were able to add different whistle signals to the same background. If we directly apply this idea to semantic segmentation data generation of natural images, we may first generate the appearance of background scene, then generate objects on it according to a desired segmentation map. If we extend this idea, we may generate the appearance of different objects separately and then add them to the background. In this way, we may fully explore combinations of varying objects and background appearances in the same segmentation layout. In our whistle extraction experiments, we did not use this extended idea, because the appearance of our foreground object (the whistles) is relatively simple, i.e., the variance of appearance is mainly rooted in the whistle contour shape and whistle magnitude. Therefore, we directly add whistle signals to the background using the third GAN in our framework. Our framework can be readily extended to extract calls of other whale species (e.g, baleen whales) and to other similar tasks (e.g., semantic image segmentation).

Though it may not affect the main contributions of this work, our data generation method can be improved in three aspects in the future. Firstly, we may use improved generative neural network architecture and training strategies. For example, we may use a generator architecture based on a style-transfer network which improves the generated sample quality [94]. The discriminator augmentation mechanism proposed in [93] may help stabilize training in limited data regimes. We may also explore generating larger patches of high quality with the method in [92]. Secondly, we may use real data in the data generation process to enrich the data variance. The real background and annotated whistle contours can be used as the input data of our GAN in the third stage, and we can generate whistle signsals of novel shapes on real background or generate whistle signals of annotated contour shapes on GAN-generated background. Thirdly, we may improve the sample selection method. In this work, we use a simple yet effective pixel-wise entropy method to select whistle contour of good quality. Metric measuring texture or semantic information like [101] may better measure the quality of our generated samples and improve the sample selection process.

Chapter 4

Learning Data Augmentation Policy for Image Classification

4.1 Introduction

Image augmentation has been proved to be an effective technique for boosting supervised models in a wide variety of multimedia applications, including image classification [100, 71, 75], video classification [91], image labeling [130], image segmentation [172], object detection [119], etc. A typical augmentation method is to apply image transformations, e.g., translation, over training images and use the transformed image together with the original image labels to train image models. Advanced image transformations include Cutout [41], Cutmix [233], and GAN-based methods [3, 182]. Image augmentation can bring data variances to the existing training set, mitigate over-fitting, and eventually improve model generalization capabilities without extra human annotation efforts. However, the enlarged training set may lead to high computational cost, depending on the number of augmented images added, which limits the use of this approach in large-scale multimedia applications. In this work, we propose an effective augmentation approach that can outperform existing methods while using fewer augmented images.

In the past literature, a classical augmentation approach is to manually apply multiple image transformations to each training image to obtain an enlarged training dataset. Researchers recently proposed to search for the optimal augmentation policy for a given training set. Cubuk et al. [34], for example, define a policy as a set of sub-policies, and each sub-policy includes two consecutive image transformations. A candidate policy's performance is evaluated by how this policy can boost a proxy task, e.g., classification of a hold-out dataset [115]. The optimal policy is obtained through searching the feasible hyper-parameter space, including image transformation types, operation magnitudes, and the possibility of transformations. This search-based augmentation method has been successfully applied in multiple image relevant tasks, including image classification [100, 71, 75], and object detection [119, 55]. Studies over multiple datasets also suggested that one dataset's optimal policies might still be effective for a separate dataset.

The above search-based methods aim to find the optimal augmentation policy for the whole dataset and then apply it to each training image. While the policy works well for a majority of the training images, it is expected that the optimal augmentation policies for individual images might be different. Moreover, some transformations (e.g., shifting 10 pixels to the right) might lead to invalid training sample-label pairs. For example, in Fig.4.1, the image of an elephant remains recognizable when translated to the left. In contrast, the image of fish might lose critical information (the fish's head) if the same transformation is applied. The optimal transformation or transformation combinations are sample-specific and should be optimized separately and independently.

There are also works exploring augmentation strategies based on image content [48, 162]. Fawzi et al. [48] search perspective transformations maximizing the classification loss, but the algorithm operates on perspective transformation matrix, and it may not easily extend

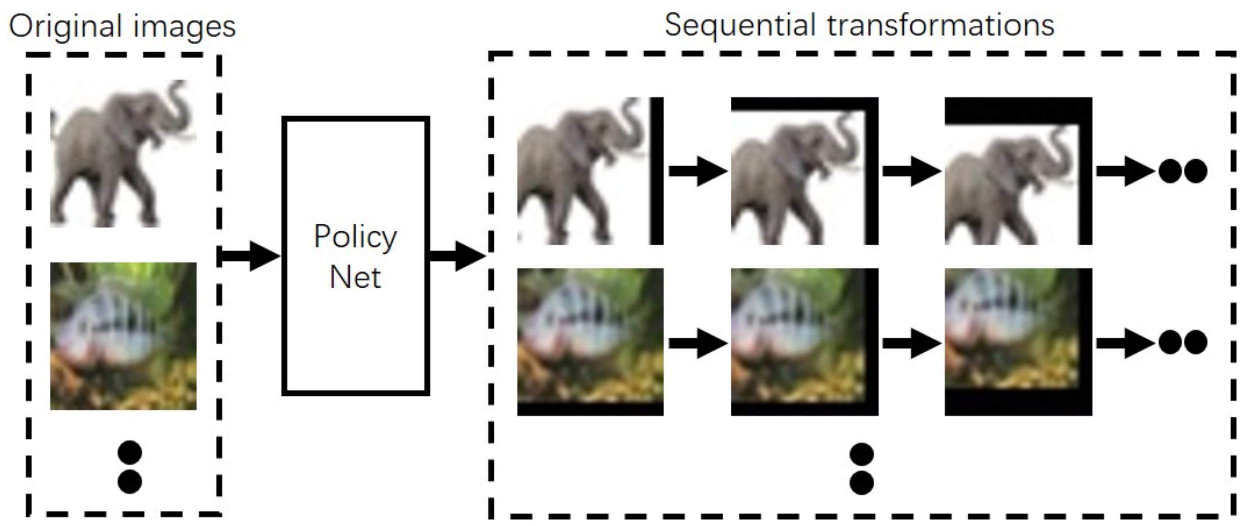


Figure 4.1: Policy-based sequential image augmentation. Left: original images; Right: transformed images by a trained policy net.

to other types of transformations. Ratner et al. [162] use generative adversarial training to generate a sequence of transformations for each sample. However, the length of the sequence is fixed, making it impossible to apply different numbers of transformations to samples. Besides, the objective of generative adversarial training encourages the transformed image follow the distribution of original data, which excludes severe transformations and difficult samples for the classifier.

In this work, we develop a learning-based sequential approach to fully exploit the potential of data augmentation. Our approach aims to seek a sequence of image transformations for a training image. The optimal transformation at each step of the sequence, including both types and magnitudes, is determined by the visual content of the input image, and different images might end up with different image transformations. This sequential image augmentation is thus *sample-specific*, in contrast to the dataset-wide augmentation strategies. For a given image, the search space for finding the optimal sequence is of high complexity due to the combination nature. For the same reason, Cubuk et al. [34] employ only two consecutive transformations in each sub-policy. To address this challenge, we introduce a deep policy network to map the current input image to its optimal transformation or action.

The transformation at a time step is selected to maximize both immediate reward and accumulated future reward. In this way, we cast the sequential image transformation task as a sequential decision process and train the policy network in the reinforcement settings.

We apply the proposed image augmentation approach for the image classification task to demonstrate its effectiveness. A joint scheme is introduced to alternatively train a classifier and the deep policy network. At each iteration, for each training image, the immediate reward of a candidate transformation is defined to encourage the generation of more difficult samples w.r.t. the current classifier. A stop condition is also introduced to prevent invalid augmented images. We empirically find that, with the proposed reward and stop condition, the policy network tends to select a few transformations from the candidate pool, and the join training scheme could largely diversify the augmented dataset.

We implement the proposed policy-driven augmentation approach for two settings: using only one type of image transformation, i.e., translation, or using a group of image transformation. Two public datasets for image classification and a newly collected dataset for material recognition are used for evaluation purposes. Experiments with comparisons to alternative augmentation methods suggested that (i) our approach with only one transformation type is more effective than other methods using a single transformation (e.g., Cutout [41]). Figure 4.1 showed a sequence of translated images where the translation magnitudes are determined by the policy network. (ii) While using multiple transformations, our approach achieved the state-of-the-art performance while using a much less number of augmented samples. The proposed approach can be easily extended to other image or video tasks.

The three contributions of this work are: (i) a novel policy-driven approach for sequential image augmentation; (ii) an iterative approach for jointly training the policy net and classifiers; (ii) improved performance and augmentation efficiency over existing augmentation approaches on both public image datasets and a newly collected image dataset.

4.2 Relationships to Previous Works

This work is closely relevant to four streams of research in the areas of visual content analysis and machine learning.

Manual Image Augmentation As aforementioned, a classical image augmentation approach is to apply one or multiple image transformations over each training image. Common transformations [71, 67, 75] include translation, reflection, scaling, cropping, and color adjustment. Each transformation also comes with different hyper-parameters, e.g., magnitudes for translation. These transformations have achieved encouraging improvement for multiple image relevant tasks, including image classification [100, 29, 173, 71, 67, 75] and object detection [119, 55]. In the recent literature, there are also multiple novel transformations proposed for various image tasks. Elastic distortion is used on digit recognition dataset MNIST [185, 217]. Patches could be randomly selected and replaced with constant value [41], random noise[241], or mixed patches [233]. In image classification tasks, images from the same class could be mixed by assigned weights [236, 197, 77]. In object detection or instance segmentation tasks, objects and their labels could be cut and pasted to other images [45, 52]. Furthermore, there are studies applying augmentation in image feature space [40, 151, 209, 211]. The above works often manually select a set of transformations empirically and apply them over all training images. The transformation parameters (e.g., magnitudes in translation) are usually fixed for all images. This manual augmentation policy might not be the optimal one in terms of boosting system performance. It is possible to employ as many transformations over a single dataset, which, however, might suffer from the exponentially grown training complexity.

Search-based Augmentation In the recent literature, there are multiple search-based augmentation approaches proposed to boost both system performance and augmentation efficiency. They aim to search for the optimal augmentation strategy for a given dataset in order to avoid manual or random augmentation strategies. In the pioneer work [34], Cubuk et al. employ a hold-out set to assist in the search for optimal transformation pairs where each pair comprises two consecutive transformations and their parameters (e.g., magnitudes). Zoph el al. [246] implement this methods to object detection task. Following AutoAug, new methodologies are proposed to reduce the policy searching complexity. Lim et al. [115] and Hataya et al. [69] remove classifier training from policy searching and directly use the performance of the classifier on transformed validation set as a reward. Ho et al. [74] and Lin et al. [116] jointly train classifier with policy searching, which leads to the non-fixed policy during classifier training. Recently, Cubuk et al. [35] propose to reduce the searching space and directly find the optimal classifier by grid search.

The above approaches could discover the optimal transformations or transformation combinations which are effective dataset-wide. These transformations might be suboptimal for individual training images. Transformations with fixed magnitudes (e.g., translation) might even lead to invalid training samples.

The proposed research is aligned with the so-called sample-specific augmentation methods [48, 162], which aim to find the optimal transformation or transformation sequence for each image. For example, Ratner et al. [162] employ the generative adversarial training framework to generate a fixed-length sequence of transformation for each image. The discriminator predicts if the transformed image becomes an outlier in the natural image distribution. Then the generator could predict a transformation sequence that does not change the information regarding to image classification. However, this method forces a fixed length of transformation sequence and may encourage generating easily recognized samples. Fawzi et al. [48] search the augmentation strategies for each sample in a restricted perspective transformation space. While they select transformation within a trust region to avoid data corruption, the transformation which causes the highest classification loss for the image is the best for augmentation. But this method might be limited to perspective transformation.

GAN based Augmentation With the mature of GAN techniques, there are multiple efforts integrating GAN models to augment the training dataset. Previous works [198, 3, 182, 186, 107] tried to generate new images from estimated distributions, which are used to enlarge the training dataset. Generative Adversarial Networks could learn the image distribution and sample data from random latent variables [3], synthetic images [182, 186], and natural images [107]. Tran et al. [198] model the data distribution by a Bayesian approach and learns a generator network by a generalized Monte Carlo EM algorithm. Those sampling-based methods are affected by the quality of generated images, and there might be artifacts affecting classifier feature learning. In particular, similar to adversarial samples [221], we empirically find that GAN-generated images and natural images bear different underlying distribution, and the mixture use of both would, in general, lead to depressed classifier performance.

Reinforcement learning The proposed method is motivated by the success of deep reinforcement learning in multiple areas, including robotic control [114, 61], game agents [143, 183, 184], network architecture search [8, 247], image restoration [231], face hallucination [17], object tracking [232], image captioning [167]. In this work, we formulate the search for sequential image transformation as a sequential decision process and employ the deep policy network to parameterize the mapping between input images and the optimal transformation (and its hyper-parameters). In this way, the agent is guided by the policy network to select a transformation based on the visual content of the input image. The agent is trained in the reinforcement settings to maximize both immediate rewards and cumulative future rewards. The proposed method is the first of its kind in learning image-specific policies for sequential image augmentation.

4.3 Our Approach

The objective of this work is to develop a policy-based augmentation approach for classification purposes. Our approach aims to jointly learn a classifier and a policy network in a reinforcement learning setting. The developed techniques can be potentially used for other image tasks, e.g., object detection, video classification, etc.

Fig.4.2 sketches the proposed augmentation approach which includes three major stages. In stage 1, a deep policy net is trained to produce the optimal transformation for each training image or its transformed version. A classifier is used to define the immediate reward of a candidate transformation, as introduced later, and thus regularize the training of the policy net. In stage 2, the policy net is applied over each training image to transform training images and generate an augmented dataset. In stage 3, both the original images and transformed images are used to update the classifier. We alternatively perform the above three stages to jointly train the deep policy model and classifier.

4.3.1 Formula: Sequential Image Augmentation

Consider image classification tasks where a set of training images are annotated for training the classifier. Traditional image augmentation methods either manually select a set of image transformations or employ various optimization methods to search for the optimal transformations or transformation combinations. In this work, we aim to apply a sequence of basic image transformations over each training image, including flipping, translating, rotating, blurring, and others as reviewed in Section 4.2. By denoting the sequence of transformation functions as f_1, f_2, ..., f_t, the training image I_0 could be transformed to I_t at step t as follows:

$$I_t = f_t \circ f_{t-1} \ldots \circ I_0. \qquad (4.1)$$

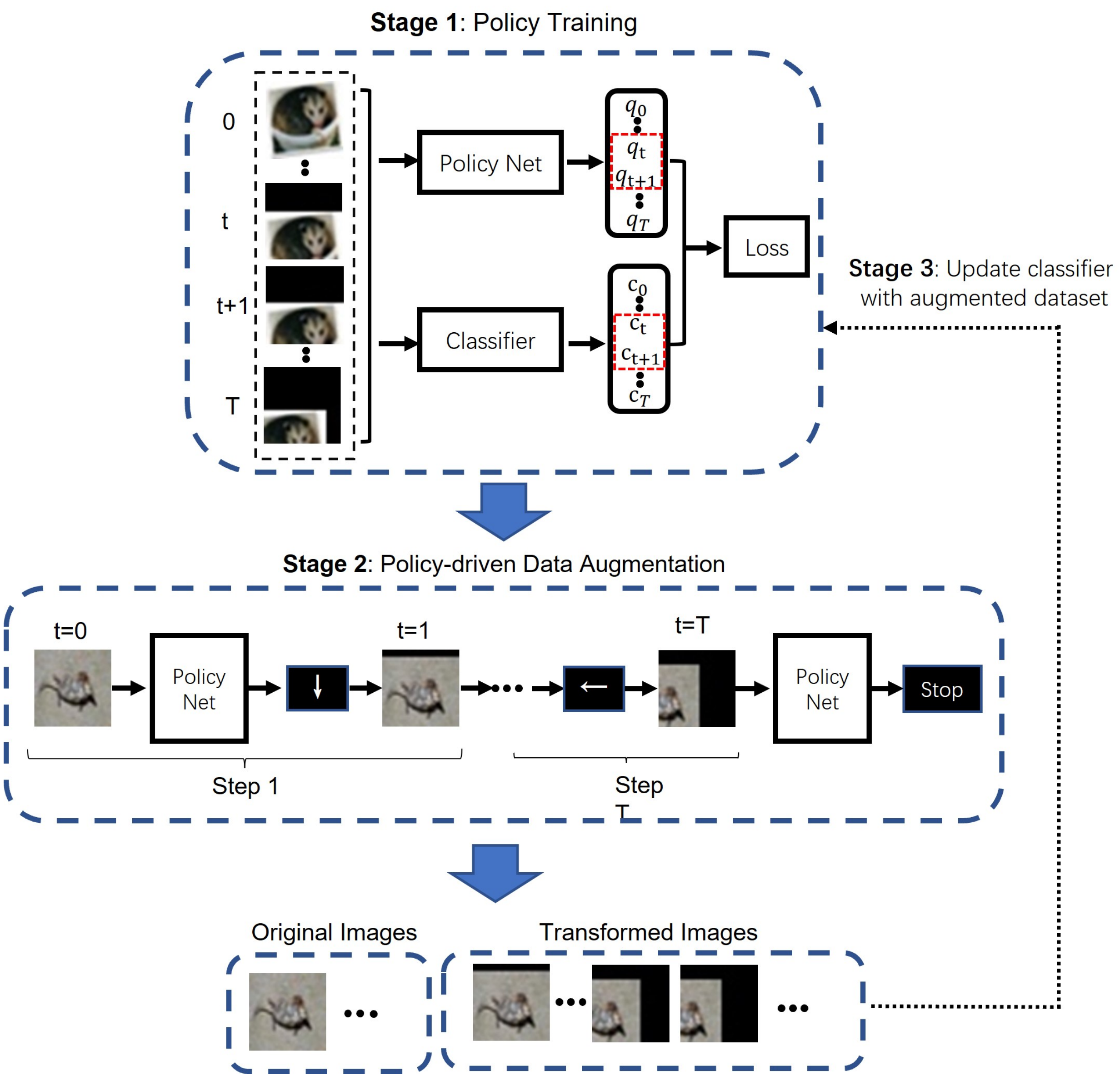


Figure 4.2: Sketch of the proposed policy-based data augmentation approach. There are three major stages, which are alternatively performed over iterations. a_t, q_t, r_t represents the actions, Q-value and reward respectively at step t. See texts for more details.

where t is the length of the sequence and $\circ$ denotes the operator of transformation. Each transformation f_i is specified by its transformation type and magnitude. The transformed image at a step of the sequence is used as the input image to the next transformation.

The sequential image augmentation is characterized by two folds compared to other augmentation strategies. Firstly, it embraces the idea of composition. For example, a translation by 15 pixels is equivalent to applying three translations by 5 pixels in a row. This composition principle can largely reduce the feasible space while searching for transformation magnitudes. Secondly, a sequence of augmentation transformation might involve multiple types of transformations and thus enhance the diversity of the augmented dataset.

The proposed sequential image augmentation approach, however, brings two major challenges. Firstly, the search for the optimal sequence of transformation is of much higher complexity than finding the optimal basic transformations, mainly due to the combination nature. In particular, different image transformations at a step of the sequence will lead to different transformed images for which the optimal transformation would be different accordingly. This means that the selection of the optimal transformation at each step should be jointly solved with the selection problems for the following steps. Secondly, the sequential augmentation will have to incorporate stop conditions to prevent the generation of invalid transformed images. Taking the image of an elephant in Figure 4.1 for instance, the three transformed images are still valid after applying multiple translations. It will, however, become non-recognizable if only the top-left corner is kept. Therefore, it is critical to terminate the sequence of image transformation once the transformed images are not semantically consistent with the class label. In the following subsection, we will explain a deep policy-based optimization method to address the above challenges.

4.3.2 Policy-based Sequential Image Augmentation

We cast the search of the optimal transformation sequence to be a sequential decision process where an agent acts to select an appropriate transformation for an input image. Each action is drawn from an action set A including all transformation types (e.g., translation or rotation) and the relevant magnitudes (e.g., displacement pixels or angels). At time step t, the agent takes state S_t as input, selects action a_t which have the largest Q value as

$$a_t = \text{argmax}_{a \in A} Q(a, S_t) \tag{4.2}$$

and transforms the image into I_{t+1}. The sequence of decisions made by the agent will lead to a set of augmented images, and we denote I_0 as an original image. This process could be illustrated as:

$$S_0 \xrightarrow{a_0} S_1 \xrightarrow{a_1} \cdots \xrightarrow{a_{T-1}} S_T$$

A policy network is trained in the reinforcement settings to guide the agent's decision, which is either an image transformation or an early stopping action.

Figure 4.2 demonstrates how the agent acts to transform a training image. For a training image I_0, the agent first employs the policy net to select an action. Then, we apply the action over I_0 to obtain I_1, i.e., the transformed image. The agent will apply the policy net over I_1 again and repeat the above steps to obtain the sequence of transformed images: $I_2, I_3, \ldots$. In the rest of this subsection, we will provide details of the agent in terms of actions, states, rewards, stop conditions, and objective functions.

Action The action space A consists of all possible image transformations that the agent could use to augment individual image-label pairs. We also introduce a special action, i.e., stop action, which, once selected, will terminate the sequence of transformation. An image transformation might be associated with two attributes: transformation type and magnitude. Table 4.1 lists the types of image transformations used in this work and their magnitude definitions. Note that both image transformations and early stops will be determined by the policy net according to the visual content of the input image.

State The state S_t of the environment represents the information available to the agent at time t. In the proposed method, S_t is defined by $S_t = \{I_t, a_{t-1}\}$, where I_t is the image at t, and a_{t-1} is the previous action before I_t. a_{t-1} will be encoded to be a one-hot vector

Action	Magnitudes	Explanation
ShearX	i, ii	(i) shear left 6%; (ii) shear right 6%
ShearY	i, ii	(i) shear up 6%; (ii) shear down 6%
TranslateX	i, ii	(i) move left 9%; (ii) move right 9%
TranslateY	i, ii	(i) move up 9%; (ii) move down 9%
Rotate	i, ii	rotate (i) counter-clockwise or (ii) clockwise 6 degree
Contrast	i, ii	contrast factor (i) 0.9; (ii) 1.1
Color	i, ii	enhance color factor (i) 0.9; (ii) 1.1
Brightness	i, ii	brightness factor (i) 0.9; (ii) 1.1
Sharpness	i, ii	sharpness factor (i) 0.9; (ii) 1.1
AutoContrast	n/a	-
Invert	n/a	-
Equalize	n/a	-
Solarize	n/a	-
Posterize	i, ii, iii	reduce color bits to (i) 5; (ii) 6; (iii) 7

Table 4.1: Transformation types and their magnitudes used in this work. Each magnitude defines a possible action for the agent.

without stopping action, and a_{-1} is a zero vector. The state could provide information on the image content and historical actions, and will be used for the agent to make decisions. For example, an agent may not choose to translate the image to the left if the main object is near the left boundary. The agent may not translate the image to the right if it just moved the image to the left in the previous step.

Reward For a training image, the reward of an action is defined to encourage the generation of difficult images that challenge the current classifier. We first train a classifier using the original training data. Then, we apply each candidate transformation over the input image and feed both the original image and transformed image to the classifier. Last, the reward of this transformation is defined as the difference between two images' classifier confidences w.r.t. the true label. Let $c_{I,k}$ denote the estimated confidence of the image I belonging to the class k. The reward of a transformation f is:

$$R = c_{I,k} - c_{f \circ I,k} \tag{4.3}$$

The reward would be positive if the transformed image becomes more difficult to the classifier, i.e., receiving a lower confidence level; negative if otherwise. Adding difficult samples to the training set has been approved to be an effective way to avoid over-fitting and improve classifier generalization capabilities [48].

Stop Condition The agent will choose to terminate the sequential transformation once the transformed image receives a wrong class prediction from the classifier, i.e., apply a stop action. This intuitive action is used to ensure that the transformation does not substantially change the visual content of the input image and the transformed images are still recognizable for the original image labels. When the stop condition is met, the reward is defined as:

$$R_{stop} = c_{f \circ I,k} - c_{I,k} \tag{4.4}$$

In contrast to Eq. (4.3), the reward is positive when the transformed image receives larger confidence than the input image. The benefits of additionally using this stop condition are two-fold. Firstly, without this stop condition, the agent will be purely driven by the reward function Eq. (4.3) and seek for difficult images and eventually lead to invalid image-label pairs. The stop condition will constrain the transformations within a "trust region" where the augmented samples are visually different from the original image but are still recognizable. Secondly, the stop condition is able to prevent the augmentation of outlier samples. For example, if an image receives a wrong label from the classifier, it will not be augmented since it leads to a stop action immediately.

Objective function We use a deep policy network to approximate the Q function and employ the concept of double Q-learning [201] to stabilize the training of the agent. In particular, during training, we maintain two policy networks: one is the online agent network, and the other one is the target network. The target network is the same as the online network, except that its parameters are copied from the online network every τ steps. The target network kept fixed for all other steps. The agent is trained to minimize the difference between target Q values (fixed network) and current estimate of Q values (online network). Like [143], we employ a temporal difference loss function. Let r_t denote the immediate reward after action a_t, $q(S_t, a_t)$ denote the estimated online Q values for the current state, $q'(S_t, a_t)$ is the target Q value by the fixed target network [201]. The loss function is defined as:

$$Loss = (y_t - q(S_t, a_t))^2 \tag{4.5}$$

where

$$y_t = \begin{cases} r_t + \gamma * max_{a'} q'(S_{t+1}, a') & 1 \leq t < T \\ r_T & t = T \end{cases} \tag{4.6}$$

γ is the discount factor, and T is the episode length. This loss function encourages the agent to have correct accumulative reward estimation. During training, the samples S_t, S_{t+1}, r_t are retrieved from replay memory and are used to calculate this objective function. In each training iteration, the gradients are propagated through the online network and used for agent weights update, with the target network fixed. The memory replay and double network learning can dramatically reduce the variance of deep Q learning.

4.3.3 Joint Training

We develop an alternative training scheme to train the classifier and policy network jointly. The scheme starts with training the classifier over the original training dataset. Then, there are three iterative steps. The first step is to train the agent with the rewards generated from the current classifiers. The second step is to employ the agent to augment the training dataset and we retrain the classifier over the augmented dataset at the last step. We repeat these three steps for multiple iterations until the classifier performance is saturated. Note that we collect all the transformed images during the iterative training to form the final augmented dataset. Our empirical studies suggest that the agent tends to select a few transformation types during a sequence of transformations, and the use of augmented data from multiple iterations will diversify the training samples for the final classifier.

4.4 Experiments

We test and evaluate the proposed policy-based augmentation approach for image classification tasks.

Datasets We employ three image datasets. The first one is a newly collected image dataset for garbage classification. It includes 6234 images from 8 categories (5 recyclable categories: Cardboard, Glass, Metal, Paper, Plastic; 3 non-recyclable categories: Wet Trash, Food, Bottle). We split the dataset into three-fold: a training set (2774 images), a validation set (310 images), a testing set (3150 images). The training set is further divided into two parts: 1664 images for classifier training and 1110 images for policy network training. All images are resized to have a longer side of 500 pixels. As shown in Figure. 4.3, the testing images were collected in different settings from the training images (lighting conditions, camera angles, etc.) from the training images, which makes this dataset a challenging task for state-of-the-art classification models.

The other two datasets are the public image datasets: CIFAR-10, CIFAR-100 [99]. We use the standard training/testing splits. For each dataset, we further divide the training set into two parts: 60% for classifier training and 40% for policy network training. The classifier and agent are iteratively trained, as stated in the last section. Then, the agent acts to augment each training image and the augmented dataset will be used for classifier training in the next

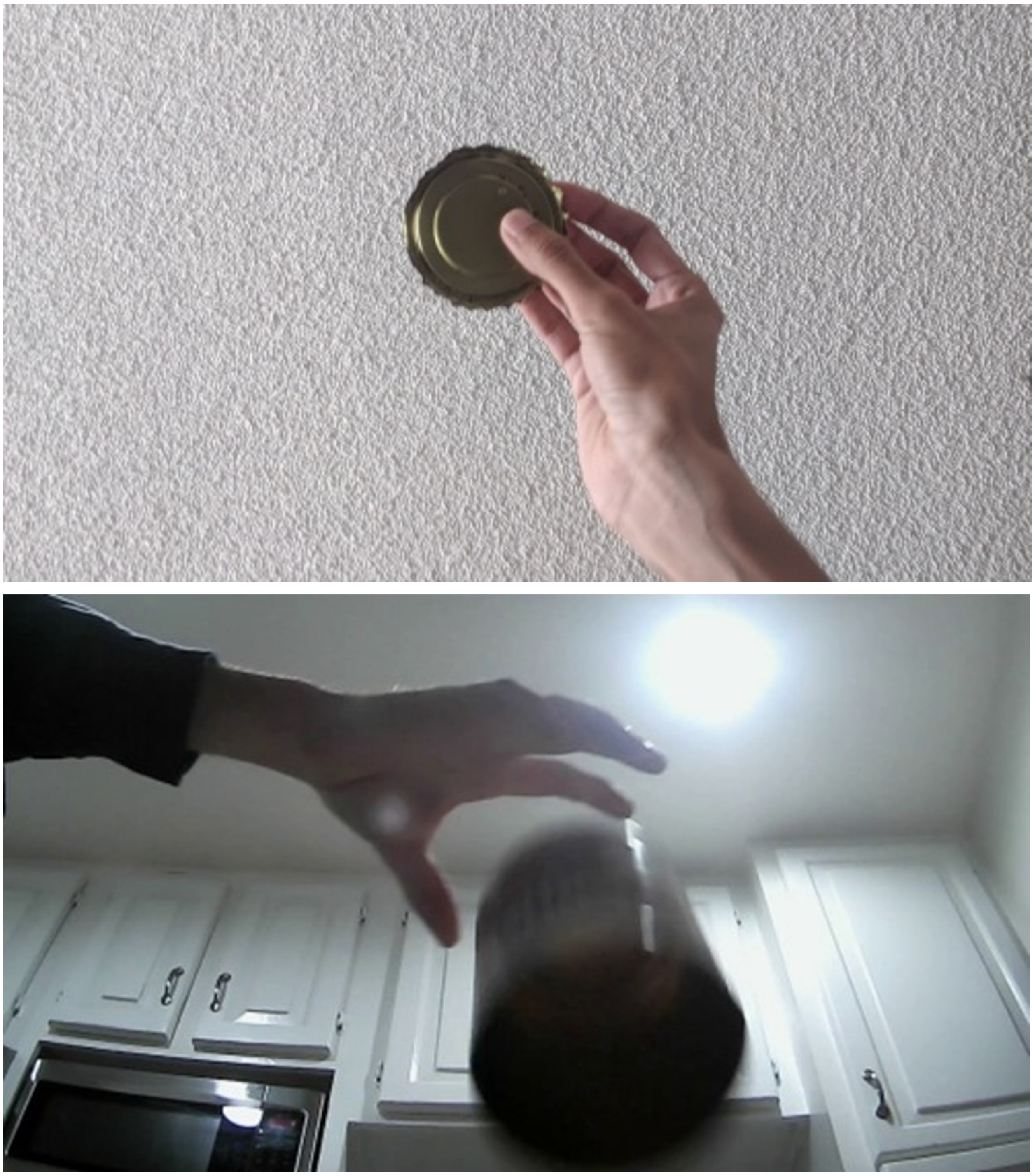


Figure 4.3: Sample images from the Garbage dataset. Two images of metal from the training (left) and testing set (right) have different appearance due to varying lighting conditions.

iteration.

Classification Networks We implement and train multiple network backbones to obtain the classifier including Wide-ResNet-28-10, Wide-ResNet-40-2, Wide-ResNet-28-2, ResNet-50, and Shake-Shake(26 2x96d) [34]. The hyperparameters of these networks are set to be the same as the previous work [34]. In particular, the variants of Wide-ResNet [234] are trained with a SGD optimizer (batch size = 128, initial learning rate = 0.1, momentum = 0.9, weight decay rate = 5e-4) for 200 epochs. The learning rate is scheduled by a cosine learning decay with an annealing cycle, and the same scheduling is applied to other networks training. The Shake-Shake net is trained with an initial learning rate of 0.01, weight decay rate of 0.001 for 1800 epochs. We also follow the same normalization for input images on the CIFAR dataset as [34](mean=[0.491, 0.482, 0.447], standard deviation=[0.247, 0.243, 0.262]). The ResNet-50 is trained with a SGD optimizer (batch size = 64, initial learning rate = 0.1, momentum = 0.9, weight decay rate = 5e-4) for 270 epochs. We preprocess the images as [115] by cropping a center patch of 224-by-224 pixels and apply normalization(mean=[0.485, 0.456, 0.406], standard deviation=[0.229, 0.224, 0.225]) to the RGB channels. Other than the classifier trained on 60% of training samples, which is used for policy network reward calculation, we also train a classifier on all training samples along with the augmented samples, whose performance is reported in Section 4.5.

Policy Network Figure 4.2 illustrates the inputs and outputs of the policy network. We employ the replay memory method and double Q-learning [201] to train the agent. At each step t, the agent takes the image and previous action as input and outputs a vector for expected Q values on all possible actions. The network architecture is the same as [231]. The input image is first encoded to a 32-dimensional feature vector by the feature extractor module(four convolutional layers and one fully connected layer). Then, a one-hot vector representing the previous action is concatenated to the image feature vector. The concatenated vector is processed by the Long Short-Term Memory module, which stores the historical transformation information. Finally, another fully connected layer outputs the Q value vector. The action with the largest Q value is executed during testing.

To study the transferibility of our policy, we train the policy network with one classifier and then apply the same augmentation policy to other classifiers. In our experiments, the policy network is trained with ResNet-50 on Garbage dataset and Wide-ResNet-28-10 classifier on CIFAR datasets. During training, we have one image from our policy training set, and apply ϵ-Greedy action selection until a stop action or the maximum step. The ϵ starts with 1 for 5e3 steps then linearly decreased to 0.1 before 1e6 steps. We update the policy network every 4 steps with an Adam optimizer (batch size = 64, initial learning rate = 1e-4) for 2e6 steps. The learning rate is decayed exponentially to 2.5e-5 during training. The target network is updated with the weights from the latest agent network every $\tau = 1e4$ steps. This sequence of data (state, action, reward) for one image is considered one episode and added to the replay memory. We set the maximum number of steps of one episode to be 10 empirically. 64 episodes are randomly sampled from the replay memory for gradient calculation in each training iteration. The replay memory size is 500,000 for CIFAR10 and CIFAR100 and 15,000 for Garbage respectively.

4.5 Results

Study on Toy Data We first investigate how the proposed stop action and reward affect the augmentation using a toy data. This toy dataset includes a set of 40 two-dimensional data points. These 20 positive and 20 negative points are generated from two Gaussian distributions (illustrated by the oval whose radius is the sigma of Gaussian at radius direction) respectively. We train a logistic regression model as the classifier. To augment a data point, we randomly draw multiple nearby data points and select the one that receives the largest reward as defined in Eq. (4.3). This augmentation strategy will choose the most difficult data points from the neighborhood of a given data point. We repeat the above step until the newly augmented point receives a wrong prediction from the classifier. Figure 4.4 plots the scatter of the data points and the sequence of transformations for a negative and a positive data point (square points with arrows between them). The augmented data points in the sequence are getting closer to the decision boundary. In this way, the reward and the stop action jointly function to generate valid yet difficult samples.

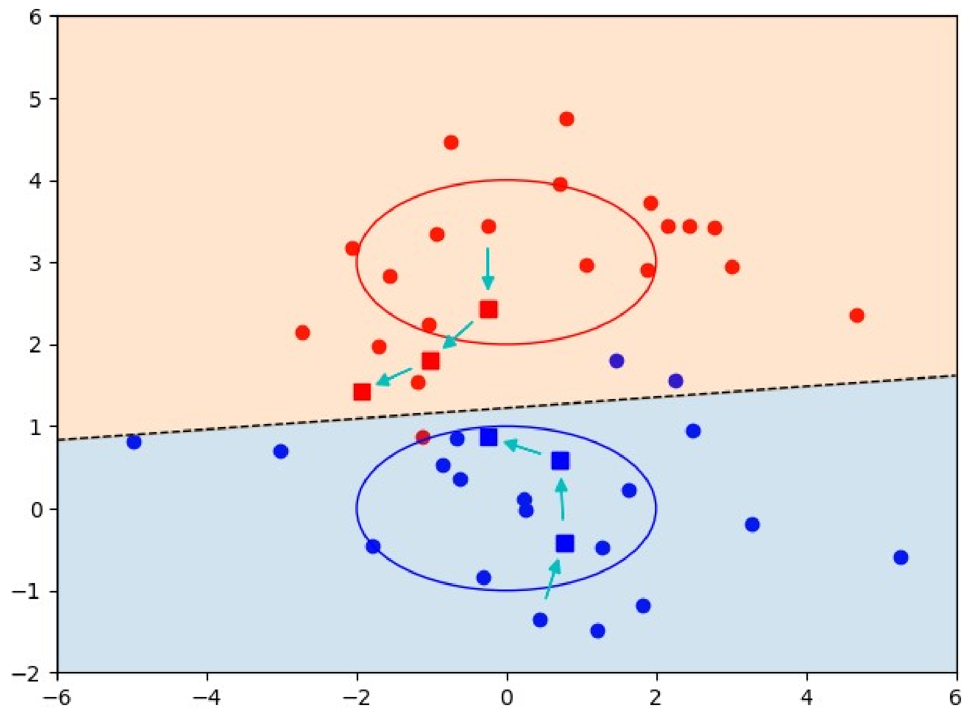


Figure 4.4: Empirical study over a toy dataset. The reward and stop actions would lead to difficult samples (being closer to the decision boundary). See Section 4.5 for more details.

Results on the Garbage Dataset Table 4.2 reports the classification accuracies of different approaches on the garbage dataset. We train three classifier networks with the original training set (baseline) or augmented dataset, and compare the result with Cutout [41]. For fair comparisons, we only use image translation as possible actions for agent. As shown in table 4.1, it involves two transformations (TransX and TransY) with two magnitude or four atomic actions. The settings for Cutout are the same as the original paper [41]. Results showed that our learned augmentation policy could clearly outperform the Cutout method while using different network architectures. This dataset is challenging because of the different settings between the train set and test set. In Figure. 4.3, the Cutout method failed to work on this testing image (right) while the proposed method made a correct prediction. It is noteworthy that Cutout did not outperform the baseline while using Wide-ResNet-28-2. In contrast, our translation-only augmentation approach consistently outperforms the baseline.

Model	Baseline	Cutout[41]	Ours(Trans)
Wide-ResNet-28-2	22.06	21.27	**22.25**
Wide-ResNet-40-2	22.31	23.49	**25.46**
ResNet-50	27.37	29.46	**33.08**

Table 4.2: Testing Classification Accuracy (%) on the Garbage dataset.

Iteration	1	2	3	4	5
Average	1.89	1.91	1.78	2.23	2.54
Std	1.35	1.45	1.22	1.49	1.68

Table 4.3: Average number and standard deviation of transformation types used in each transformation sequence on CIFAR-100 across all joint training iterations.

Results on CIFAR10 and CIFAR100 We first study how the classifier's performance changes over iterations in the proposed joint training scheme. Figure 4.5 plots the performance of Wide-ResNet-28-10 over iterations on CIFAR100. When we uses 14 kinds of transformations, we find the performance becomes stable around the third iteration, and the classifier reaches the best performance at iteration 5. Similarly, while only using translation for augmentation, the classifier achieves the lowest test error at iteration 3. This empirical study showed that the joint training scheme could largely improve system performance.

Table.4.4 reports the classifier performance (error rate) over the challenging CIFAR-100 while using Cutout [41] or the proposed policy-based augmentation method with only translations as the actions. We observe that our policy significantly outperforms Cutout, especially for Wide-ResNet-40-2 (error rate decreased by 3.4 with our policy, error rate decreased by 0.8 with Cutout). This result shows the effectiveness of learned translation compared to random patch erasing in Cutout.

Table.4.5 includes the results of state-of-the-art data augmentation methods. We implement

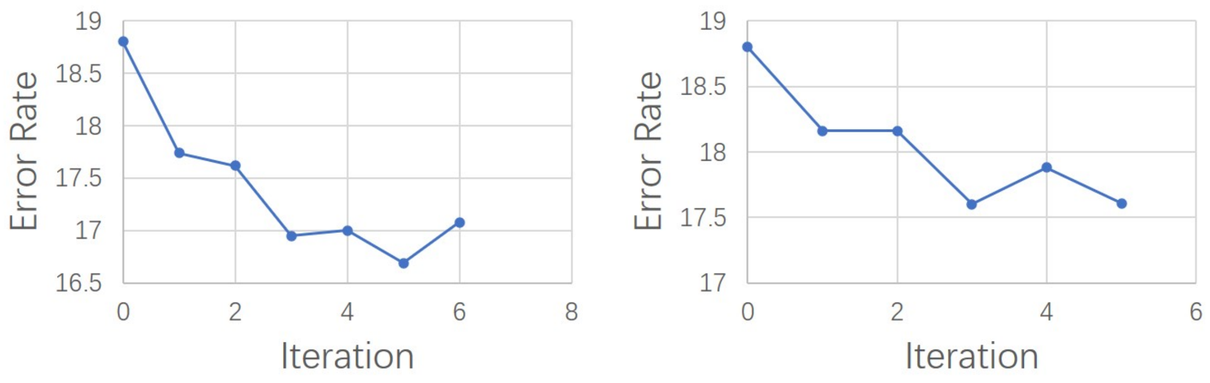


Figure 4.5: Classifier errors over iterations of augmentation.The proposed augmentation method uses 14 transformations (left) or translation only (right).

Dataset	Model	Baseline	Cutout [41]	Ours(Trans)
CIFAR-100	Wide-ResNet-40-2	26.00	25.2	**22.58**
	Wide-ResNet-28-10	18.80	18.4	**17.60**

Table 4.4: Classification error rate (%) on testing set of CIFAR-100. The proposed method only used the translation actions. See texts for more details.

the proposed policy-based method with all the 14 transformations. We also report for each augmentation approach (last row) the relative size of the augmented data comparing to the original training data. From the table, we observed similar or higher performance compared to previous methods on both datasets while using a much less number of augmented samples. For example, the AA method [34] employs 25 augmentation sub-policy, which results in 75 different transformed images for each original image. The proposed method augments each image less than 50 times on CIFAR100 and less than 30 times on CIFAR-10. The above comparisons indicate the efficiency of sample-specific augmentation policy over dataset-wide augmentation policy.

Dataset	Model	Baseline	AA [34]	PBA [74]	Fast AA [115]	Faster AA [69]	RA [35]	Ours
CIFAR-10	Wide-ResNet-28-10	3.90	2.60	2.60	2.70	2.60	2.70	2.70
	Shake-Shake (26 2x96d)	2.90	2.00	2.00	2.00	2.00	2.00	2.06
CIFAR-100	Wide-ResNet-40-2	26.00	20.70	-	20.70	21.40	-	20.38
	Wide-ResNet-28-10	18.80	17.10	16.70	17.30	17.30	16.70	16.69
Augmentation Size		–	75×	200×	75 ×	75×	196×	¡50×

Table 4.5: Classification error rate (%) on testing sets of CIFAR10, CIFAR-100. The last row lists the number of transformed images for each training image.

We visualize several images and their sequential transformations and report the overall statistics of the selected actions on CIFAR-100. In this set of qualitative experiments, we employ the proposed policy-based method with translation only since this transformation can be better visualized than others. Fig.4.6 plots multiple images (column 1) and the sequence of transformed images. We can observe that the trained policy can generate well-diverse yet valid augmented images which are still recognizable after transformations. This well demonstrates the effectiveness of the proposed reward and stop condition. Fig.4.7 plots the transformed versions of an image over iterations.We can observe that the agent tends to translate the image to different directions across iterations, resulting in localized images highlighting different parts of the object. The iterative joint training scheme dramatically increases the diversity of augmented images. Note that, at iteration 4, the agent tends to stop further augmentations and output very similar augmented images over the sequence. This is because the current classifier, after multiple iterations, can well recognize most transformed images and is less motivated to transform the original image.

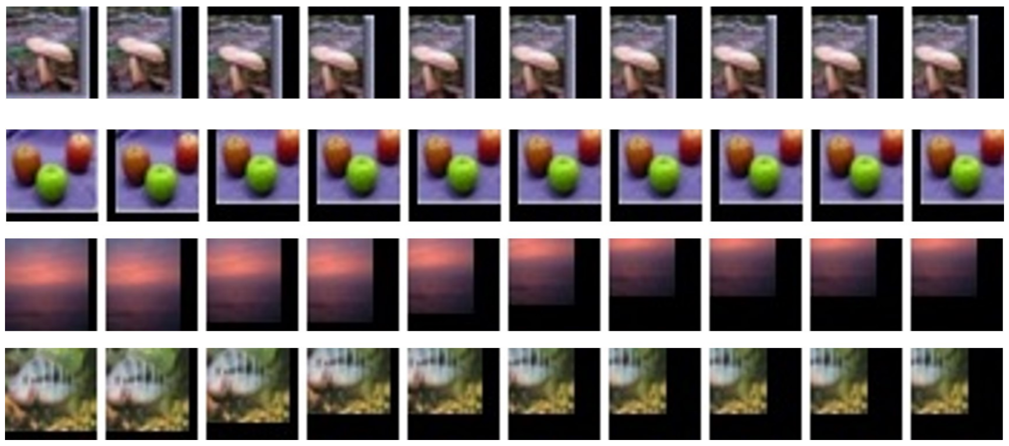


Figure 4.6: Examples of images transformed by our method. Two transformations: TransX and TransY are used by the agent.

We calculate the number of transformation types that each sequence might involve while using 14 transformations over CIFAR-100. Figure 4.8 shows the frequency of transformation types in the transformation sequences. We can observe that the agent used less than 3 transformation types for most transformation sequences. Table 4.3 also lists the average

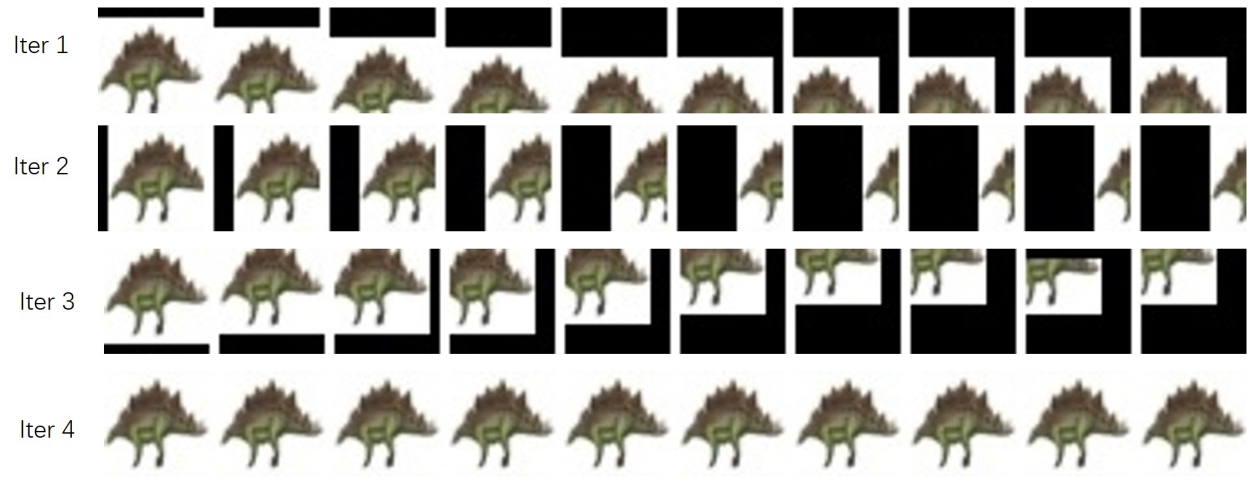


Figure 4.7: Examples of images generated at different iterations. Two transformations: TransX and TransY are used by the agent.

number of transformation types across iterations. On average, the first iteration uses 1.89 transformation types and the fifth iteration uses 2.54 transformation type. The above statistics shows that the iterative joint training scheme can actually diversify the ways to augment images and thus improve the quality of the augmented data.

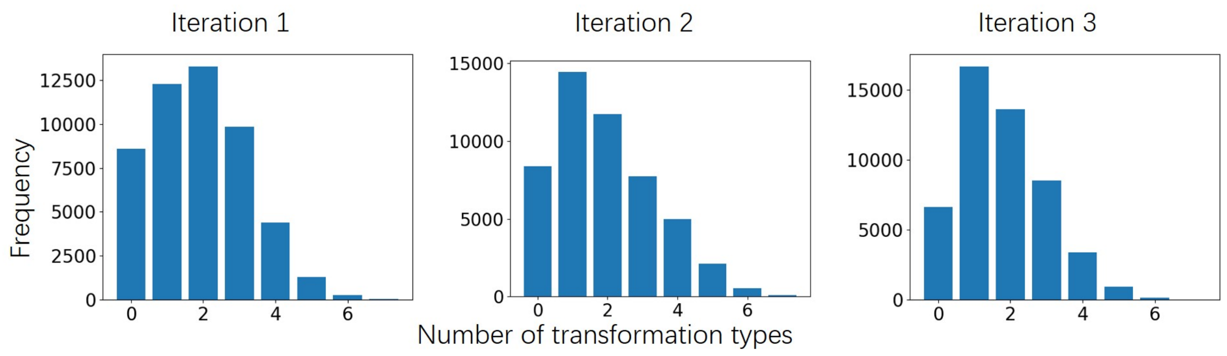


Figure 4.8: Frequency of transformation types in the augmentation sequences for CIFAR-100.

4.6 Conclusion

We developed a policy-based sequential image augmentation approach to augment a training image according to its visual content. We cast the search for sample-specific transformation sequences as a sequential decision process and introduce a method to jointly train the classifier and agent in the reinforcement learning settings. Experiments on both newly collected image dataset and public datasets show that our method achieves similar or better performance than previous augmentation methods while using significantly less transformed images. Notably, even with only translations, our method significantly outperforms the popular Cutout method. Our augmentation approach can be readily extended to solve other multimedia content understanding tasks, e.g., object detection, video classification, and video parsing, etc.

Chapter 5

Learning Data Augmentation for Scene Text Recognition

5.1 Introduction

Recognizing text in natural images is the foundation of many intelligent applications, e.g., autonomous driving [163], assistive reading for blinding persons [228]. In recent years, deep learning methods achieve significant success in scene text recognition [179, 37, 213] especially when a large STR dataset is available. To achieve better performance in application scenarios, training with the data from the target scenes can help better improve the model to recognize text and understand specific conditions [122], e.g., blurring, text size, and font type. However, building a large real-world STR dataset is expensive. Therefore, data augmentation is usually applied to improve model performance in practice.

In the past literature, data augmentation has been widely utilized in computer vision to increase the size of training images with no labor cost. A typical augmentation method is to apply an image transformation (e.g., rotation) to a training image, which results in a new training image paired with the original image labels (e.g., class label in image classification). One might manually adjust the transformations (types, magnitudes, etc.) in order to impose variance in the augmentation operations and increase sample variance. Image augmentation can effectively improve system performance while addressing multiple computer vision tasks, e.g., image classification [100, 71], object detection [237], image segmentation [172], and person re-identification [242]. However, a practical challenge is how to select the optimal image transformations for a training set or a specific training image. Actually, the choices of transformations form a large hyper-parameter space due to the combination nature.

There are recent works [34, 115, 69, 74, 35, 110, 128, 6] emerging to automatically search for the optimal augmentation strategies according to certain evaluation metrics, e.g., model evaluation accuracy trained with the augmented dataset [34]. The search for the optimal transformations might be performed for the whole dataset or each image. The former will apply the same transformations over all the images of one dataset, whereas the latter aims to find the optimal transformations for each training image. The dataset-wise strategy is characterized by its simplicity yet suffers from lower performance or lower data efficiency than the sample-specific strategy [110]. In this work, we aim to learn to determine the optimal augmentation strategy for each training image of scene text recognition (STR), a classical computer vision task that aims to recognize a character sequence with arbitrary length in a text image.

Figure 5.1 shows exemplar errors in scene text recognition caused by image transformations. The first row includes two text images and the other rows (2-7) show multiple transformed versions of those two images. After being transformed, the text images become nearly unrecognizable even for human being or well-trained STR networks. Including these transformed images in training is equivalent to adding noises or outliers to the training process. It is therefore needed to explore augmentation strategies finding optimal transformations. There are works exploring augmentation strategies design for STR [128, 138, 6]. However, methods in [128, 138] may limit to Thin-Plate-Spline (TPS) or affine transformations, and they did not optimize for a sequential transformation setting which can further increases the augmented sample variance. Atienza [6] employs a random augmentation scheme [35] to design a dataset-wise augmentation strategy, which may not be optimal for certain training samples and may have less augmentation efficiency.

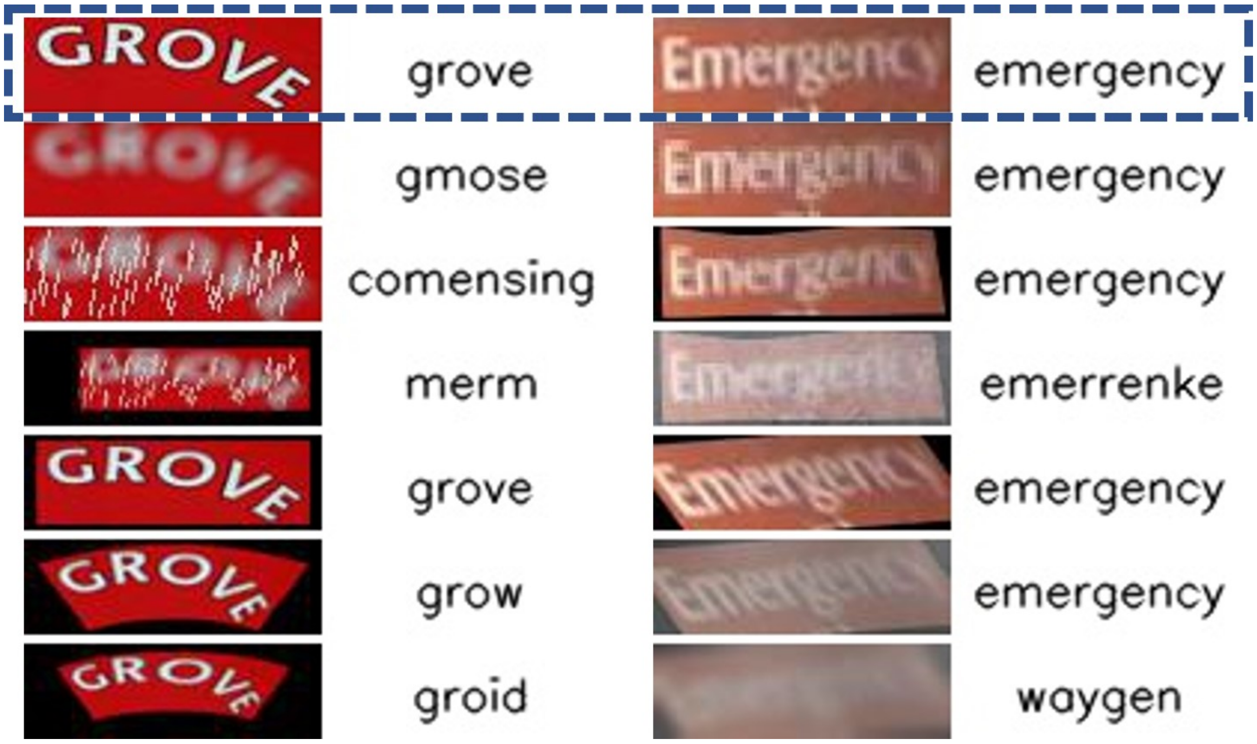


Figure 5.1: Typical errors in scene text recognition caused by image transformations. Row 1: two original text images with their text labels. Row 2-7: the augmented text images with different image transformations. Column 2, Column 4: text recognition results for images in column 1 and 3 respectively .

To address the above issue, we develop a policy-driven approach to learn to augment a training image based on its visual content. Our approach aims to find a sequence of image transformations that can improve STR model and preserve the image's perception features. Transformations in the sequence are applied one at a time over the training image and the transformed image at the previous step is used as the input to the next step. We cast the selection of image transformation as a sequential decision process and train a deep policy network to select the optimal transformation. The policy network is learned in the reinforcement learning setting to select an appropriate action to maximize future rewards. A stop condition is also imposed to avoid transformations that lead to unrecognizable samples.

We introduce a self-supervised schema to define the reward function and guide the learning of this agent. In particular, we partition the training dataset into two non-overlap subsets, each of which is used to train a separate STR model. We expect to learn to augment each subset to minimize the divergence between the two subsets. Inspired by the works on Learning Without Forgetting [112], we train a STR model on each of the two subsets and develop a distillment reward function to quantify the consistency between the predictions of the two STR models over the same image. In this way, the agent is trained to find a transformation that can maximize the reward or equivalently minimize the divergence between the two STR models. Technically, our approach is different from [112] which aims to learn a new model that behaves similarly with the old model over the same image set. In contrast, we assume the two models are given and fixed and aim to find a set of image transformations so the trained images receive consistent predictions from the two models.

This work has three major contributions. (1) We present a general policy-driven framework which trains an agent to find the optimal sample-specific sequential transformations for augmentation. 2) We propose to learn the policy network with a distillment reward function, which essentially employs cross-subset consistency to guide the learning of the agent. (3) We empirically show that the proposed method significantly boosts current text recognition models. It is worthy to note that the proposed method can be applied to multiple other image-based tasks, including image classification, object detection, etc.

5.2 Related Works

This work is closely related to four research streams in the areas of computer vision and multimedia computing.

Image Auto-Augmentation Traditional image augmentation methods employ manual augmentation strategies [71, 67, 75] which essentially allow researchers to manually specify the hyper-parameters of image augmentation, including transformation types, magnitudes and possibility used for data augmentation. For a specific task, the search for the optimal augmentation strategy remains challenging because the feasible space of augmentation strategies is huge due to the combination nature. For example, brute-force search or heuristics search methods are essentially impractical due to their high complexity. In the recent literature, researchers began to develop various optimization methods to address this challenge and automate the search of optimal augmentation strategies. These auto-augmentation methods achieved impressive successes in classical image tasks, including image classification [34, 35, 110] and object detection [246, 25]

In the literature, there are two types of augmentation strategies: dataset-wise and sample-specific. The dataset-wise augmentation strategy will apply the same set of transformations to all the images of the same dataset, whereas the latter employs a different transformation for each image. Cubuk et al. [34] present a search-based method to seek the optimal dataset-wise augmentation strategies for image classification. This method requires retraining of classifiers while training the agent and is time-consuming. Other works concentrated on reducing the searching complexity by evaluating rewards from trained classifiers [115, 69], searching the strategies during classifier training [74], or reducing the hyper-parameter space [35]. In contrast, the sample-specific strategy will optimize the transformations applied over each training image. Fawzi et al. [48] propose to find the perspective transformation leading to the worst classifier performance. Ratner et al. [162] design an adversarial training framework to transform an image so that the transformed images are still following the distribution of the original dataset. Li et al. [110] use a reinforcement learning framework to predict the transformation sequences on each image in order to improve the classification difficulty and employ a policy-driven method to augment training images for image classification tasks. In this work, we extend our method [110] to scene text recognition task and propose a self-supervised scheme with a distillment reward function.

Scene Text Recognition The goal of scene text recognition (STR) is to recognize character sequences that appear in images. There are mainly two types of approaches for STR: bottom-up and top-down. The former type of approach [205, 206, 113] first detects and recognizes individual characters, then groups characters into texts. One of the recent bottom-up approaches [113] utilizes a character segmentation model and generates text prediction by the character detection results.

The top-down approaches [178, 120] directly recognize the entire text from the image. Those methods usually treat STR as a sequence recognition problem and employ Sequence-to-Sequence (Seq2seq) models. A classical approach [178, 120] utilizes the Connectionist Temporal Classification(CTC) [58] an d directly predicts a fixed-length output sequence. The CTC loss encourages the model to match the output sequence with the ground truth sequence. Since consecutive characters in output may correspond to the same character in the image, a post-processing step is required for generating the final text predictions. Another stream of methods are based on the attention-based framework [179, 104]. They usually employ encoder-decoder architectures and utilize the attention mechanism in the decoder to recognize characters one by one until a stopping action is predicted. Previous works used 1-D attention [179] or 2-D attention [226, 108] in the decoder, and recent works also applied the attention scheme in the encoder [106]. In this work, we employed the prevalent attention-based methods as the baseline model in our experiments, and the proposed data augmentation method is essentially compatible with all STR models.

Image Augmentation for STR Image augmentation has been approved to be an effective technique for text recognition. Most image transformations, including image rotation, perspective distortion, blurring, etc., are applicable for augmenting synthetic text images [80], or real work text images [47, 144]. There are also transformations specially designed for text recognition. Wigington et al. [216] apply the elastic distortion on the handwritten text images. Chowdhury et al. [28] transfer text images to different domains (e.g., daylight, night pictures). Ren et al. [166] transfer the font style between different characters for novel text images. Atienza [6] summarizes a total of 36 image transformations for text images, including elastic distortion, pattern or noises, blurring and sharpness adjustment, weather effect, and other common image transformation functions, e.g. equalization. In contrast, Nuriel et al. [152] transform training images in the feature spaces and introduce a set of statistic metrics to guide the transformations.

Another critical challenge is how to find the optimal augmentation strategies for a STR model. Luo et al. [128] explore sample-specific Thin-Plate-Spline (TPS) augmentation by jointly training an agent network and STR model. The agent network predicts fiducial points movement distribution for TPS transformation. Meng et al. [138] design a gated module to apply either TPS transformations or affine transformations on each training image. The transformation parameters are estimated by a separate transformation module which aims to generate difficult samples and increase STR loss.

The proposed method extends the previous sample-specific augmentation methods [128] and [138] in three aspects. Firstly, these methods only work with a specific type of transformation (TPS or affine), whereas our method can work with all types of image transformations which can have discrete transformation magnitudes defined. Secondly, our method can provide sequential image augmentations. This means that our method can explore more complicated transformation combinations. Finally, we use reinforcement learning and design novel rewards to search for sample-specific sequential transformations. We empirically used the individual image transformations in [6] in our experiments to examine our method. We compare our methods with the method in [6] and [128].

Deep Reinforcement learning Deep reinforcement learning has been widely used in various image tasks, including scene text recognition. It involves an agent executing the actions in an environment that gives feedback with rewards. In the past, deep reinforcement learning has succeeded in machine tasks including robotic control [114, 61], game agent [143, 183], network architecture search [8, 247], image processing [231], etc. One common method in deep reinforcement learning is deep Q-learning [143] which uses an agent model to predict the Q values (expected accumulative reward after one action). The agent picks the action with the largest Q-value during inference. It has the advantage of a simple model design but it may suffer from the problem of convergence and Q value overestimation [201]. Double DQN [201] uses a target network, which has delayed weights updating from the agent model, to mitigate this problem. In this work, we employ Double DQN [201] to train an agent capable of providing the optimal image augmentation strategy.

5.3 Auto-Augmentation With Distillment Rewards

5.3.1 Objective: Sequential Image Augmentation

The goal of this work is to learn to augment a training dataset for scene text recognition. For each training image, we aim to find a sequence of atom image transformations that work the best for the training. Each atom transformation is defined by its transformation type (e.g., adding noises, curving the text image) and magnitude. The transformations are sequentially applied over the image, i.e., the transformed image at one step is used as the input for the next transformation. We denote the original image as I_0, the sequence length as T, and the sequence of transformations as f_1, f_2, .., f_T. so the transformed image I_t after t steps of transformations is

$$I_t = f_t \circ f_{t-1} \ldots \circ I_0. \tag{5.1}$$

The $f \circ I$ represents the transformation f operating on the image I and returning the transformed image. The sequence formulation of transformation exponentially increases the transformation space size and may introduce more variance in augmented samples. While brute-force searching in such transformation space for each image is almost intractable, we treat it as a sequential decision problem and solve it with a reinforcement learning algorithm. We note that any transformation function can be used in our augmentation strategies, which gives us the flexibility for extending our strategies with novel transformations in the future.

5.3.2 Overview of Our Method

Figure 5.2 summarizes the sketch of the proposed approach. We split the training dataset into two subsets and train one STR model on each subset. Our goal is to augment one subset to approximate the distribution of the other subset and vice versa. We formulate this self-supervised objective as a reward function and use it to guide the training of a policy network with the observed experiences. Our framework will be executed in an iterative fashion. Firstly, two STR models are trained over the two subsets respectively. Secondly, a policy network is trained to maximize the accumulative rewards which are defined over the two STR models. Thirdly, an agent will follow the current policy network to predict the sequence of image transformations for each training image, and the augmented images are used to train the STR models. These iterative steps repeat until the STR models converge.

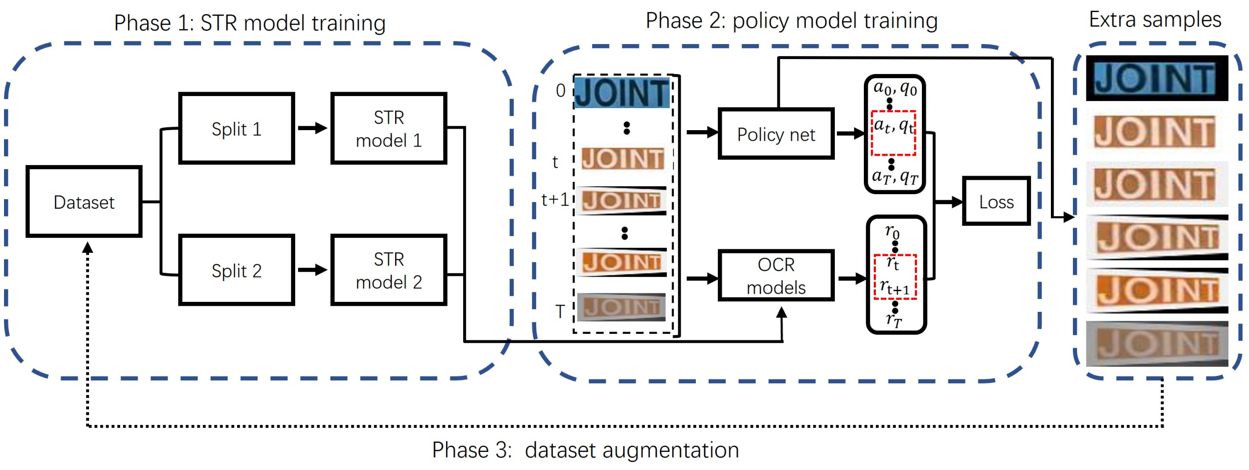


Figure 5.2: Illustration of our data augmentation framework. a_t, q_t, r_t represents the actions, Q-value and reward respectively at step t. See texts in Section 5.3.2 for more details.

5.3.3 Policy Learning

We employ a deep Q-learning network (DQN) to predict the optimal transformations for an image according to its visual content. For each image, finding the optimal sequence of transformations is a sequential decision process, which may have a total of T steps. At step t, we have the current state S_t which is a function value of current image I_t and previous actions a_0, .., a_{t-1}. The DQN model takes S_t as input and predicts the Q value for each action, where the Q value is the expected accumulative rewards after executing this action at step t. The action a_t which has the maximum Q value in the action space A will be included in the policy and executed next, i.e., transforming I_t to I_{t+1}.

$$a_t = \text{argmax}_{a \in A} Q(S_t, a) \tag{5.2}$$

This process can be represented as:

$$S_0 \xrightarrow{a_0} S_1 \xrightarrow{a_1} \cdots \xrightarrow{a_{T-1}} S_T$$

We note that other reinforcement learning algorithms, e.g., action-critic approach [65], may replace the DQN in our framework. We employ the deep Q-learning method to explore the discrete action space. In the rest of the subsection, we will show the details of the states, action, reward, stop condition, and objective function in the policy learning.

Action An action a_t depicts the transformation type and magnitude at time t. Any suitable transformation for text images can be included in our action space. Suppose we have n transformation functions $g_1 \ldots, g_n$, and function g_i has m_i number of magnitudes, then we have a total of $\sum(m_i)$ actions. We also include a stopping action to stop the transformation early when the image becomes unrecognizable to STR models. Overall, our agent have $\sum(m_i) + 1$ possible actions.

State State S_t includes the information we provide to the agent at time t, which can be a function of all previous images and previous actions. Our choice of S_t is the combination of current image I_t and previous action a_{t-1}. The S_t is used as the agent input unless the previous action is stopping. Therefore, a_t can be encoded as a one-hot vector for all possible actions except stopping. At step 0, we set the initial action a_{-1} as a zero vector in S_t. This state guides the decision of the agent by the image content and previous action. For example, an agent may not further curve the text image if it contains curved text, and the agent may not continuously repeat a transformation (e.g., equalization) multiple times while the image is not changed by the transformations.

Distillment Reward We propose a distillment reward to implement the proposed self-supervised augmentation scheme. Let D_1 and D_2 denote the two non-overlapping subsets of training images. The key idea is to augment training images in one subset so that the distribution of the augmented subset becomes more similar to the distribution of the other subset. Let M_1 and M_2 denote the two models trained from the D_1 and D_2 respectively. Like [112], the distribution of D_1 or D_2 can be well represented by M_1 and M_2, respectively. If M_1 and M_2 have similar predictions over the same image, it is reasonable to assume that the distributions of D_1 and D_2 are similar to each other and vice versa. Accordingly, an image receiving the same predictions from M_1 and M_2 can be used to reduce the difference between the distributions of the two datasets. Therefore, our goal is to learn a policy that can produce augmented images receiving similar predictions from M_1 and M_2. Let X denote a training image, Y the probability vector of labels, and X' a transformed image by an action. The reward of this action is defined as:

$$R = \mathcal{D}(M_1(Y|X), M_2(Y|X)) - \mathcal{D}(M_1(Y|X'), M_2(Y|X')) \tag{5.3}$$

where $\mathcal{D}()$ measures the divergence between two conditional distributions. In our implementation, we set $\mathcal{D}$ to be the L_1 distance between the prediction probabilities of the two STR model over the same text images. We have,

$$\mathcal{D}(M_1(Y|X), M_2(Y|X)) = \left\| \hat{Y}_1 - \hat{Y}_2 \right\|_1 \tag{5.4}$$

where $\hat{Y}_1$ and $\hat{Y}_2$ are the vectors of probabilities for characters predicted by M_1 and M_2. $\hat{Y}_1$ and $\hat{Y}_2$ are padded with empty character token so that $\hat{Y}_1$ and $\hat{Y}_2$ has the same length. Figure 5.3 visualizes a training image, two transformed images and their network outputs over the two models M_1 and M_2. The image transformation associated with the dotted box received a higher reward based the above equation. The above metric directly measures the differences between the two STR models' outputs over the same training image. The images X can be selected from D_1, or D_2 or the whole dataset. It is also noteworthy that one might replace the L1 distance with other distance metrics, e.g., the edit distance between predicted texts.

Stop Condition We introduce a stop condition which, once triggered, can terminate the sequence of transformations over an image. As shown in Figure 5.1, some image transformations might lead to unrecognizable images which are not useful for the training procedure. In particular, we apply the current STR model over each transformed image and check if

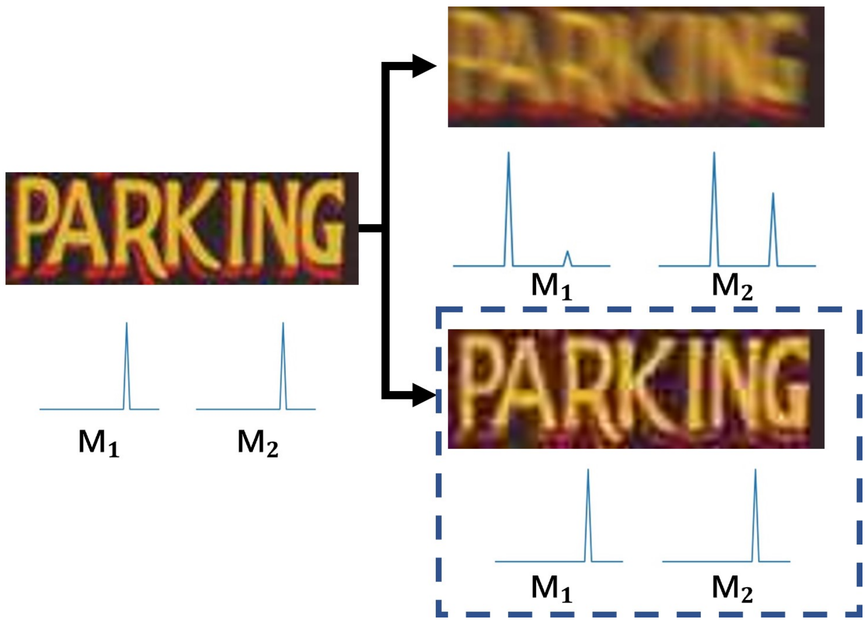


Figure 5.3: Samples of the proposed distillment rewards. Left: original image; Right: two transformed images. The curves under each text images are the probability distribution of characters predicted by the two STR models, M_1, M_2, respectively. The transformed image in a dotted box obtained similar predictions from both models and the relative action received a higher reward.

its prediction labels differ from the original labels. Once changed, we will terminate the transformation sequence and withdraw the action that may corrupt the image content. In this way, an early stop condition is imposed on the sequential augmentation process. Let $\mathrm{ED}(S, T)$ denote the edit distance between the estimated text V and ground truth text T. We employ the normalized edit distance to define the stop condition:

$$\mathrm{ED}(V, T)/\textsc{max}(\textsc{len}(V), \textsc{len}(T)) > \theta \tag{5.5}$$

where θ is a pre-defined constant and $\textsc{len}()$ return the length of a text. While training the STR models and policy network, the above condition can be used to avoid unrecognizable image augmentations. We will include this stop condition in the inference time of the agent to avoid unrecognizable scene text images included in the training set. This stop condition can be considered as a constraint to enforce the transformed images in a "trusted region" defined in the previous work [48].

Objective function We employ the double DQN approach [201] to train the augmentation agent, which is then used to predict the Q value of an action. Specifically, there are two agent models during training: one is the online agent and the other is a target agent whose weights are updated by copying the online agent every τ steps. The training of the agent models are guided with the following temporal difference loss

$$\textsc{Loss} = (y_t - q(S_t, a_t))^2 \tag{5.6}$$

where

$$y_t = \begin{cases} r_t + \gamma * max_{a'} q'(S_{t+1}, a') & 1 \leq t < T \\ r_T & t = T \end{cases} \tag{5.7}$$

γ is the discount factor, and q' is the Q value predicted by the target network. This definition follows the Bellman equation to reduce the Q value estimation errors within two consecutive steps. Usage of the target model may mitigate the overestimation problem in DQN [201].

5.4 Experiments

5.4.1 Datasets

We will apply our method over Synthetic datasets and real-world datasets all of which are public benchmarks for scene text recognition.

Synthetic datasets We used 2 large-scale synthetic datasets for STR model pretraining.

MJSynth (**MJ**) [80] contains 8.9 M synthetic text images. The data generation algorithm blends the word, border, and shadow on the real-world images. Perturbation of projective distortion and noises may add to the synthetic text in the generation process. SynthText (**ST**) [64] dataset contains 5.5 M synthetic text images. This dataset was introduced for scene text detection where texts are blended on natural images. The text regions are cropped for training and evaluating scene text recognition models.

Real-world datasets We use 7 real-world datasets for STR model training. IIIT5K-Words (**IIIT**) [141] have 2000 text images for training and 3000 images for evaluation. The dataset is collected by searching words on Google image search. Street View Text (**SVT**) [205] contains cropped text from Google Street View. It has 257 training text images and 647 evaluation images. ICDAR2003 (**IC03**) [127] refers the dataset on ICDAR 2003 Robust Reading competition. The competition's goal is to read camera-captured scene texts. It contains 1156 and 1110 images for training and evaluation respectively. We use the images without non-alphanumeric characters, which leads to 867 images for evaluation. ICDAR2013 (**IC13**) [90] extends the IC03 dataset, and it has 848 images for training and 1095 images for testing. We prune the non-alphanumeric character images and have 1015 images for evaluation. ICDAR2015 (**IC15**) [89] was introduced in 2015 Robust Reading competition. It provides 4,468 training images and 2,077 evaluation images. Real-world scenes are collected by moving persons wearing Google Glasses, which may lead to blurry and perspective distorted text images. SVT Perspective (**SVTP**) [159] is also collected from Google Street Views while its images contain perspective distortions. It has 645 images for evaluation. CUTE80 (**CT**) [169] contains curved text images from the natural scenes and has 288 text images for evaluation.

Training, validation and evaluation datasets We use the synthetic datasets for STR model pretraining and then finetune the model on the real-world dataset. We split the evaluation set of each real-world dataset (IIIT, SVT, IC03, IC13, IC15, SVTP, CT) into three subsets by the ratio of 35%, 35%, and 30%. The first two subsets are used for text recognition model training, and the last one is used for evaluation.The data augmentation is only applied to real-world datasets. We use the union of the training sets IC13, IC15, IIIT, and SVT as the validation dataset.

5.4.2 Implementations

Image transformations We adopt the augmentations for text images in [6], which includes 36 types of transformations. The transformations include six categories: elastic deformation by warping fiducial points, geometric transformation, adding grids, adding noise, blurring the image, simulating weather effects, camera sensor tunable setting changing simulation, other image processing functions. Each type of transformation has 3 magnitudes, which leads to 108 actions for our agent.

Baseline methods We use two STR models in our experiments: RARE [179], and ABINET [47]. The RARE model is pretrained with the AdaDelta optimizer (decay rate is 0.95) on synthetic datasets. The batch size is 192 while the mixing rate between ST and MJ is 0.5/0.5, i.e., 96 samples from ST and 96 samples from MJ in each mini-batch. The pretraining lasts for 3×10^5 iterations and the last saved model is used as pretrained weights. We finetune the RARE model on original or augmented real-world datasets with AdaDelta optimizer (decay rate is 0.95) for 3×10^4 iterations. On the other side, we directly use the pretrained ABINET model (trained on ST and MJ datasets) provided by [47], and finetune it on the real datasets or augmented real datasets for 500 epochs with Adam optimizer (initial learning rate = 0.0001, batch size = 64). The learning rate is decayed to 1×10^{-5} after 300 epochs.

Policy model During our policy model prediction, the input image is firstly processed by a feature extractor module (4 convolutional layers, 1 fully connected layer). Then the resulting 32-dimensional feature and the previous action one-hot vector are concatenated as the input for a one-layer LSTM. Finally, the LSTM output is transferred to the Q value predictions by a fully connected layer.

At each step during training, an ϵ-greedy exploration is used for generating newly observed data, i.e., the online agent will pick a random action with a possibility of ϵ and pick the action with maximum predicted Q value otherwise. The total steps for training are 1e6, and ϵ is decreased linearly from 1 to 0.1 between 5e4 steps to 1e5 steps. The observed data is stored in the replay memory of size 2e5. Each image is transformed until the stop action is applied or maximum length (6) reached, and this process is called one episode. After 5e4 step, we randomly select 64 episodes from the replay memory for loss calculation and agent weights update every 4 steps. The target Q network is updated by copying the online agent model weights every 1e3 steps. An Adam optimizer with an initial learning rate of 1e-4 is used in training, and the learning rate is decreased exponentially to half every 1e5 steps.

5.4.3 Distillment Reward on Toy data

We first apply the proposed method to augment an one-dimensional dataset and study the effectiveness of the proposed distillment reward. Figure 5.4 summarizes the augmentation process. Consider two datasets of n 1-D samples drawn from the standard Gaussian distribution $\mathcal{N}((0,1)$. Let X_i denote the i-th sample of a dataset. The distribution of a dataset can be estimated as:

$$\mu = \frac{1}{n}(\sum_{i=0}^{n} X_i) \tag{5.8}$$

$$\sigma^2 = \frac{1}{n-1}\sum_{i=0}^{n}(X_i - \mu)^2 \tag{5.9}$$

. Let $j = 1, 2$ index the two datasets. Due to the nature of sampling, the estimated means and variances from the two datasets will be different, as shown in Figure 5.4 (top-left).

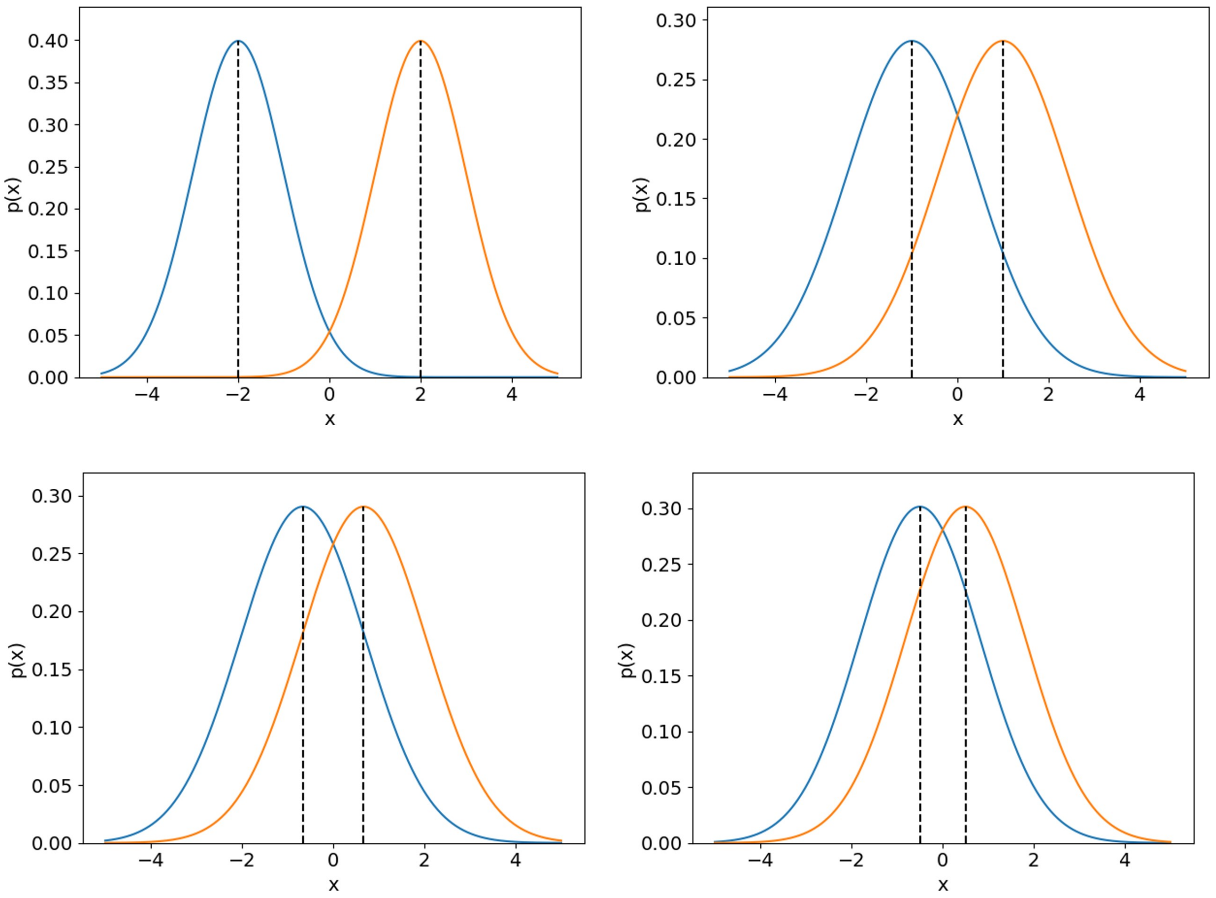


Figure 5.4: Toy experiment for the proposed reward. The blue and orange curves show the possibility density of the estimated Gaussian distribution from two datasets respectively. Top-left: distribution estimation before augmentation. Top-right, bottom-left, bottom-right: distribution estimation after augmenting the datasets for 1, 2, 3 times respectively. For each dataset of n samples, we add n augmented sample at each step of augmentation.

The goal of this toy experiment is to augment these two datasets to minimize the difference between their distributions. Like the proposed method, we progressively add more samples into the two datasets. Differently, we employ a greedy strategy to find a sample x that can minimize the probability difference between the two estimated distributions. By definition, the optimal x is the middle point of mu_1 and mu_2. To accelerate the augmentation procedure, we draw samples from the distribution $\mathcal{N}((\mu_1 + \mu_2)/2, \sigma)$ and add them to one dataset. The augmented dataset is characterized by a revised distribution with the following parameters:

$$\mu_1^{t+1} \approx \frac{2t+1}{2t+2} * (\mu_1^t) + \frac{1}{2t+2}(\mu_2^t) \tag{5.10}$$

$$(\sigma_1^{t+1})^2 \approx \frac{t}{t+1}(\sigma_1^t)^2 + \frac{1}{t+1}\sigma_3^2 + \frac{t}{4(t+1)^2}(\mu_1^t - \mu_2^t)^2 \tag{5.11}$$

where the t indicates the t-th time augmenting the first dataset. μ_1^t and σ_1^t are the estimated mean and standard deviation before the t-th augmentation. Similarly, we can add the drawn samples to the other dataset and estimate its mean and variance μ_2, σ_2 accordingly. Figure 5.4 graphically visualize how the two distributions approach each other over augmentation steps. After a large set of samples is augmented, the two distributions will eventually converge to be the same one. This toy experiment suggested the proposed distillment reward function can effectively minimize the divergence between the distributions of the two datasets.

5.4.4 Experiments on Text DataSets

We first train the STR models on the synthetic datasets and then finetune them on our training dataset without augmentations, and use them as our baselines.

Learned policy We iteratively train the RARE model [179] and policy networks as introduced in section 5.3.2. To obtain our proposed reward, we finetune the STR models on each split of the training set. After each augmentation iteration, the finetuning is based on the augmented dataset which includes the original samples, augmented sample from early iterations, and the augmented sample in the current iteration. The augmented samples include both intermediate transformed images and final images after applying the transformation sequence. The iterative augmentation process stops when RARE's performance converges. Then we report the STR model performance trained on the full training dataset.

Random policy We implement a naive version of the proposed policy-driven method that applies random transformations over training images, and we follow the same iterative augmentation process as above. In each augmentation iteration, we apply 6 random sequential transformations for each image. The augmented training dataset is used to train the STR network, and we compare this result with other augmentation methods.

Transfer policy among models We evaluate how well the augmentation policy learned from RARE can work with other STR models. We apply the same policy as the learned policy above to train one of the state-of-the-art STR model ABINET [47].

Results Figure 5.5 reports the testing accuracy of the RARE model at each iterative step that we described in Section 5.3.2. We observe a significant performance increase before iteration 5, which indicates the models' generalization ability improves with the learned augmentation policy. According to this result, we use the augmented samples in and before iteration 5 for policy transfer experiments. Examples of the learned augmentation strategies are shown in figure 5.6. Among different augmentation iteration, the policy network tends to predict different transformations for augmentation, which indicates that our policy may prefer to explore new data variants which do not appear in previous iteration.

Figure 5.7 reports the statistics of our learned augmentation policy. Over the iterative training, we observe that the policy network tends to predict longer transformation sequences in later iteration (Row 1, left v.s. right), which suggests that the policy network tends to find more complicated transformations by extending the transformation sequence in later iteration. We also see the differences of the predicted actions between different iterations and different transformation steps (Row 2-4), which indicates that our learned policy can

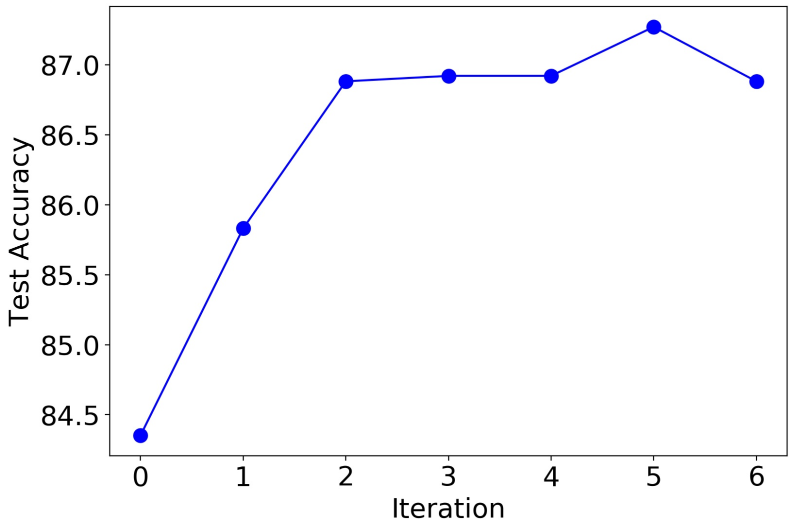


Figure 5.5: Testing accuracy of RARE model over multiple iterations of augmentation.

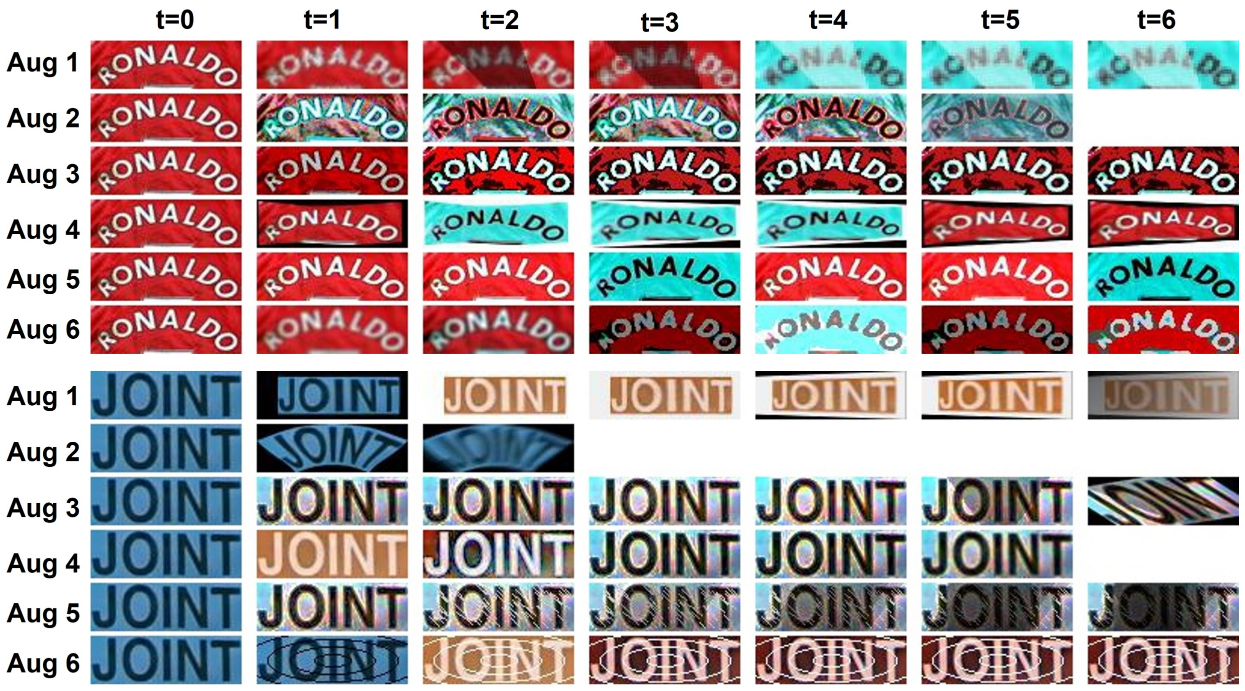


Figure 5.6: Two examples of the augmented samples from our iterative training. Each row includes the original image (t=0) and a sequence of transformed images in one augmentation iteration (t=1,2,...,6).

bring rich data variance to the training set.

Table 5.1 reports the quantitative results of these STR methods. Our STR models are

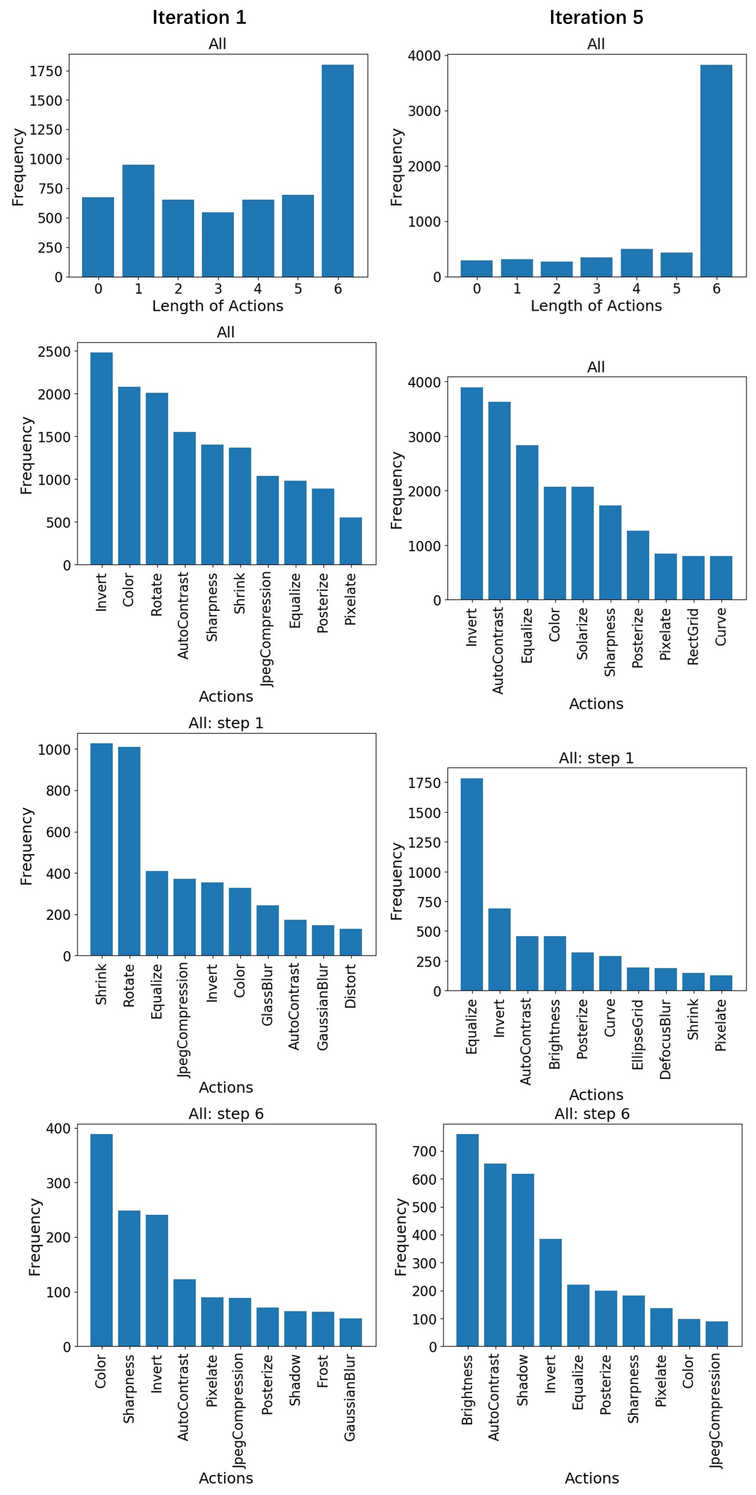


Figure 5.7: Statistics of the learned augmentation policy. Column 1, 2: statistics of augmented samples when the policy network is trained for the 1st time and 5th time, respectively. Row 1: distribution of transformation sequence length. Row 2, 3, 4: distribution of transformation types in all steps, the 1st step, and the 6th step, respectively.

Table 5.1: Text recognition accuracy (%) of various methods. See texts for more details.

Dataset	CT	IC03	IC13	IC15	IIIT	SVT	SVTP	Total
RARE [179]	80.68	96.17	92.46	69.6	88.67	89.23	80	84.35
+ Random Policy	76.14	96.17	91.48	69.92	90.33	91.28	84.1	85.21
+ RandAug	78.41	96.94	93.44	73.6	90.78	94.87	87.69	87.19
+ RandAug (n=6)	77.27	96.55	89.84	71.2	88.56	93.85	82.56	84.86
+ Our Learned Policy	82.96	97.32	93.77	73.92	92	94.36	85.13	87.7
ABINET [47]	88.64	96.55	93.12	80.48	95.33	95.9	88.21	90.85
+ Random Policy	86.36	95.02	92.13	76.64	92.33	95.9	92.31	88.83
+ RandAug	89.77	96.17	93.12	79.68	94.33	97.44	91.8	90.7
+ RandAug (n=6)	82.96	95.02	91.15	77.44	89.33	95.9	91.28	87.66
+ Our Learned Policy	88.64	96.55	93.44	80.16	95.00	98.46	94.36	91.36

trained with five settings: no augmentation, random policy augmentation, RandAug [35] augmentation, RandAug augmentation with transformation sequence length **n** of 6, and our learned policy augmentation. For RandAug, we follows the settings in [6] and report the performance model with the best validation accuracy. From the table, we observe that our augmentation policy significantly increases model accuracy across different evaluation sets. Notably, we achieve a margin of 3.35% for the total accuracy using the learned policy compared to the baseline model without augmentation, and surpassed the random policy and RandAug by 2.49% and 0.51%, respectively. Those results indicates that our learned policy effectively find beneficial augmented samples, and our data augmentation method can reach a comparable or better performance than the state-of-the-art dataset-wise augmentation method. Our policy surpasses RandAug (n=6) by 2.84%, which indicates the advantage of our methods in exploring longer sequence of transformations than RandAug. When we transfer the augmentation policy to ABINET, we have an accuracy increase of 0.51% with our learned policy compared to baseline without augmentations. We also observe that random policy harms the performance and RangAug does not significantly affect the performance. These results show that our policy can transfer well on different models to improve the text recognition performance.

5.4.5 Comparison with Method [128]

We further compare our method to a sample-specific augmentation method [128] which aims to learn to augment an individual text image. Since [128] did not release their source codes for training, in this experiment, we train our method in the exactly same setting as [128] and compared the results of our method to the results in their paper. We iteratively train the RARE model [179] and a policy network on the Real-50k dataset, and use the learned augmentation policy to train the ASTER model [180]. Following the training setting in [128], the Real-50k dataset includes the training sets of SVT, CT, IC13, IC15, and COCO-T, which leads to around 50k training images. We compare our results to the best result and the baseline result without augmentation from [128] in Table 5.2. [128] reported a text recognition accuracy of **66.5**% on the combined testing dataset of IIIT, SVT, IC03, IC13, IC15, SVTP, and CT. With the same STR model ASTER [180], our policy-driven augmentation method obtains an accuracy of **67.2**%. This significant improvement suggested that our augmentation method can achieve state-of-the-art text recognition accuracy.

Table 5.2: Testing accuracy when trained with Real-50k dataset

Method	Test Acc (%)
ASTER [128]	54.1
+ Learn to Augment [128]	66.5
+ Our Augment	67.2

5.5 Conclusion

We developed a policy-driven approach that can learn to sequentially augment a training image for text recognition. A policy network is introduced to predict the optimal sequence of transformations for an image based on its visual content. To train the policy network, we introduce a self-supervised scheme which divided the training set into two subsets and trained an agent to augment the two subsets so that they become more similar with each other.

Like the previous learning without forgetting method [112], we trained a text recognition model for each of the two subsets and introduced a disstillment reward to characterize the similarity of the two models. We then trained an agent in the reinforcement setting to select transformations that can maximize the immediate and future distillment rewards.

Our experiments over both toy data and real-world text images show that the proposed augmentation framework can significantly improve text recognition performance and reach comparable performance as state-of-the-art auto-augmentation methods on scene text recognition datasets. The augmentation policy also transfers well among different STR models. The experiment on toy data demonstrates that our augmented samples reduce the distribution discrepancy among two data subsets, which leads to decreased prediction discrepancy of models. This work concentrates on the application over scene text recognition but the proposed augmentation with distillment rewards can be adapted to other image-based tasks, e.g., classification, face recognition, boundary detection, landmark recognition, etc.

Chapter 6

Improving Loss Function for Pseudo-label Learning on Whistle-extraction Data

6.1 Introduction

There are currently 72 recognized species of odontocetes (compared to only 14 baleen whale species), of which approximately two-thirds are known to produce whistles [218]. Odontocete whistles are highly complex and variable communication signals and contain not only information about the species that produced the vocalization [54, 86] but also behavioral states [195, 187], and in some cases, individual identity [82]. Consequently, marine biologists frequently deploy hydrophones to study these marine mammals. However, the mid to high-frequency signals [155] require high sampling rates (typically 200 kHz), resulting in extensive sound archives to be analyzed. Automated extraction (and subsequent species classification) of whistles from these data remains a significant challenge in the field of animal bioacoustics, and new methods are needed to make the extraction process more efficient and reliable.

Most odontocete whistles feature characteristic contour shapes in the time-frequency (t-f) domain. Whistle extraction aims to determine the t-f bins of whistle in spectrograms, which then facilitates the subsequent tasks, e.g., classification of these acoustic signals to the species level. While biologists can manually extract whistles as t-f contours in spectrograms, this task is highly labor intensive. To speed up the acoustic analysis process, various automated whistle extraction algorithms have been developed over the years (e.g., [132, 215, 137, 171, 54, 60, 109, 208, 33]).

Whistle extraction methods (e.g., [171]) typically contain two steps. Most algorithms start by using peak detection algorithms to find regions of high energy that may belong to a whistle. In most cases, these are then examined to see if they are near other peaks and, therefore, likely to be parts of a whistle. This may be done using deterministic (e.g., [137]) or probabilistic (e.g., [60]) trajectory models. These sets of peaks may be subjected to additional processing but are eventually reported as a whistle contour.

More recently, deep-learning-based methods have been applied to whistle extraction. Li et al. [109] trained convolutional neural networks (CNNs) to find candidate t-f bins of whistles in spectrograms. The CNN model outputs a confidence map for the spectrogram, where each t-f bin has a confidence score of whether this node contains part of a whistle signal. T-f nodes with confidence scores above a pre-defined threshold are connected into whistle contours using a graph search method [171]. Compared to using spectral peaks to extract whistles, the deep-learning-based method improved the F1-score by around 20% on a 2-species (*Delphinus capensis*, *Tursiops truncatus*) benchmark dataset. However, training the model used thousands of manually annotated whistles, with analyst annotations produced over several months.

To facilitate the application of deep-learning-based methods in situations where large datasets are unavailable, we explore ways to train the model with pseudo-labels. Pseudo-labels are approximative labels that are generated by using other methods, such as the examples discussed in the prior literature above. We do not expect the pseudo-labels to be as accurate as those produced by human analysts. Our setting does not require human analysts to annotate whistles nor to validate the generated pseudo-labels. Training deep neural networks with pseudo-labels is challenging due to the increased errors, or noise, in the pseudo-labels as compared to analyst-generated ones. Neural networks learn by adjusting network parameters to minimize a loss function that measures the difference between predictions and expected labels. Consequently, when a deep neural network is trained to fit these noisy pseudo-labels well, the model will result in unsatisfactory performance. To address this challenge, we introduce modifications to the loss function of model training that regularize the label noise.

The machine learning community has proposed three categories of modified loss functions to improve model robustness to label noise [188]. Firstly, researchers developed novel distance metrics in loss functions. Ghosh et al. [53] showed that symmetric loss functions, e.g., mean absolute error (MAE), led to a smaller performance drop compared to nonsymmetric loss functions, e.g., categorical cross-entropy (CCE), when there are noisy labels. Zhang et al. [240] further generalized MAE and CCE with negative Box-Cox transformation. Wang et al. [210] added reversed cross-entropy to the original cross-entropy loss, which formed a symmetric cross entropy loss and reduced model overfitting to noisy labels. Ma et al. [129] normalized loss functions by dividing the sum of loss among all possible labels, but the resultant model tended to underfit. To address this problem, they further proposed active passive loss that combined normalized loss of two types: one only optimized on the label class (active loss) and one optimized on all classes (passive loss). Kim et al. [96, 97] proposed negative learning loss which encouraged the model not to predict incorrect label.

Secondly, samples may be weighted differently in the loss function. Natarajan et al. [147] assumed the existence of class-dependent label noise on a binary classification dataset, and they modified the original loss to a weighted surrogate loss according to manually assigned noise rates and sample labels. Wang et al. [207] calculated the gradients of the training loss with regard to the logit vector, and improved MAE by giving samples different weights according to the magnitude of gradients. Annotator robust loss [191] balanced the contribution of edge pixels and non-edge pixels by weighting each category of pixels with the number of pixels in the other category.

Finally, the noise distribution may be estimated to correct the loss function. Goldberger and Ben-Reuven (2016) viewed the correct label as a latent random variable and modeled the noise by an additional softmax layer which predicted the probability of correct hidden labels. Patrini et al. [158] combined noise rate estimation algorithms and deep neural networks, where the estimated transition matrix corrected the loss function to make it equal to the original loss computed on clean labels. Tanno et al. [194] modeled the annotation errors of each annotator with a confusion matrix, and they added a regularization term that jointly optimized the confusion matrix and the model predictions. Xia et al. [220] trained the classifier with noisy labels and initialized the label transition matrix based on their classifier's predictions, and then retrained the model with a learnable variable that automatically revised the transition matrix.

Inspired by loss functions with sample weighting schemes, we propose a method to re-weight different components in the loss function. When models are directly trained from pseudo-labels with the Charbonnier loss (defined later in Eq. 6.1), we observe a tendency for the network to overfit label noise, resulting in poor performance. To address this problem, we divide the t-f bins into two categories according to the pseudo-label: foreground where whistle signals occur in this node, and background otherwise. We add a regularization term to re-weight foreground and background t-f bins in the loss function. The modified loss encourages the model to make correct predictions under certain label noise, e.g., when the pseudo-label missed part of the whistles. However, as the pseudo-label may contain multiple types of errors, the regularization term may encourage the model to have one type of error while suppressing another type of error. For example, improving the weight of foreground t-f bins may help reduce false negative predictions, but it also increases the chance of false positive predictions. Inspired by the work of focal loss [117], we add a multiplication factor that adjusts the weight according to the prediction recall or precision in each training sample.

Our contributions are threefold. Firstly, in order to eliminate the need for manual annotations, we explore two automated whistle extraction methods to automatically generate training datasets for CNN-based whistles detector. Secondly, we show that the method presented in [109] leads to inferior performance when trained with pseudo-labels, and we present loss functions that enable the model to make robust predictions. Finally, we conduct extensive experiments to evaluate our fully automated whistle detector. On a four-species benchmark dataset, a whistle detector trained from pseudo-labels without any human annotation correctly extracted 83.33% of whistles with a precision of 89.55% (F1-score 86.31%), which is close to a model trained with 12,539 expert-annotated whistles. We also show that the method is effective with pseudo-labels generated by other algorithms, and present results with similar performance (correct extraction of 85.78% of whistles with a precision of 88.78%, F1-score 87.2%) for an algorithm that required a small number (185) of analyst annotated whistles.

6.2 Methods

6.2.1 Dataset

We used the acoustic data from the Detection, Classification, Localization, and Density Estimation (DCLDE) workshop [32] for model training and evaluation. This dataset consists of 393 recordings collected for five species of odontocetes: bottlenose dolphins (*Tursiops* *truncatus*), long- and short-beaked common dolphins (*Delphinus capensis*, *Delphinus delphis*), (*Peponocephala electra*), melon-headed whales (*Stenella longirostris*). Two types of hydrophones were deployed, ITC 1042 (Intl. Trandsucer Corp., Santa Barabara, CA) and HS 150 (Sonar Research and Development Ltd., Beverly, UK) hydrophones, for collecting the data. The hydrophones were towed by the R/V David Starr Jordan, mounted to the stationary platform R/P FLIP (Fisher and Spiess 1963), and deployed from small boats. The deployment depths of the hydrophones were 10 to 30 meters. The acoustic signals were sampled at 192 kHz with 16 or 24 bit quantization.

Data preparation We transformed the acoustic data into log-magnitude spectrograms before using them to generate pseudo-labels or as input to a trained whistle extraction model. Discrete Fourier transforms (DFT) were performed on 8 ms Hamming-windowed frames (125 Hz bandwidth) every 2 ms. We empirically restricted the log10-magnitude spectrogram to the range [0, 6] , clamping values to a range of 0 to 6. This corresponds to an uncalibrated intensity range of 0 to 120 dB, which was then normalized to the range [0, 1]. We limited the spectrogram to the frequency range of 5-50 kHz (361 frequency bins) that covered most delphinid whistles and their harmonics. The spectrograms were divided into 3-second long non-overlapping segments for model training and evaluation. The spectrogram segments from training datasets were further divided into patches of size 64 (128 ms) x 64 (8 kHz) before model training.

Training dataset We used two non-overlapping subsets of the DCLDE data for model training. Firstly, we used one subset for our supervised training experiments. Analysts provided detailed t-f annotations of whistles for 45 recordings. Among these annotated recordings, we chose 30 recordings that were not used for evaluation in Roch et al. [171] as our "labeled dataset". These audio files recorded 127 minutes of odontocete calls and included 12,539 annotated whistles. Secondly, we used the acoustic data without analyst annotation in our pseudo-label experiments. These data consist of 348 recordings, and the total duration is around 29 hours. This set of data is referred to as our “unlabeled dataset”.

Evaluation dataset We used a subset of annotated acoustic data from the DCLDE workshop 2011 for evaluation. This subset consists of 12 audio files for bottlenose dolphins, long-beaked common dolphins, melon-headed whales, and spinner dolphins. The total duration of those recordings was around 43 minutes, and the t-f coordinates of 6,011 whistles were annotated by analysts. All these files were used for evaluation in the work of (Roch et al., 2011). We did not use the recordings of short-beaked common dolphin (*Delphinus delphis*) due to some annotation errors. The details of the audio files are summarized in Table. 6.1, along with the number of annotated whistles per species that we expected to retrieve. Criteria for which whistles were expected to be retrieved is detailed in in the Section 6.2.5.

Table 6.1: Summary of the number of whistles per species in the evaluation dataset and the specific DCLDE 2011 audio files used.

Species	Whistles	Files
Bottlenose dolphin (*Tursiops truncatus*)	354	Qx-Tt-SCI0608-N1-060814-121518 palmyra092007FS192-070924-205305 palmyra092007FS192-070924-205730
Long-beaked common dolphin (*Delphinus capensis*)	557	Qx-Dc-CC0411-TAT11-CH2-041114-154040-s Qx-Dc-CC0411-TAT11-CH2-041114-154040-s QX-Dc-FLIP0610-VLA-061015-165000
Melon-headed whale (*Peponocephala electra*)	338	QX-Dc-FLIP0610-VLA-061015-165000 palmyra092007FS192-071004-032342 palmyra102006-061020-204327_4
Spinner dolphin (*Stenella longirostris*)	686	palmyra092007FS192-070927-224737 palmyra092007FS192-070927-224737 palmyra102006-061103-213127_4

6.2.2 Pseudo-label Generation

We use the spectral peak detection and graph search algorithm implemented in *Silbido*[1] [171] to extract whistles from the unlabeled dataset. The spectral peak detection algorithm smooths the spectrograms with a median filter over each 3 × 3 time-frequency grid. Then it subtracts the mean value over a 3-second window in each frequency bin. If one t-f bin has a signal-to-noise ratio (SNR) larger than 10 dB and no other bins within +/- 250 Hz have a larger magnitude than this t-f bin, it is considered a spectral peak. Next, the graph search algorithm manages the candidate detections with sets of graphs. Each graph depicts one or more candidate whistle contours where a sequence of spectral peaks is connected. Each spectral peak either starts a new graph or is added to existing graphs. Peaks are added to existing graphs if they are a good fit to adaptive polynomial predictions of graph trajectories, and otherwise used to seed new graphs. Polynomial order is driven by goodness of fit as measured by the adjusted R2 coefficient [43], and spectral peaks are merged into an existing graph when they are within 50 ms of the last endpoint in the graph and 1000 Hz of the fitted polynomial curve. Graph state is maintained across 3 s blocks, permitting graphs to represent spectral peaks from whistles that cross processing blocks. Once a graph is no longer eligible to incorporate additional spectral peaks, whistles are extracted from the graph. When interior nodes have more than a pair of edges, the rate of change on both sides are examined to determine if multiple whistles crossed the interior node. We remove detected whistles that are shorter than 150 ms as per Roch et al. [171].

To examine our method's sensitivity to the pseudo-label generation algorithm, we used Gruden and White's [60] Sequential Monte Carlo Probability Hypothesis Density (SMC-PHD) whistle extractor[2] to generate the second set of pseudo-labels. Briefly, the algorithm uses

[1]We used beta2 version of *Silbido* at `https://roch.sdsu.edu/index.php/software/`. The latest version of *Silbido* is availiable at `https://github.com/MarineBioAcousticsRC/silbido`.

[2]The preprocessing code is available at `https://doi.org/10.5258/SOTON/D0316`. The SMC-PHD code is available at `https://github.com/PinaGruden/SMC-PHD_whistle_contour_tracking`.

computationally tractable approximation of the multi-target Bayes filter to track whistle contours based on spectral peaks from pre-processed spectrograms. Pre-processing of spectrograms is based on established methods [54, 59] in order to reduce noise and interfering signals. If t-f bins have magnitudes larger than 8 dB on the normalized spectrogram and are within the frequency range of 2-50 kHz, they are considered spectral peaks. These peaks are used as measurements for the SMC-PHD algorithm to track whistles. The SMC-PHD filter is a recursive filter that propagates the first-order moment of the multi-target posterior (called PHD) in time through prediction and update steps. The PHD function at each time step is approximated by a cloud of weighted particles. Particle locations and weights are predicted and updated according to the sequential Monte Carlo principles and PHD equations, respectively. The SMC-PHD implementation of Gruden & White [60] used in this work employs a trained radial basis function (RBF) network to estimate the particle locations in the prediction step. The training data consists of 3 min of recording and 185 annotated whistles, and these data are not included in our training or evaluation dataset. New whistles are introduced to the filter through a birth model that incorporates measurements and priors based on a training data. Additionally, the filter incorporates false alarms and missed detections in the problem formulation. At each time step, whistle states (representing whistle contour peaks) are estimated, and their identity tracked based on labeled particles as outlined in Gruden & White [60].

Irrespective of the whistle extraction algorithm, we generate bin-wise pseudo-labels for each 3-second spectrogram segment. The pseudo-label is initialized as a zero matrix of the same size as the spectrogram segment. We draw the whistle contour on the matrix with the cv2.polylines() method in the Python OpenCV library [13]. The thickness of the polyline is empirically set to 2. The pseudo-labels have element values normalized to values between 0 (background) and 1 (whistle), respectively. Similar to the training spectrograms described in previous subsection of data preparation, the pseudo-labels are divided into 64$\times$64 patches that match the spectrogram patches for model training. If the pseudo-label marks at least one t-f bin in the patch as containing whistle energies, we consider this patch as a "positive patch". Otherwise, the patch is considered a "negative patch". As there are many more negative patches than positive patches, we balance the training dataset by randomly selecting the same number of negative patches as positive patches for model training.

6.2.3 CNN-based Whistle Extraction

We use the Deep Whistle Contour (DWC) detector[3] implemented by Li et al. [109] as our model for whistle extraction (Fig. 6.1). Firstly, a CNN model, the Whistle Extraction Network, takes a spectrogram as input and predicts a confidence map of the same size as the input spectrogram. The confidence within each t-f bin indicates the probability that this bin contains whistles. The confidence map is used to predict peaks. Confidence map t-f bins are labeled as peaks when the probability of attribution to whistle energy is larger than 0.5 and the bin contains a local maximum along the frequency axis. Whistle contours are produced from the set of peaks using a modified version of the graph search method summarized in previous subsection of pseudo-label generation.

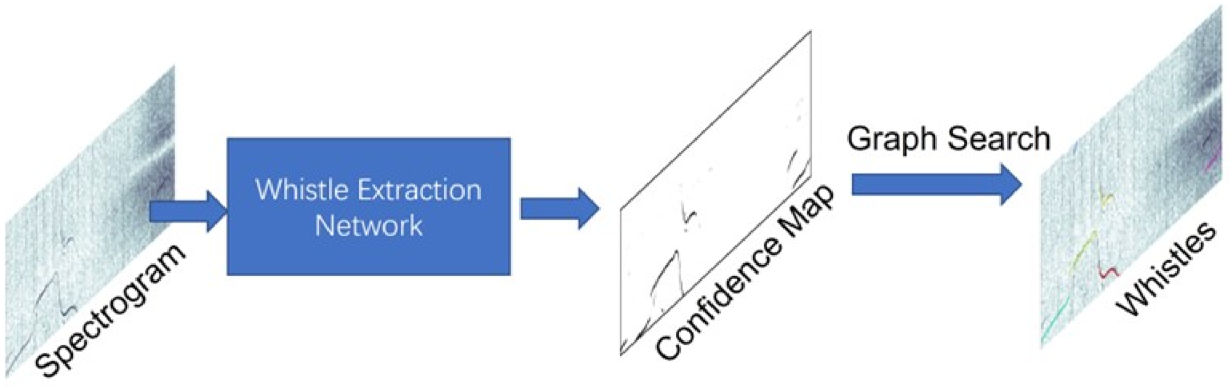


Figure 6.1: Illustration of the whistle extraction algorithm from Li et al.[109]. The neural network identifies whistle energy from input spectrograms and is processed by a subsequent algorithm to extract whistle annotations.

Complete details of the network may be found in [109], but to summarize briefly, the network

[3]Code is available at `https://github.com/Paul-LiPu/DeepWhistle`.

consists of 10 convolutional layers. The inner 8 convolutions consist of 4 residual network blocks [71] each followed by batch normalization [78] with a parametric rectified linear unit activation function [72]. Denoting y and $\hat{y}$ as the vectorized label and CNN output, respectively, the training loss function is

$$\mathrm{L}_{\mathrm{base}}\left(\hat{y}, y\right) = \sqrt{||\mathrm{y} - \hat{y}||_2^2 + \varepsilon^2} \tag{6.1}$$

where ε is a small constant (1×10^{-3}). This baseline loss function encourages the CNN model to predict the same confidence as the label. We train the model for 1 million iterations. The learning rate is initially 0.001 and multiplied by 0.1 every 400K iterations. The other training hyperparameters and graph search parameters are the same as the implementation of Li et al. [109].

6.2.4 Pseudo-label Learning

Let us consider the errors in the pseudo-labels. Fig. 6.2 shows two typical examples of whistles extracted by graph search. The extracted whistles typically have high bin-wise precision but low bin-wise recall, i.e., the extracted contours mostly cover t-f bins that have whistles but there are a significant number of whistle t-f bins missed.

We use a synthetic toy example (Fig. 6.3) to illustrate the impact of whistles that are missed by the pseudo-label generator. In this case, the true label contains two whistles and the pseudo-label generator missed one of them. As these whistles have similar appearance, our whistle extraction model may tend to make the same prediction for both whistles. Therefore, there are two likely predictions: the CNN model recognizes both whistles (prediction 1) or misses all of them (prediction 2). These two predictions have the same loss value underL_{base}, which means that the model may choose either one of the predictions during training. In order to encourage the model to have the correct prediction, we modify the loss function in

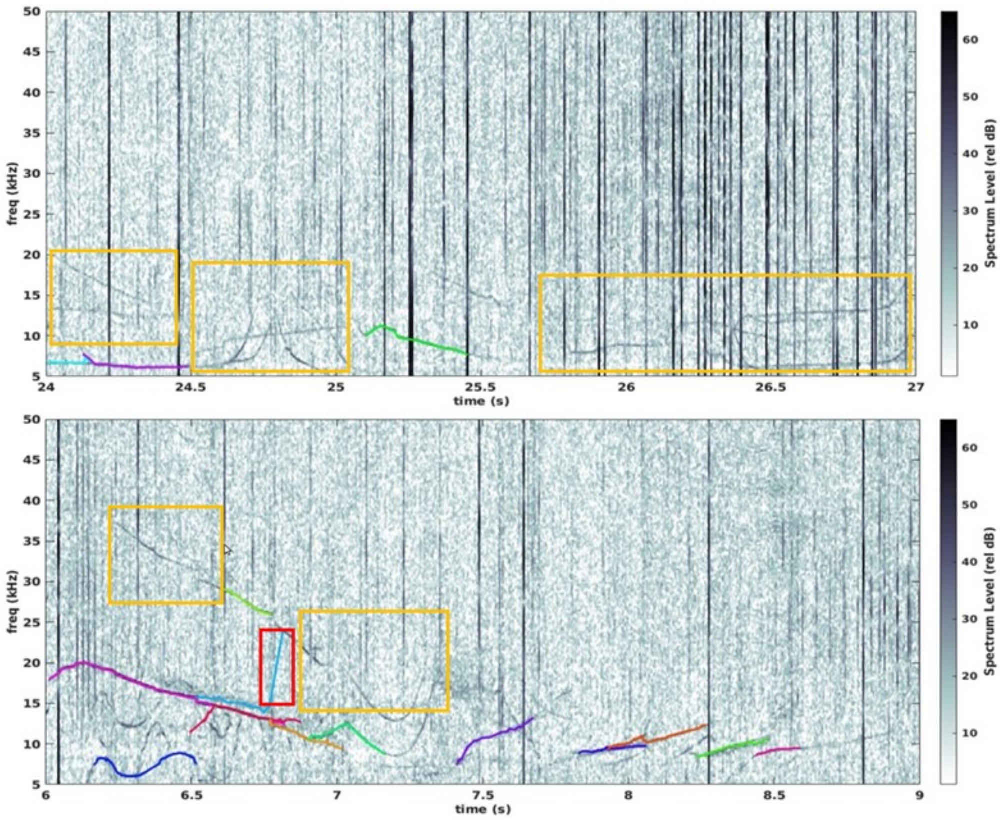


Figure 6.2: Two examples of the whistles detected by graph search [171]. The extracted whistles are shown as colored polylines. The contrast of the spectrogram is improved for better visualization. We highlight examples of the missed whistles and false positive detections with orange and red bounding boxes, respectively.

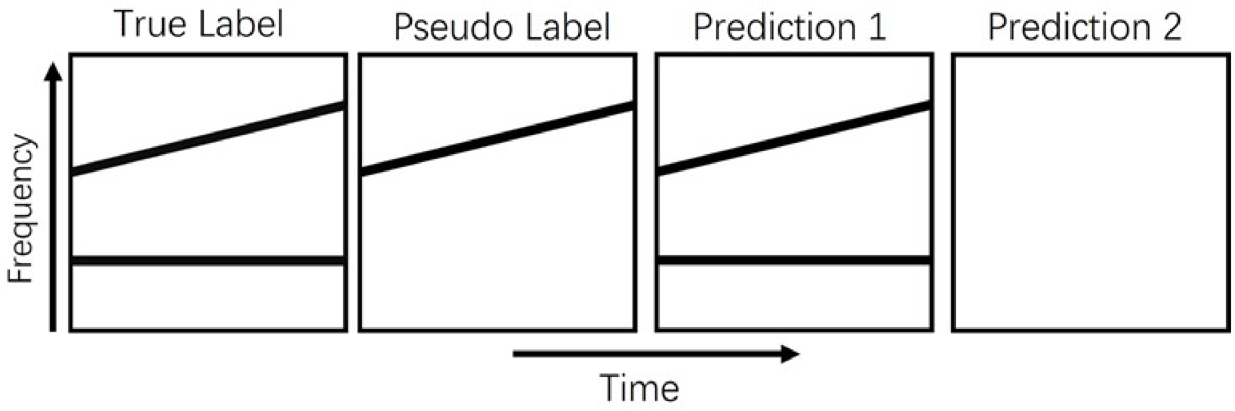


Figure 6.3: A toy example for the case when pseudo-labels do not include some of whistles in ground truth label.

Eq. 6.1 by adding a regularization term:

$$L_{recall}(\hat{y},\ y) = L_{base}(\hat{y},\ y)\ + \lambda\sqrt{||(\hat{y}-y)y||_2^2 + \varepsilon^2} \tag{6.2}$$

where $\lambda \in \mathbb{R}^+$ is a constant number and ε is 1×10^{-3}. By modifying the training objective, we increase the penalty for the model missing t-f bins that pseudo-labels have marked as containing whistle energy. When $\lambda > 0$, prediction 1 has a lower loss than prediction 2, i.e., the model will prefer prediction 1 during training. Therefore, the modified loss function will help the model detect whistles missed in pseudo-label and increase prediction recall.

However, the above conclusion may not apply to the case where pseudo-labels incorrectly predict whistles in areas of background noise or confounding signals (Fig. 6.4). Similar to the previous case, inaccurate pseudo-labels are driving the model to associate examples of whistles and background noise/confounding signals as the same category. This can result in the model predicting noise or confounding signals as whistles (prediction 1) or other training examples may result in the model correctly recognizing the inaccurate pseudo-label as background (prediction 2). L$_{recall}$ (Eq. 6.2) will make the model prefer prediction 1 instead of prediction 2, which leads to an increased false positive rate. To mitigate this problem, we modify the loss function to be:

$$L_{recall}(\hat{y},\ y) = L_{base}(\hat{y},\ y)\ + \lambda(1 - R_{soft}(\hat{y},\ y))^{\gamma}\sqrt{||(\hat{y}-y)y||_2^2 + \varepsilon^2} \tag{6.3}$$

where $\lambda \in \mathbb{R}^+$ and $\gamma \in \mathbb{N}^+$ are constant parameters. $R_{soft}\left(\hat{y}, y\right)$ is the soft recall of prediction $\hat{y}$ to pseudo-label y:

$$R_{soft}\left(\hat{y}, y\right) = \frac{||y\hat{y}||_1}{||y||_1 + \epsilon^2} \tag{6.4}$$

which measures the rate at which t-f bins marked as whistles in the pseudo-label are detected. ϵ is a small positive constant (10^{-5}). The $\left(1 - R_{soft}\left(\hat{y}, y\right)\right)^{\gamma}$ term increases the penalty $\lambda\left|\left|\left(\hat{y}-y\right)y\ \right|\right|_2$ when t-f nodes pseudo-labeled as whistles are predicted with low confidence. Compared to the first proposed penalized loss (Eq. 6.1), the updated regularization term leads to lower penalties when the model predicts higher scores on t-f nodes with whistle pseudo-labels. For example, if we have γ=1, the penalty weights become 0 and 0.5λ for prediction 1 and 2 in Fig. 6.4, respectively. At the same time, the penalty weights are 0 and λ for prediction 1 and 2 in Fig. 3, respectively. While we have the same penalty as Eq. 6.1 for the wrong prediction (prediction 2) in Fig. 6.3, we reduce the penalty for the model to have the correct prediction (prediction 2) in Fig. 6.4. Therefore, the model is less encouraged to recognize background noise as whistles with Eq. 6.3. The parameter γ may adjust the influence of recall. We note that Eq. 6.3 is a generalization of Eq. 6.1 and Eq. 6.2. If $\gamma = 0$, Eq. 6.3 is the same as Eq. 6.2. If γ is infinitely large and the recall is below 1, Eq. 6.3 is the same as Eq. 6.1.

Eq. 6.3 helps model training when the pseudo-label contains more false negatives than false positive detections. When the opposite is true and false positives are more prevalent in in

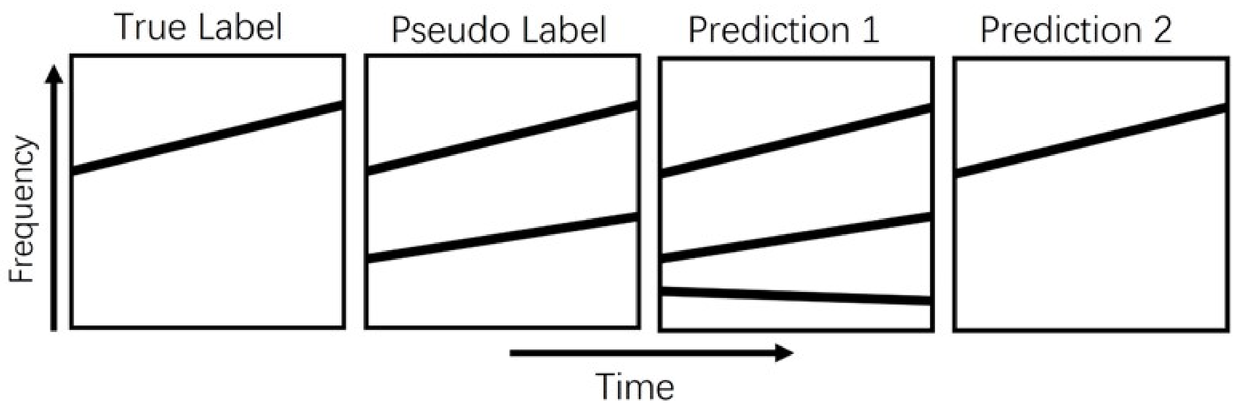


Figure 6.4: A toy example for the case when we have false positive detections in pseudo-labels.

the pseudo-labels, we can revise the training objective to address this:

$$\mathrm{L}_{\mathrm{prec}}\left(\hat{y}, y\right) = ||\mathrm{y} - \hat{y}||_2 + \lambda(1 - P_{soft}\left(\hat{y}, y\right))^{\gamma} \sqrt{||(\hat{y} - y)\left(1 - y\right)||_2^2 + \varepsilon^2} \tag{6.5}$$

where $\lambda \in \mathbb{R}^+$ and $\gamma \in \mathbb{N}^+$ are constant parameters, and $P_{soft}\left(\hat{y}, y\right)$ is the soft precision of prediction $\hat{y}$ to pseudo-label y:

$$P_{soft}\left(\hat{y}, y\right) = \frac{||y\hat{y}||_1}{||\hat{y}||_1 + \epsilon} \tag{6.6}$$

where ϵ is a small positive constant number (10^{-5}). L_{prec} will encourage the model to generate prediction 2 in the case of Fig. 6.4, reducing false positives. In pseudo-label training, the effectiveness of L_{recall} or L_{prec} is likely to depend on the pseudo-label noise statistics. If the pseudo-labels include more false positives, L_{prec} will likely be the better choice. Conversely, if the pseudo-labels include more false negatives, L_{recall} should be chosen for model training.

6.2.5 Metrics

We evaluate the model performance using the evaluation code in *Silbido*. The evaluation starts with a matching process between detections and ground truth labels. If detected and annotated whistles overlap in time, and the mean frequency difference is less than 350 Hz, this pair of detected and annotated whistles are considered a match. As in [171, 109], we only consider the analyst-annotated whistles with a duration $\geq$150 ms and a signal-to-noise ratio (SNR) $\geq$ 10 dB over at least a third of the whistle. Any annotated whistles that did not meet these criteria were omitted from the analysis. Detections that matched discarded ground truth annotations were neither counted towards nor against performance.

After matching, *Silbido* provides precision, the percentage of correctly detected ground-truth whistles, recall, the percentage of ground-truth whistles missed, deviation, the average frequency deviation of the detected whistle to matched annotation, coverage, the percentage of detected whistle signals being covered by detections, and fragmentation, the average number of detections matched to the same ground truth whistle. We calculate the F1-score, the harmonic mean of precision and recall, as an overall metric of the extraction performance. We evaluate our model on each species independently and report the performance averaged on different species.

6.3 Results

6.3.1 Pseudo-label Generated by Graph Search

We designed a series of experiments to validate our proposed methods. In a first step, we extracted whistles with spectral peaks detection and graph search method in *Silbido*, and this experiment is referred to as "graph search." Next, we trained two models with the L_{base} loss function (Eq. 6.1). The first of these models used the analyst annotations and is referred to as "$\mathrm{L}_{\mathrm{base}}$ annotation." The second model, "$\mathrm{L}_{\mathrm{base}}$ graph," used the same loss function trained with *Silbido* graph search generated labels from the larger unlabeled dataset.

To examine the effectiveness of the proposed regularization penalties L_{recall} or L_{prec} on the unlabeled dataset, we trained additional models on the unannotated data using graph search labels. Experiments using these models are denoted "$\mathbf{L}_{recall}$ graph" and "$\mathbf{L}_{prec}$ graph". Various values of λ and γ were empirically explored to find the optimal parameter setting on our dataset. Specifically, we use values of 0, 1, 2, or 4 for the exponent parameter γ in L_{recall} and L_{prec}. For each fixed value of γ, we explored varied λ values until we find a peak F1-score. In L_{recall} experiments, we used $\lambda \in \{0.5, 1, 2, 3, 4\}$, $\{2, 4, 6, 8\}$, $\{4, 6, 8, 10\}$, $\{4, 6, 8, 10, 15, 20, 25, 30\}$ for γ =0, 1, 2, and 4 , respectively. We explored larger values of λ when γ is larger in L_{recall} because larger γ led to lower weight for the regularization term. And we used for all values of γ in L_{prec}. We used the same set of λ in L_{prec}, $\lambda \in \{0.01, 0.1\}$, since we observed that larger λ resulted in lower F1-score and experiments with $\lambda = 0$ had the best F1-score.

We present the precision-recall performance in Fig. 6.5, where each point reports the performance of one of the above models. Points joined into a curve show the results for a fixed γ with variation of λ as specified above. By applying graph search on spectral peaks, "graph search" detects 72.28% (recall) of the analyst annotated whistles with a precision of 81.13%. The network model trained using analyst annotations and the baseline loss function, "$\mathbf{L}_{base}$ annotation," results in a whistle extraction recall of 85.93% and a precision of 89.50%. Replacing the analyst-annotation training data with graph search generated pseudo-labels, "$\mathbf{L}_{base}$ graph" extracts whistles with a recall of 61.53% and a precision of 94.15%. The comparison between "$\mathbf{L}_{base}$ graph" and "graph search" shows that the model is fairly accurate in its detections, but it misses many whistles that would have been detected by the algorithm that was used to generate the pseudo-labels.

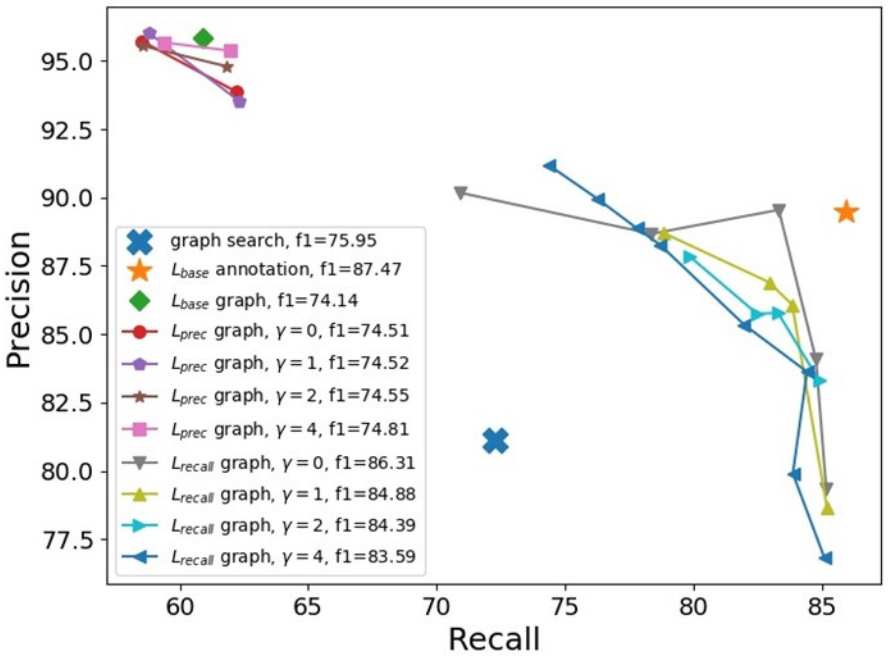


Figure 6.5: Summary of model performances derived from our graph search experiments. Each color curve shows the system performance when models are trained with L_{recall} or L_{prec} under a fixed γ and varied λ. The best F1-score among the experiments in each curve is shown in the legend. Smaller values of λ result in lower recall for L_{recall} curves, and lower precision for L_{prec}.

For models trained with L_{recall}, we observe a significant increase in recall ($> 21\%$ in the best case) compared to "L_{base} graph" while still achieving a reasonable precision (89.6%). For models trained with pseudo-labels and L_{prec} loss, we find that precision is slightly increased compared to the unaltered loss function L_{base} when we apply a larger value λ (0.01), but that there is a slight decrease in recall. In comparison to graph search, precision is greatly increased at the cost of significant loss in recall. These comparisons show that the new loss metrics, L_{recall} and L_{prec}, can increase recall or precision with respect to the performance of the algorithm that generated the pseudo-labels. Models trained with L_{prec} lead to comparable F1-scores with "L_{base} graph" and L_{recall} result in a significant F1-score increase (12.17%). The "L_{recall} graph" produce results that are similar to the performance on human-analyst training data, and the F1-score of 86.31% (λ=2, γ=0) approaches the "L_{base} annotation"

F1-score of 87.47%.

6.3.2 Pseudo-label Generated by SMC-PHD

To further validate our proposed method, we substituted an alternative whistle extraction method to generate a different set of pseudo-labels. We used SMC-PHD [60] to extract whistles, and the extraction result was referred to as "SMC-PHD". We used the radial basis function motion model that requires a modest amount of analyst-annotated training data, with Gruden and White using a small training set of 185 whistles from several minutes of annotated data that do not overlap with our test data. Consequently, this method is not entirely free of analyst annotations. As our method and graph search discarded detections that were shorter than 150 ms, we created a second set of pseudo-labels where only detections that were at least 150 ms were retained: "SMC-PHD$\geq$150ms."

We trained the whistle extraction model with the L_{base} loss function on these two sets of pseudo-labels, which are hereafter referred to as "$\mathbf{L}_{\boldsymbol{base}}$ SMC-PHD" and "$\mathbf{L}_{\boldsymbol{base}}$ SMC-PHD$\geq$150ms", respectively. As SMC-PHD exhibits the same characteristics as the graph search of tending to produce more false negatives than false positives, we trained models using the L_{recall} loss function and varied the values of γ and λ on these two sets of pseudo-labels, which were referred to as "$\mathbf{L}_{recall}$ SMC-PHD" and "$\mathbf{L}_{recall}$ SMC-PHD$\geq$150ms", respectively. We use γ=0,1 for $\mathbf{L}_{recall}$. Specifically, we used $\lambda \in \{1, 2, 3, 4, 5, 6\}$ for $\gamma = 0$ and $\lambda \in \{2, 4, 6, 8, 10, 12, 11, 12, 14, 16\}$ for $\gamma = 1$ for "$\mathbf{L}_{recall}$ SMC-PHD$\geq$150ms". For the experiment that did not discard short detections, "$\mathbf{L}_{recall}$ SMC-PHD," we used $\lambda \in \{1, 2, 3, 4\}$ for $\gamma = 0$ and $\lambda \in \{2, 3, 4, 5, 6, 8, 10\}$ for $\gamma = 1$.. We did not execute experiments using the $\mathbf{L}_{prec}$ loss function as SMC-PHD exhibits similar patterns of error in the pseudo-labels.

The precision-recall performance is shown in Fig. 6.6. As a competitive baseline, SMC-PHD detects 92.45% (recall) of annotated whistles with a precision of 76.55%. After removing the detections that are shorter than 150 ms, SMC-PHD achieves a precision of 95.85% while the recall drops to 60.88%. This indicates that SMC-PHD tends to extract more shorter whistle contours than graph search. The longer detections of SMC-PHD are more likely to be correct detections. Adjusting the time threshold for SMC-PHD might achieve a better F1-score on our evaluation datasets, but this experiment is beyond the scope of this paper.

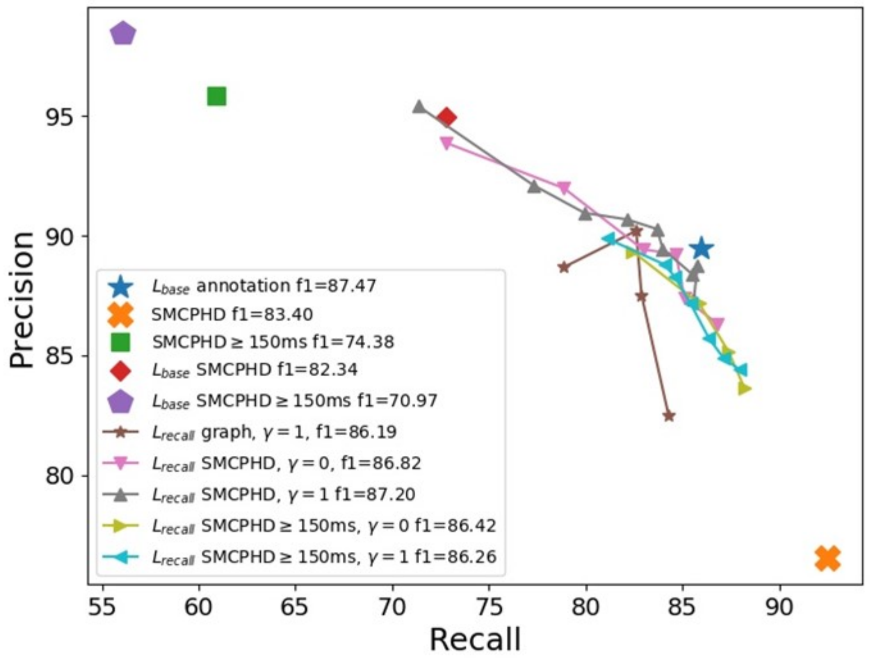


Figure 6.6: Summary of model performances derived from our experiments using SMC-PHD generated pseudo-labels. The color curves show the system performance when models are trained with $\mathbf{L}_{recall}$ under a fixed γ and varied λ. The best F1-score among the experiments in one curve is shown in the legend. For comparison, a curve corresponding to graph-search generated labels (L_{recall} graph) is also shown.

When we trained the model with $\mathbf{L}_{\boldsymbol{base}}$, we received an F1-score of 82.34% using all SMC-PHD detections and 70.97% after removing shorter detections. These F1 scores were lower than "SMC-PHD" and "SMC-PHD≥150ms", respectively. When we applied L_{recall} for model training, we observed a maximum F1-score of 87.2% in the experiment of "$\mathbf{L}_{\boldsymbol{recall}}$ SMC-PHD≥150ms" when we used $\lambda = 14$ and $\gamma = 1$, which was a significant increase compared to "$\mathbf{L}_{\boldsymbol{base}}$ SMC-PHD≥150ms".

6.3.3 Summary of Whistle Extraction Performance

We summarize the performance of the experiments having the best F1-score under different settings in Table. 6.2. Our modified loss function (Eq. 6.3) leads to an F1-score of 87.2% with L_{recall} SMC-PHD≥150 ms, which is almost identical to the model trained with a large human annotated dataset (F1-score of 87.47%). Additionally, we observe a significant improvement in coverage (more than 5.5 %) when we train the model with L_{recall} compared toL_{base}+Graph. We also have fewer fragments in our L_{recall} experiments. Combining these observations, our model trained with L_{recall} can correctly predict more t-f bins as whistles. Finally, although we observe a higher mean frequency deviation in pseudo-label experiments compared to graph search and SMC-PHD, the increase in deviation is less than one frequency bin width (125 Hz) on our spectrogram.

Table 6.2: Summary of the performance. Summary of the performance. Scores indicating the harmonic mean (F1) of precision and recall, the mean deviation in frequency from analyst annotations (μ_σ), the percentage of each whistle that was detected (coverage), and the mean number of connected segments for each whistle (fragmentation).

Method	F1	Precision	Recall	μ_σ Hz	Coverage	Fragmentation
L_{base}+ Annotation	87.47	89.50	85.93	92.00	88.08	1.13
Graph Search	75.95	81.13	72.28	101.00	81.05	1.23
SMC-PHD	83.40	76.55	92.45	108.00	70.93	1.80
SMC-PHD≥150ms	74.38	95.85	60.88	103.50	72.88	1.23
L_{base}+ Graph Search	74.14	94.15	61.53	144.75	77.50	1.18
L_{base}+ SMC-PHD	82.34	94.98	72.75	122.75	81.75	1.18
L_{base}+ SMC-PHD≥150ms	70.97	98.45	56.08	119.75	73.70	1.20
L_{recall}+ Graph Search	86.31	89.55	83.33	154.25	86.73	1.18
L_{recall}+ SMC-PHD	86.42	87.18	85.85	134.25	87.30	1.15
L_{recall}+ SMC-PHD≥150ms	87.20	88.78	85.78	134.75	87.30	1.18

6.3.4 Visualization of Model Output and Whistle Extraction Result

We show examples of network output and extracted whistles produced by our algorithms in Fig. 6.7 and Fig. 6.8. We compare the network outputs (confidence maps) of model trained with L_{recall} and L_{base} in Fig. 6.7. The model trained with L_{recall} had higher response to whistle energy and produced more continuous coverage of whistles compared to the model trained with L_{base}. When the confidence map predictions are processed by Silbido's graph search algorithm (and likely many other whistle extraction algorithms), this results in more and longer extracted whistles (Fig. 6.8).

6.4 Discussion

Our results show that it is feasible to train competitive deep-learning-based models for whistle extraction without using analyst annotations as training data. With pseudo-labels generated by graph search and the proposed loss function, we are able to extract whistles with an F1-score of 86.42%, which significantly surpasses graph search (F1-score: 75.95%). With pseudo-labels generated by SMC-PHD which uses 185 annotated whistles for training, we are able to further improve the whistle extraction performance to an F1-score of 87.2%, which is close to the whistle extractor trained with 12,539 annotated whistles. By using different whistle extractors to generate pseudo-labels, the proposed method is able to eliminate or greatly reduce the human analyst annotation effort.

The L_{recall} loss showed strong F1 performance gains over the algorithms used to produce the pseudo-labels used to train the CNNs. We observe that the system improves whistle coverage and reduces fragmentation in both quantitative (Table II) and qualitive results (Fig. 6.7 and Fig. 6.8). The longer whistle detections with fewer gaps may better facilitate downstream

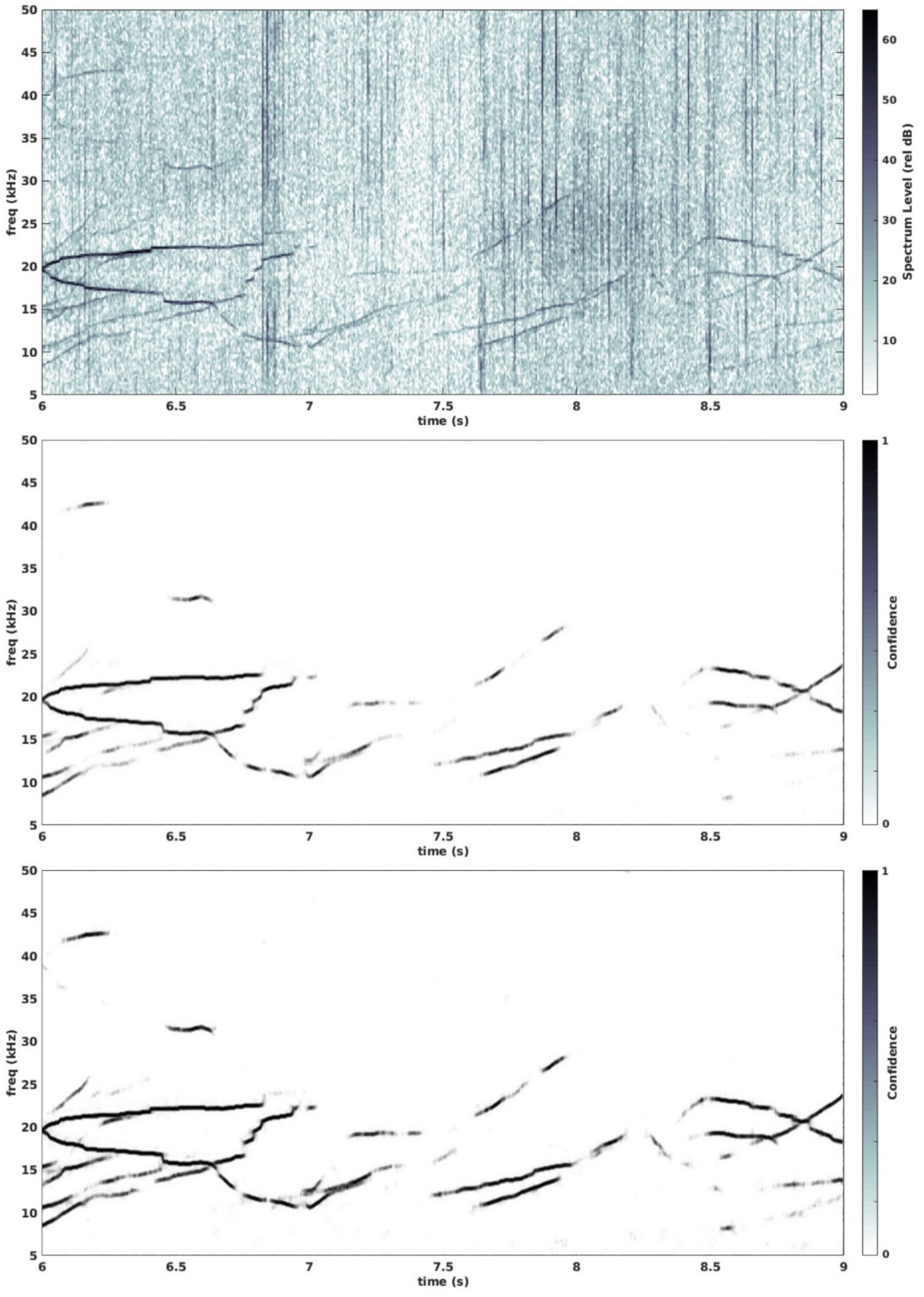


Figure 6.7: Comparison of CNN confidence map predictions from a spectrogram (Upper) using different loss functions. Middle: predictions using L_{base} loss. Lower: predictions using L_{recall} loss.

research, e.g., species identification. The L_{prec} loss, which was only tested on labels generated by algorithm that tends to produce more false negatives than false positives, provided gains

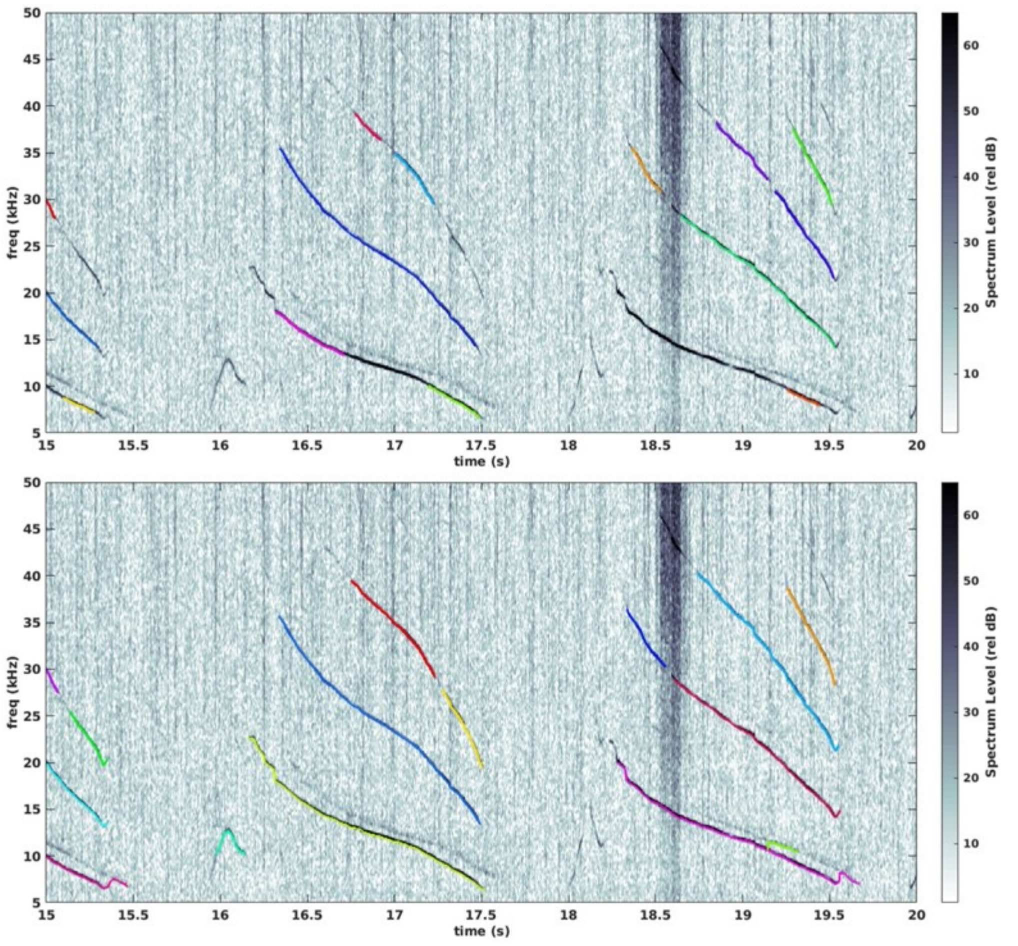


Figure 6.8: Comparison of the detected whistles among different experiments. Each whistle is colored differently. Upper: extracted whistles by CNN trained with L$_{base}$. Lower: extracted whistles by CNN trained with L$_{recall}$.

in precision at the cost of significant drops in recall on these data. We suspect that it would fare better on pseudo-label sets with higher false positive rates. In contrast, models trained using the baseline loss function, L$_{base}$, were unable to produce F1-scores that exceeded the performance of the algorithms used to produce the pseudo-labels.

There were likely more false negatives than false positives in our pseudo-labels for both label generation methods. We observe a higher precision than recall in "**graph search**" and "**SMC-PHD $\geq$150ms**". Although "**SMC-PHD"** has higher recall (92.45%) than precision (76.55%), the coverage was around 71%, suggesting that roughly 29% of t-f bins in whistles were not detected and that the t-f bin-level recall was lower. Furthermore, since we balanced the number of negative patches and positive patches in the training dataset and false negative patches only covered a small portion of negative patches, many false negatives were excluded from training. While the proposed L_{recall} and L_{prec} were effective in improving whistle extraction recall or precision, respectively, L_{recall} increased F1-score significantly more than L_{prec}. This observation also indicated that false negatives (missed whistles) in the pseudo-labels affect our whistle extraction model more than false positives.

As a result, the CNN-based whistle extraction algorithm demonstrated the ability to extract whistles with high precision when the CNN was trained with pseudo-labels. For example, the experiment of "L_{base} graph" had a precision of 94.15%, while the precision of "graph search" was only 81.13%. For the experiments with L_{recall}, though the precision is decreased compared to experiments with L_{base} graph, the extraction precision was still reasonably good (>87%). However, when the CNN model was trained with L_{base} and pseudo-labels, the recall was significantly lower than the methods generating pseudo-label while the F1-scores were similar. As the pseudo-labels frequently missed whistles or portions of whistles, cases like Fig. 6.3 frequently occured in the training data. In this situation, L_{base} could have biased the model towards treating whistle signals as background noise at a higher rate than the pseudo-labels used to train the model. Because of the low recall, L_{base} resulted in a slightly worse F1-score than the whistle extraction methods that generated pseudo-labels.

6.5 Conclusion

We have developed a convolutional deep neural network that can be trained without any analyst annotations that is able to extract whistles with a performance comparable to one trained from a rich set of analyst annotations. Instead of using the expensive and time-consuming annotations produced by analysts, we used methods that required no (graph search) or minimal training data (SMC-PHD) to extract whistle annotations used as pseudo-labels for model training. We evaluated extraction performance on a diverse four-species evaluation dataset consisting of 1,935 analyst-annotated whistles (duration ≥ 150 ms; $\geq 1/3$ of the t-f bins have an SNR ≥ 10 dB). Performance of a baseline CNN model using a standard loss function (L_{base}) produced F1 scores comparable to the performance of the algorithms used to produce the pseudo-labels. However, there was a tendency to increase precision at a non-trivial cost to recall.

The proposed loss functions significantly improve whistle extraction performance. Regularization penalties compensated for errors in pseudo-labels and prioritized recall (L_{recall}, Eq. 6.3) or precision (L_{prec}, Eq. 6.5). Our experiments demonstrated that missed whistles in pseudo-labels affect the CNN model more than the incorrectly detected whistles, and the proposed L_{recall} loss function outperformed L_{base} with an absolute F1-score increase of 12.17% (graph search pseudo-labels) and 3.8% (SMC-PHD pseudo-labels), respectively. In the best case, a model trained without any analyst annotations using SMC-PHD detections of at least 150 ms duration, detected 85.78% of the whistles with a precision of 88.78%. The F1-score (87.2%) was comparable to a model trained with 12,539 annotated whistles (87.47%), showing the potential to generate whistle extraction models with near state-of-the-art performance with little to no human annotation effort.

Chapter 7

Metropolis-Hastings Sampling for Selecting Whistle Extraction Data and Pseudo-labels

7.1 Introduction

Extracting the skeletal structure of whistles on a time-frequency spectrogram is a crucial step in analyzing tonal calls produced by marine mammals. This information can be used for a variety of biological research purposes, such as abundance estimation [84], species identification [54, 86], communication and social activities [196]. Most dolphins (family Delphinidae) produce whistles that appear as highly variable contour-shaped signals on spectrograms. Therefore, it is possible to identify and analyze these whistles in spectrogram representations of audio recordings using trained analysts or specialized algorithms [171, 54, 60].

Inspired by the success of deep neural network applications in image tasks, Li et al. [109] proposed a deep learning approach for the task of whistle extraction. Their work uses a residual network [71] to predict the candidate points of whistles on spectrograms. While the learning-based method outperforms traditional optimization-based methods [171], the model requires over 7000 human-annotated polylines for training, which can be costly and time-consuming to compile. In this work, we investigate ways to use unsupervised extraction methods to generate pseudo-labels for training the model, removing the dependency on human annotation.

Training deep models with noisy pseudo-labels is a challenging task because of the ubiquitous errors in the pseudo-labels. Figure 7.1 illustrates this by showing a spectrogram with human-labeled whistles (top row) and the extraction results using Graph-Search (bottom row). Examples of false positives and false negatives are shown in red and yellow boxes, respectively. Additionally, a true positive detection may only cover a portion of the actual whistle, leaving the remaining portion unlabeled (shown in green box).

The problem of using noisy labels in machine learning applications is a well-known challenge, and it applies to our case as well. Early attempts [105] used all pseudo-labels, and later works improved this by introducing data selection mechanisms [27], label correction algorithms [239], or training strategies that are robust to label noise [153, 219]. In this work, the training samples are patches cropped from spectrograms along with their pseudo-labels. Our approach is to select samples that better help model training.

We evaluate the quality of the pseudo-labels from three perspectives. First, we assess the correctness of the pseudo-labels by evaluating the consistency between the model output and the pseudo-labels. The model is trained using all the pseudo-labels, and pseudo-label outliers are expected to be inconsistent with the model output. Second, we take into account the complexity of the whistle signals in each training sample. Spectrogram patches with more time-frequency bins labeled as whistle energy are typically representative of longer and more complex whistles or multiple whistles, and these samples may be more useful for model training. Third, we introduce a diversity metric to select samples with distinct

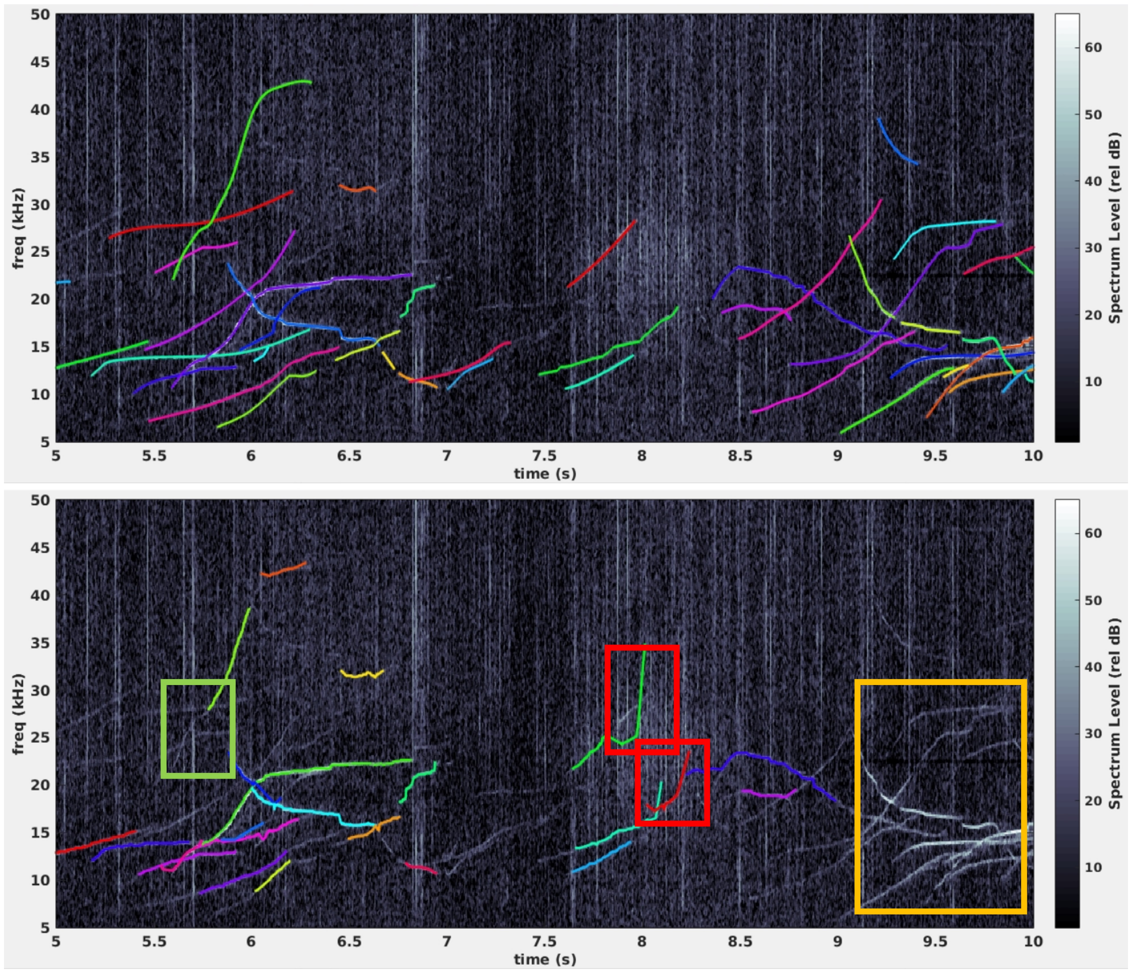


Figure 7.1: **Top**: human annotated whistles; **Bottom**: whistles detected by the Graph-search [171]. Orange box: false negatives; Red box: false positive detections; Green box: partly detected whistles.

contour shapes. We compare the Histogram of Gradient (HoG) features [38] of the pseudo-labels and use the feature distance as a measure of diversity. Finding samples with optimal diversity is usually NP-hard, and it is difficult to integrate the diversity metric with the other two metrics directly. Furthermore, the direct addition of metrics may introduce multiple weighting parameters that require additional tuning.

We introduce a probabilistic model to integrate the above three measures and cast the selection of pseudo-labels as a sampling process. We use Metropolis-Hastings sampling [26], a Markov Chain Monte-Carlo (MCMC) method, to find the optimal state of our sample selection. We start with an initial selection state and then propose replacing selected samples with others. The proposals can be accepted based on an acceptance probability defined by our three metrics: consistency, complexity, and diversity. This allows us to find a selection that balances these three factors without manual weighting parameter tuning.

We apply the above sampling method to select audio data with high-quality pseudo-labels as the seeds and expand the selection by finding samples with similar whistle signals but different background noise. This seed-and-expansion selection enables us to use contrastive learning techniques [70], such as triplet loss [21], as one seed sample and its expanded samples form a positive pair, while other samples form negative pairs.

The three main contributions of this work are: (1) The "learning via sampling" method that selects pseudo-labels using multiple measures: correctness, complexity, and diversity; (2) A sample expansion procedure with a triplet loss that effectively regularizes the learning of deep models; (3) Achieving state-of-the-art performance on a public bioacoustic dataset without the need for human annotation. Through a series of ablation experiments on a public dataset, we demonstrate that the proposed method improves whistle extraction F1-score from 80.75% to 84.72%.

7.2 Related Works

This work is closely relevant to three research streams in the areas of machine learning and bioacoustics.

Learning from pseudo-labels Pseudo-labels are widely used in various machine-learning tasks to increase the volume of training data with minimal human effort and improve model generalization capabilities [105, 227, 135]. However, pseudo-labels are often noisy and contain errors [212]. To mitigate the problem of label noise, there are two categories of approaches related to this work. The first category of approaches is to identify and select a subset of samples that have high-quality pseudo-labels. Choi et al. [27] proposed a density-based approach to cluster data and assume that samples in the clusters of high densities are likely to have correct pseudo-labels. Nishi et al. [149] calculated the loss values on augmented samples and assume that samples with lower loss values had pseudo-label of higher quality. Zhou et al. proposed to use loss dynamics and the consistency of model outputs among different augmentations for pseudo-label evaluation [243]. Another method proposed in [225] selects pseudo-labels by prediction discrepancy among a sequence of model checkpoints. In this work, we follow this line of research and develop a sampling framework to select samples with high-quality pseudo-labels.

The second category of approaches aims to leverage pseudo-labels with various designs of loss functions or network optimization processes. Castell et al. introduced the SuperLoss [19] and used a regularization term to learn the optimal weights of individual samples in the loss function. The work of [153] utilizes two types of pseudo-labels and designs a noise-aware loss function that reduces the weight of the loss on pixels where the pseudo-labels disagree with each other. Tan et al. [192] apply a contrastive loss and a structural similarity loss to reduce the impact of noisy labels. Xia et al. [219] categorize model parameters into two types, one playing a key role in learning clean labels and the other tending to fit noisy labels. This method applies different update rules on those two types of parameters to reduce the effect of noisy labels. In this work, we introduced a triplet loss function to fully explore the similarities between selected samples.

Active Learning Active learning is a machine learning paradigm that selects the most informative samples for annotation. Popular active learning algorithms select data using multiple metrics [165], including sample diversity, prediction uncertainty, and prediction consistency. For example, the work of [175] employs a core-set approach to specify dataset diversity and design a greedy algorithm to increase the diversity by increasing inter-sample distance. Yoo et al. [229] proposed to assess sample's prediction uncertainty by predicting the sample's loss value. Beluch et al. [10] used the entropy of classification probability as the measure of prediction uncertainty. Gal et al. [50] employ dropout to get multiple networks which are applied over each sample to get multiple network predictions. The consistency of these predictions are used to measure the difficulty of a sample. Following the same strategy, Belunch et al. [10] introduced an ensemble model to produce multiple predictions for the same sample. Our proposed metrics for assessing pseudo-label is inspired by previous works on active learning.

Whistle extraction Whistle extraction algorithms typically involve two steps. In the first step, the algorithms identify spectral peaks on spectrograms, which are the potential points containing the whistle signals. Spectral peaks can be determined by a Signal-to-Noise Ratio (SNR) threshold after applying denoising algorithms on the spectrogram [171, 60]. Recently, Li et al. [109] developed a deep neural network to predict the confidence of each point as a whistle candidate. In the second step, the whistle extraction algorithms connects spectral peaks into whistle contours. One stream of research [54, 171, 137, 109] tracks whistle points in the time-frequency space by line/curve fitting. Probabilistic modeling mothods, including Bayesian inference and sine wave modeling [66], Kalman filtering [132], and Monte-Carlo density filters [215, 171, 60] are also developed for the connection task.

The goal of this work is to learn the deep whistle models without any human annotations. We contribute to the literature a set of metrics for assessing the quality of pseudo-labels and a learning-via-sampling algorithm. Our method also achieves state-of-the-art performance over public dataset.

7.3 Our Method: Learning Whistle Models From Raw Data

Figure 7.2 illustrates the proposed learning via sampling method for training deep whistle models from raw data. The method begins with the application of an unsupervised detector to identify whistles in spectrograms, followed by the proposed sampling framework to select the most informative training samples with pseudo-labels. The selected samples are then expanded by adding other samples with similar whistle contour shapes but different background noises. Finally, those samples are used for model training. This labor-free training framework eliminates the need for expensive human annotation efforts and is particularly useful for the ocean bioacoustics community.

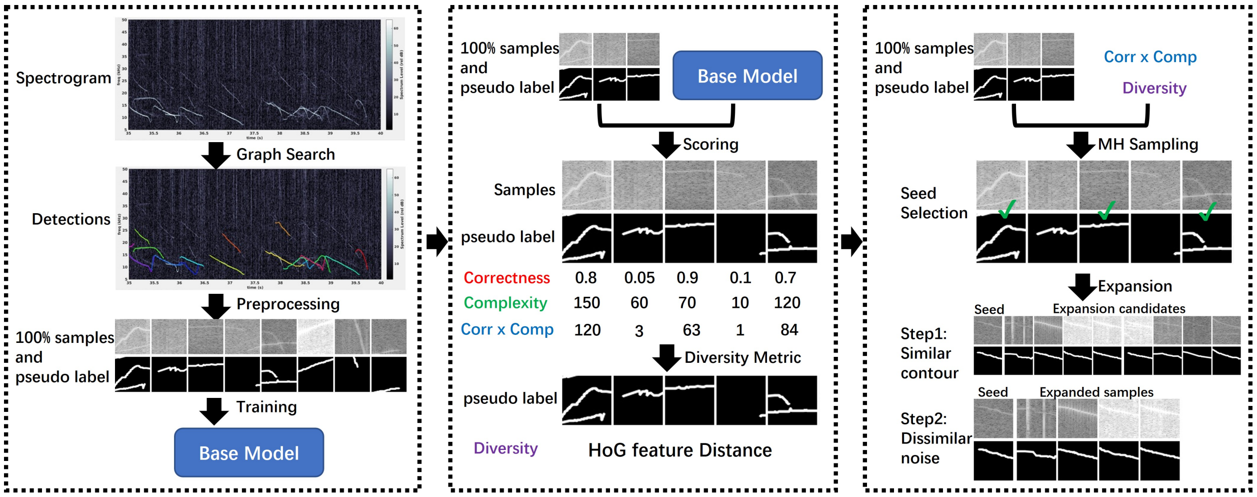


Figure 7.2: Sketch of the proposed learning-via-sampling method. The unsupervised Graph-Search algorithm [171] is used to generate whistle pseudo-labels, from which a base whistle model is trained. The labels of each sample are evaluated in terms of correctness, complexity, and diversity. A sampling algorithm is then designed to select seed samples with high-quality pseudo-labels (green check marks). Finally, we expand the seed samples to include extra pseudo-labels with similar contour shapes but different background noises. The selected samples are used for model training.

7.3.1 Notation

Let $D = \{(x_i, y_i)\}_{i=1}^{n}$ denote n training samples, where x_i is the time-frequency spectrogram patch and y_i is the pseudo-label of x_i. x_i and y_i are two-dimensional matrices that have the same size. Let $x(l)$ denote the energy value at location l on a sample x where $l \in \mathcal{Z}^2$. Label $y(l)$ is 1 when $x(l)$ contains whistle energy and 0 otherwise. We denote the whistle extraction model as f and the output of the model on sample x as f_x.

7.3.2 Quality Measures of Pseudo-labels

The proposed method starts by applying an unsupervised algorithm, Graph-Search [171], to extract whistle skeletons from time-frequency spectrograms. The algorithm generates a large number of pseudo-labels. However, many of these labels contain errors. To mitigate this problem, our method employs a set of numerical measures to evaluate the quality of the pseudo-labels. As shown in Figure 7.2, we use these measures to select a subset of high-quality pseudo-labels for training deep whistle models, which improves the overall performance of the models.

Measure I: Correctness

We first introduce a metric to examine the correctness of the labels. Since there are no human annotations available, we train a base network using all available pseudo-labels and use the output of this network to distinguish between high-quality and low-quality labels.

For a sample x, both the outputs of the base network f_x and its pseudo-label y are real-valued maps where their values are between 0 and 1. We use a soft version of IoU [199] to measure the similarity between two real-valued maps:

$$S_{corr}(x) = \frac{\sum_l f_x(l)y(l)}{\sum_l [f_x(l) + y(l) - f_x(l)y(l)]} \tag{7.1}$$

It is noteworthy that when f_x and y are binary maps, the extended IoU is equivalent to the standard IoU. A high IoU indicates that the pseudo-label matches the network output well, i.e., the network output has few false positives and few false negatives for the pseudo-label, which indicates that the pseudo-label is less likely to be an outlier.

Measure II: Complexity

We also introduce a simple yet effective metric to measure the complexity of the pseudo-labels for each sample. Generally, we prefer more complex labels because these samples carry more information than others. The number of time-frequency bins labeled as whistle energy can be an indicator of the complexity of whistle signals on one sample. We calculate the complexity measure by summing up the values of all points in the pseudo-label:

$$S_{comp}(x) = \sum_l y(l) \tag{7.2}$$

A higher value of this metric means that there are more whistle points detected by the unsupervised method, which typically indicates that there are longer and more complex whistles. We also observe that samples with higher complexity might include more false positives. However, this problem can be mitigated when we combine the complexity and correctness measures in our learning-via-sampling process.

Measure III: Diversity

Another measure encourages diversity among the selected samples. This is done to ensure that the model is exposed to various whistle contour shapes during training, which can help improve its generalization ability. In the literature, diversity is a popular standard used in active learning [175]. Given a set of selected samples, we need to measure the diversity of a candidate sample x_i based on its similarity to the selected samples. To do this, we extract the HoG features [38] of selected samples and calculate the minimal distance between x_i and other samples:

$$S_{div}(x_i) = min_{j \neq i} ||HoG(y_i) - HoG(y_j)||_2 \tag{7.3}$$

This metric is used to identify samples with different contour shapes and increase the overall diversity of selected samples.

The above three measures each characterize a different aspect of a candidate sample and should be effectively combined to optimize the selection of pseudo-labels. The first two measures operate on a single sample, whereas the third measure operates on a set of samples. We have provided examples of samples ranked using the first two measures or the combined measure in Figure 7.3. The next subsection presents a more nuanced method for combining these measures than simply adding them together [140].

7.3.3 Selecting Pseudo-labels by Sampling

We formulate the selection of pseudo-labels in a probabilistic framework and develop a Metropolis-Hastings (MH) algorithm [26] to select raw spectrograms with high-quality pseudo-labels. Our MH algorithm simulates a Markov process moving towards a target state and usually has two steps: proposing a new candidate sample and accepting or rejecting the can-

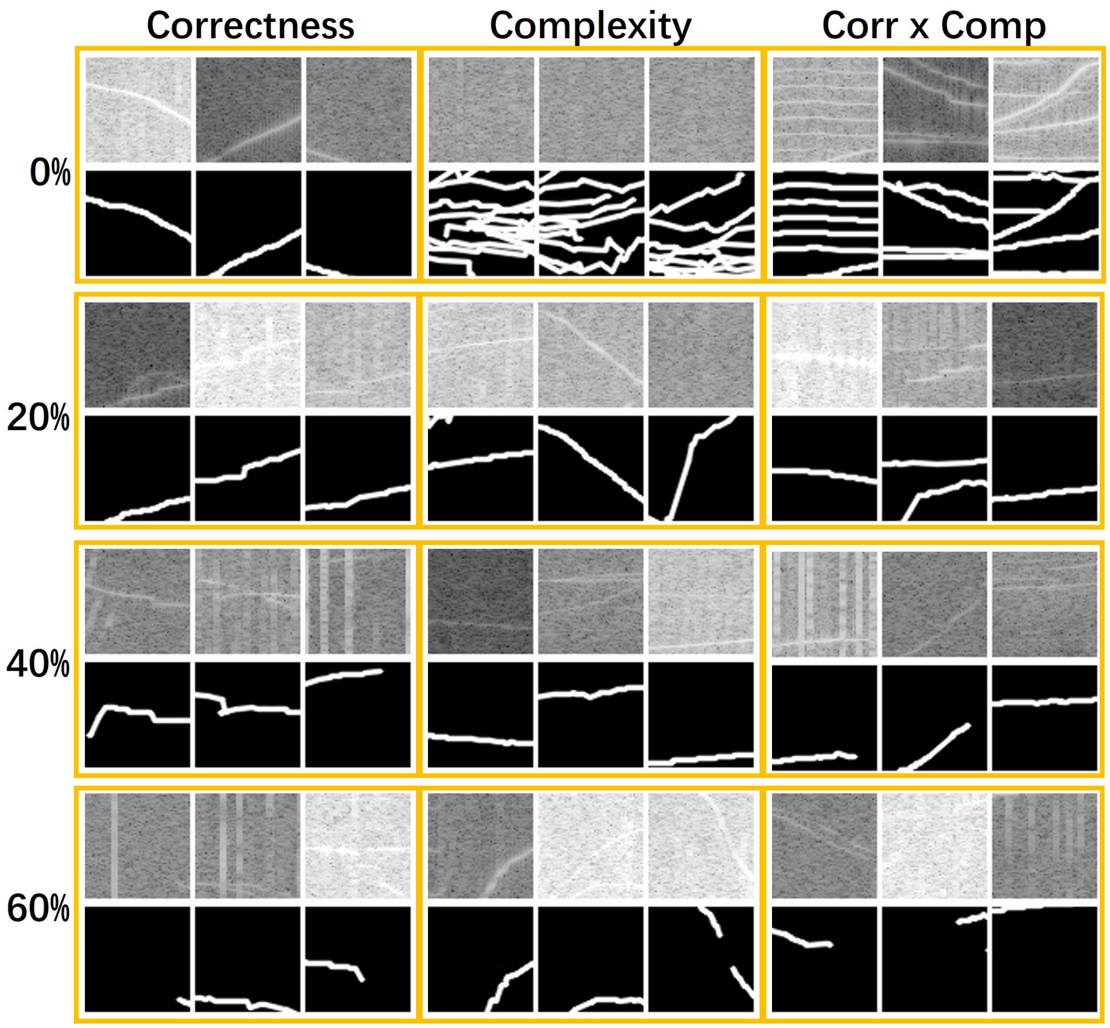


Figure 7.3: Three quality measures for sample selection. On the left, we present the rank of samples with measures shown on the top, e.g., 0% indicates that 0% samples rank higher than those samples. "Corr x Comp" is the multiplication result of correctness and complexity measures. For each pair of rank and measure, we present three spectrogram patches with their pseudo-labels in one yellow box.

didate sample based on an acceptance probability. The acceptance probability is calculated based on the three measures introduced in the previous section: correctness, complexity, and diversity. This way, samples with high-quality pseudo-labels are more likely to be selected. We repeat the MH algorithm for a fixed number of iterations to obtain a set of high-quality pseudo-labels for training the deep whistle model.

Let $W = \{w_1, w_2, ...w_i, .., w_N\}$ represent the state of the Markov process, where N is the number of raw spectrograms with pseudo-labels and $w_i \in \{0, 1\}$ indicate if a spectrogram is selected. Denote the desired number of selected samples is n, we have a constraint that $\sum_i w_i = n$. A transition of the state is caused by the status changes of one or multiple state variables. Let W' denote a new state. Let $P(W|X)$ denote the posterior probability of W where X represents the set of raw spectrograms. The Metropolis acceptance ratio is calculated as:

$$A(W,\ W') = min(1,\ \frac{P(W'|X)g(W \mid W')}{P(W|X)g(W' \mid W)}) \tag{7.4}$$

where $g(W'|W)$ indicates the proposal probability of W' given the current state W. For each state transition proposal, we generate a uniform random number $u \in [0, 1]$ and the proposal is accepted if $u < A(W,\ W')$.

The posterior distribution $P(W|X)$ is defined over the three quality measures of pseudo-labels: correctness, complexity, and diversity.

$$P(W|X) \propto \prod_i p(-S_{corr}(x_i) \cdot S_{comp}(x_i)) \cdot P(-S_{div}(x_i)) \tag{7.5}$$

where $p(E)$ is energy-based probability function:

$$p(E) \propto exp(-\frac{E}{T}) \tag{7.6}$$

where T is a temperature parameter. Before using this equation, we normalize the three scores by dividing them by the maximum value among the samples. In this way, we are able to prioritize samples with higher correctness and complexity and more diversity when making a selection. We simply use a uniform distribution while proposing changes to the current state W. Therefore, we have $g(W \mid W') = g(W' \mid W)$.

7.3.4 Sample Expansion

Once we selected a set of samples with pseudo-labels, we employ an expansion process to improve the effectiveness of the training process, drawing inspiration from the recent success of contrastive learning [70]. Specifically, for each selected sample, we expand it to a group including other samples with similar pseudo-labels but distinct backgrounds. This allows us to increase the diversity of background noises in training data and further enhance the learning process.

The expansion score of a candidate sample x_j to a seed sample x_i is defined as

$$\mathcal{E}_{<i,j>} = \frac{\mathcal{S}(y_i, y_j)}{\mathcal{S}(x_i, x_j)} \tag{7.7}$$

where $\mathcal{S}()$ returns the similarity between two labels or two spectrograms. Note that each label y_i is a binary map and a spectrogram x_i is a real-valued image. We extract HoG features for both binary label maps and real-valued spectrograms and calculate their similarity as reciprocal of the Euclidean distance.

In the implementation, we first employed the MH sampler to select a set of spectrograms with high-quality pseudo-labels. We then considered each selected spectrogram as a seed. We empirically found that directly using Equation 7.7 may give high scores to samples whose pseudo-label is not similar to the seed sample. Therefore, we break the expansion into two steps. In the first step, we select $2t$ samples with the most similar pseudo-labels to the seed sample. In the second step, we select t samples from the $2t$ samples that have the most dissimilar background feature. We repeated this expansion process for each seed sample selected by the MH algorithm. Note that the expanded samples may include other seed samples and any sample may be used in the expansion of multiple seeds.

Figure 7.4 shows exemplar results of the expansion method, which is able to select samples (green boxes) with similar pseudo-label shapes but different background noise compared to the seed samples (first column).

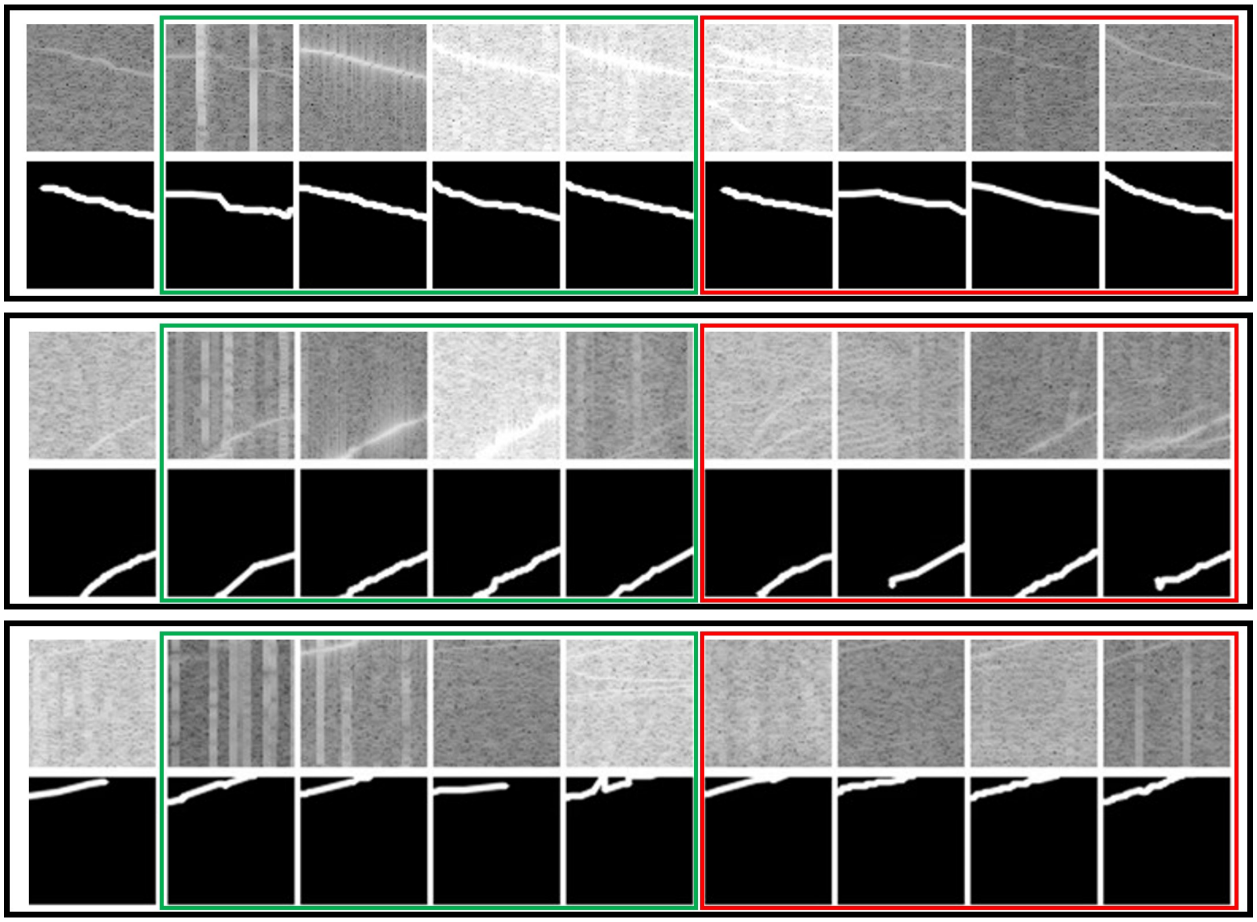


Figure 7.4: Two-step seed expansion. Each example is shown in a black box. Column 1: the seed samples. Columns 2-5: the sample selected in the second step of expansion. Columns 6-9: the samples were selected in the first step but discarded in the second step.

7.3.5 Triplet Loss

With the proposed seed-expansion process, we utilize a triplet loss [21] to fully explore the similarities and contrastiveness between different training samples. Let x_a and x_n denote a pair of seed samples that are supposed to have different shapes of whistle contours. We also select one sample from the expansion set of x_a and denote it by x_b. The pair of x_a and x_b are expected to have similar whistle contours. Let f_{x_a} denote the network output over the sample x_a, $HoG(\cdot)$ the HoG feature of a spectrogram or of a binary label. We define a triplet loss function for training the network f:

$$\begin{aligned} L_{triplet} = &||HoG(f_{x_a} \cdot y_a) - HoG(f_{x_b} \cdot y_b)||_2 \\ &- ||HoG(f_{x_a} \cdot y_a) - HoG(f_{x_n} \cdot y_n)||_2 \end{aligned} \tag{7.8}$$

which essentially trains the network to produce similar outputs for the positive pairs $< x_a, x_b >$ and dissimilar outputs for the negative pairs $< x_a, x_n >$.

We employ a least squared loss to train the network and add the above triplet loss into the loss function as follows:

$$L = \frac{1}{n}\sum_{i=1}^{n} ||y_i - f_{x_i}||_2 + \lambda \frac{2}{n} L_{triplet} \tag{7.9}$$

where n is the number of training samples and λ is a constant parameter.

7.4 Implementation

Dataset We apply the proposed method over a public dataset from the DCL workshop 2011[1], which includes recordings of five species. We use audio files that have human annotations for our experiments. We use 30 recordings for model training and 12 recordings for evaluation, with a total of 127 minutes of recording with a sampling frequency of 192 kHz and 12,539 human-annotated whistles in the training dataset, and 43 minutes of acoustic data with 6,011 annotated whistles in the evaluation dataset. We follow the same method [2] in [109] to transfer recordings to spectrograms.

[1]Available online: https://www.mobysound.org/workshops_p2.html

[2]Their code: https://github.com/Paul-LiPu/DeepWhistle

Generation of pseudo-labels We use the Graph-Search method outlined in [171] with the default parameters in *Silbido*[3] to generate whistle detections. Then we use OpenCV's Polyline function to draw detections with a white color on a black background with the same size as spectrogram. Spetrograms and their pseudo-labels are partitioned into 64x64 patches. Finally, we have 91,712 patches containing whistle detections and 91,712 patches containing no detections. The sample selection is performed on patches with whistle detections.

Network Architectures The same network architecture and training objective as described in [109] is utilized throughout the experiments except for the triplet loss experiments. The network consists of ten convolutional layers among which there are four residual blocks [71]. The model is trained with the Adam optimizer and default parameters in PyTorch for 1×10^6 iterations. The learning rate is initialized at 1×10^{-3} and is multiplied by 0.1 every 4×10^5 iterations, and a batch size of 64 is used for the training process.

Evaluation The evaluation of the experiment results is done using the *Silbido*'s scoring code with default parameters. The evaluation is conducted separately for each species, and the average values of F1-score among different species are reported.

7.5 Experiments

7.5.1 Sample Selection Variants

We implement multiple variants of sample selection methods and evaluate their performance under the same experimental setting. A total of five algorithms are considered for comparison.

Variant 1: Random selection We first introduce an intuitive method that learns deep

[3]Available online: https://roch.sdsu.edu/index.php/software/

models from the raw data. Once pseudo-labels are produced, a portion $n\%$ of them are randomly selected for training. We try n=10, 20, 30, ...100. When $n = 100$, the model training uses all the pseudo-labels, and we use this model in Equation 7.1 for the following experiments. In our results, this experiment is denoted "random."

Variant 2: Greedy selection Three scoring functions are applied to evaluate individual training samples, and n% of the training data with the highest scores is chosen as the training samples. We use n=10, 20, 30, ...90. The scoring functions are the correctness metric in Equation 7.1, the complexity metric in Equation 7.2, and the product of those two metrics. Because it is computationally infeasible to find the samples with the highest diversity with Equation 7.3, so this metric is not included in this set of experiments. These experiments are reported as "correctness", "complexity" and "correctness x complexity" when using S_{corr}, S_{comp}, and $S_{corr} \times S_{comp}$, respectively.

Variant 3: Greedy selection and expansion We greedily select n% seed samples with the top scores of $S_{corr} \times S_{comp}$ and expand them with t=4 samples. This set of experiments is used to study the effectiveness of the proposed sample expansion algorithm. In particular, we use n=5, 10, 20, 30, 40, 50, and 60. Note that we report the number of unique selected samples instead of n in our results, and this also applies to the following experiments with sample expansion. We call this set of experiments "greedy expansion."

Variant 4: Seed sampling and expansion In this set of experiments, we select n% seed samples using the proposed sampling method and expand each of the selected samples with t=4 extra samples. The sampling method is expected to identify seed samples with high-quality pseudo-labels. We perform experiments with n=5, 10, 20, 30, 40, 50, and 60. We use T=0.1 in Equation 7.6. We use the seeds selected in the experiments of "greedy expansion" as the initial state of the Markov process, and we run 10000 iterations before we have the final set of seeds. We call this set of experiments "sampling expansion."

Variant 5: Triplet loss We add the triplet loss to the models used in the above experiment of "sampling expansion" and study how this addition changes the performance. We use λ=1 in Equation 7.9. The same hyperparameters as the experiments on seed sampling and expansion are used. This set of experiments is referred to as "sampling expansion + triplet loss."

7.5.2 Results

Quantitative result Figure 7.5 presents the performance curves of various methods. The proposed method achieved the best performance of 84.72% in terms of F1 over the public test dataset. This is a significant improvement compared to the Graph-Search [171] method, which achieved 75.95%, and one recent probabilistic method [60], which achieved 83.40% . It is worth noting that these improvements were achieved without any labeling effort and can potentially be further improved with more raw data being collected and processed. The results with comparisons also lead to multiple observations.

Firstly, it is observed that the method "random" achieved F1-scores between 77.9% (when using 10% samples) to 80.91% (when using 60% samples). As a baseline, the model trained with 100% data without any selection has an F1-score of 80.75%, indicating that random selection may not significantly improve the model's performance. This study also suggests using all the pseudo-labels without any selection process is not the optimal way.

Secondly, sample selected by scoring functions "correctness," "complexity" and "correctness $\times$ complexity" consistently outperform the random methods while using different portions of the pseudo-labels. Among the three scoring functions, the use of the correctness function, S_{corr}, was not as competitive as using the complexity metric, S_{comp} or the the combined metrics $S_{comp} \times S_{corr}$. Using the correctness metric alone can select samples that have little whistle information in them and may explain why using the complexity metric alone or with

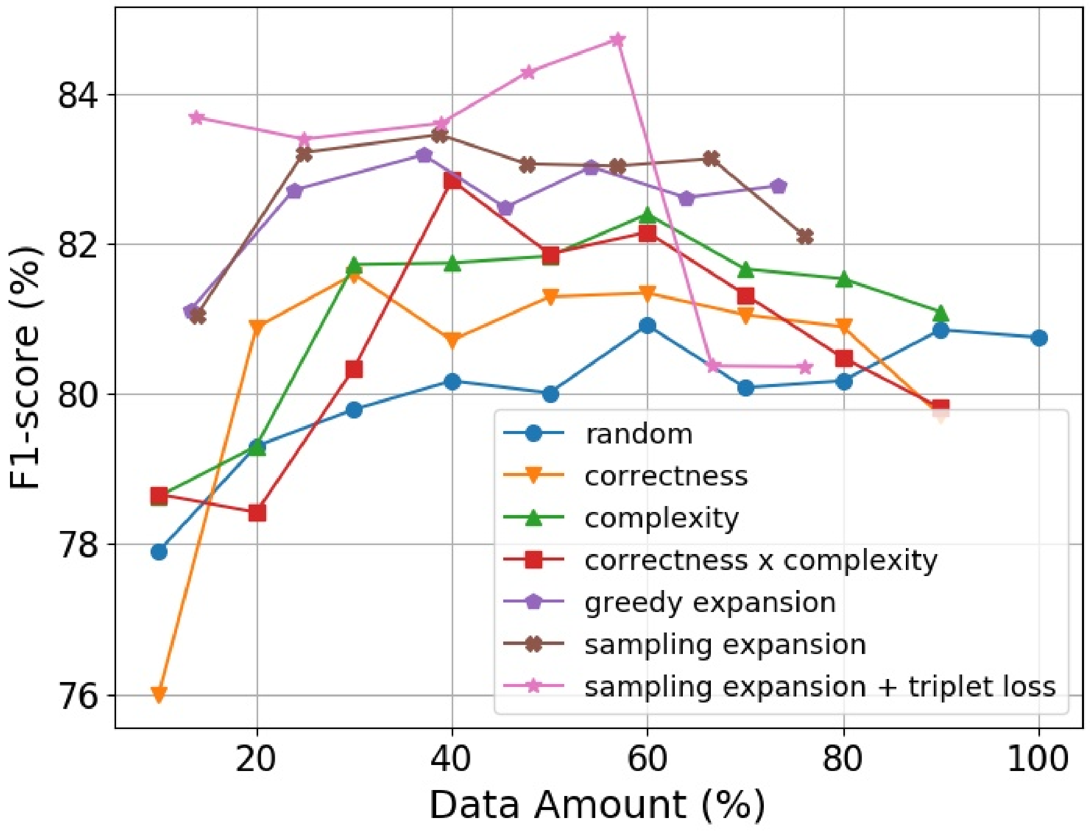


Figure 7.5: The performance curves by different sample selection methods.

correctness improves performance. When small dataset sizes are used, using correctness alone tends to be advantageous, but as additional data are sampled, the product of complexity and correctness becomes more effective. The results show that the greedy sample selection by $S_{corr} \times S_{comp}$ leads to consistently better results compared to random selection across a range of 30% to 80% of the samples. These results demonstrate the effectiveness of our scoring functions in selecting samples that improve the performance of the whistle extraction model.

Thirdly, the proposed methods (sampling and seed expansions) result in superior performance, especially while using relatively small portions of the pseudo-labels. For example, the method "sampling expansion" outperforms "greedy expansion" while the latter work much better than any of the three greedy methods "correctness", "complexity", and "correctness$\times$complexity". The expanded samples regularize and help the model training. With the same amount of seed samples, sampling process leads to increased number of unique samples than "greedy expansion", which indicates enlarged diversity of seed samples. These results suggest that the learning-via-sampling method can be used to improve training effectiveness without humans in the loop. However, the improvements brought by our sampler are not that significant while using a relatively large percentages of the available pseudo-labels. When more than 80% of the pseudo-labels are included, many of the proposed solutions do not work well. As the percentage increases, the similarity to the baseline random sampling increases, and any advantages of selective sampling disappear. An effective sampler attempts to increase diversity and remove poorly labeled data. When nearly all data are present, this is no longer possible.

Last, Our proposed triplet loss technique has been shown to effectively improve network performance when utilizing less than 60% of the data. The results of our comparisons demonstrate the efficiency of our seed expansion framework and triplet loss in achieving this improvement. However, when using a larger dataset that includes low-quality samples, the inclusion of these samples negatively impacts the model training and leads to poorer performance. These findings indicate that the triplet loss technique is more effective when applied to a smaller, higher-quality dataset.

Qualitative result Figure 7.6 illustrates the differences in how greedy-based selection and sampling-based selection work. In this example, the greedy selection uses the scoring function $S_{corr} \cdot S_{comp}$. To better visualize the results, all samples selected by both algorithms (i.e., common selections, shown in red boxes) are first identified. Other selected samples are sorted according to their similarity of pseudo-labels to the common samples. By examining the most similar samples in each selection, we can gain an understanding of how diversity changes between our sampling and greedy selection. In the figure, we use brown boxes to highlight the first different samples selected by both methods. Those examples indicate that the greedy algorithm tends to select samples with highly similar contour shapes. On the other hand, the sampling-based algorithm tends to select samples with more diverse contour shapes.

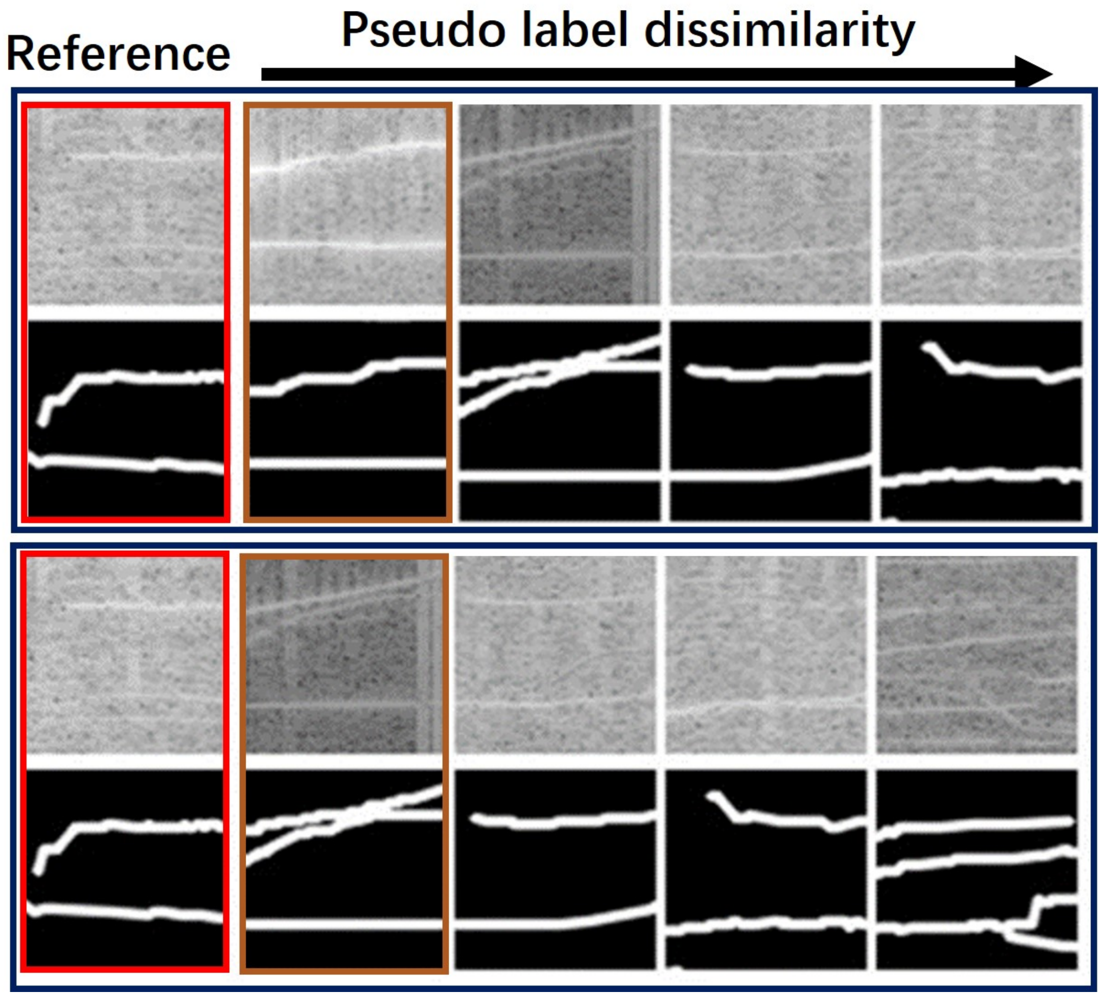


Figure 7.6: Selection of pseudo-labels by greedy (top row) or sampling-based (bottom row) methods. Column 1: common selections of samples. Column 2-5: the samples that have the most similar pseudo-labels as the common samples.

7.6 Conclusion

This chapter presents an effective framework for training deep whistle models from raw audio data without the need for human annotations. A series of metrics are proposed to evaluate the quality of pseudo-labels, and a sampling algorithm is designed to explore the optimal ways to combine these metrics. Additionally, an expansion method is introduced to find samples contrastive in background noises, which regularizes model training. The effectiveness of our methods is demonstrated through extensive comparisons and ablation studies on public datasets. While this work focuses on whistle extraction, the proposed learning-via-sampling methods have the potential to be applied in other areas of AI, including computer vision, multimedia computing, and natural language processing.

Chapter 8

Conclusion

In this dissertation, we introduce a series of data-centric algorithms designed to reduce the requirement for manual annotation in visual learning. Firstly, we implement a heuristic approach for synthesizing whistle extraction data. This approach combines whistle-like contours and background noises of time-frequency spectrogram. With the aid of generated samples, learning-based model is able to outperform traditional whistle detection methods in detecting whistles. Secondly, we propose a stage-wise generative adversarial network (GAN) framework to generate diverse whistle contours and background noise, and composite them to form realistic time-frequency spectrogram samples. The proposed method learns from real samples and outperforms the heuristic data generation method in terms of sample quality and variance. Thirdly, we implement policy-learning framework for sample-specific data augmentation on two image-based tasks: image classification and scene text recognition. For the image classification task, we incentivize the policy model to transform an image to have reduced prediction confidence. For the scene text recognition task, we encourage the policy model to generate transformed images that minimize the prediction distance between two STR models trained on different datasets. Our approach achieves comparable or better performance than alternative methods on both tasks. Finally, we investigate effective methods for utilizing raw data and pseudo-labels to train models for whistle extraction. We propose an improved loss function to address the challenge of missing whistle annotations in pseudo-labels. In addition, we develop a sampling method to select raw data with pseudo-labels based on various criteria, such as correctness, complexity, and diversity. Moreover, we select samples with similar pseudo-labels but different background noises for contrastive learning. The proposed methods enhance the performance of the model on various datasets and pseudo-label generation methods.

Future research. Apart from our proposed methods above, there are many intriguing problems for future research.

- Semi-supervised learning for whistle extraction. Our research involves the use of either annotated data or raw data. To leverage the strengths of both annotated and raw data, it would be beneficial to explore semi-supervised learning methods that utilize both types of data. In this scenario, we would need to select a subset of raw data while considering the annotated data, which differs from using raw data alone in chapter 7.
- Searching for optimal transformations for augmenting time-frequency spectrogram. Since the whistle signal recognition is highly sensitive to orientation and background noise, traditional image transformations such as rotation and flipping may not be effective for whistle extraction data. Our policy-based learning approach is capable of analyzing a range of potential transformations and selecting the most effective ones for improving performance.
- Active learning for whistle extraction. Annotating whistles is a time-consuming and expensive task, so it would be beneficial to identify the most informative samples for annotation. Our proposed scoring and sampling method could be adapted to an active learning setting. This approach may significantly reduce the annotation effort required and improve the performance of the model.

- Learning from human feedback. Providing human feedback to pseudo-labels and whistle extraction results is less costly than directly annotating the whistles. This feedback can be used to improve the training of models on raw data, thereby reducing the amount of manual annotation required.

Bibliography

[1] S. Agarwal and H. Farid. Detecting deep-fake videos from aural and oral dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 981–989, 2021.

[2] A. Althnian, D. AlSaeed, H. Al-Baity, et al. Impact of dataset size on classification performance: an empirical evaluation in the medical domain. *Applied Sciences*, 11(2):796, 2021.

[3] A. Antoniou, A. Storkey, and H. Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[4] P. Arbelaez, M. Maire, C. Fowlkes, et al. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

[5] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[6] R. Atienza. Data augmentation for scene text recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1561–1570, 2021.

[7] W. W. Au, D. A. Carder, R. H. Penner, et al. Demonstration of adaptation in beluga whale echolocation signals. *The Journal of the Acoustical Society of America*, 77(2):726–730, 1985.

[8] B. Baker, O. Gupta, N. Naik, et al. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

[9] J. G. A. Barbedo. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. *Computers and electronics in agriculture*, 153:46–53, 2018.

[10] W. H. Beluch, T. Genewein, A. Nürnberger, et al. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9368–9377, 2018.

[11] R. M. Bittner, B. McFee, J. Salamon, et al. Deep salience representations for f0 estimation in polyphonic music. In *ISMIR*, pages 63–70, 2017.

[12] C. Bowles, L. Chen, R. Guerrero, et al. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.

[13] G. Bradski. The opencv library. *Dr. Dobb's Journal of Software Tools.*

[14] H. Brumm and S. A. Zollinger. The evolution of the lombard effect: 100 years of psychoacoustic research. *Behaviour*, 148(11-13):1173–1198, 2011.

[15] M. C. Caldwell and D. K. Caldwell. Individualized whistle contours in bottle-nosed dolphins (tursiops truncatus). *Nature*, 207(4995):434–435, 1965.

[16] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[17] Q. Cao, L. Lin, Y. Shi, et al. Attention-aware face hallucination via deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 690–698, 2017.

[18] R. G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 18(7):690–706, 1996.

[19] T. Castells, P. Weinzaepfel, and J. Revaud. Superloss: A generic loss for robust curriculum learning. *Advances in Neural Information Processing Systems*, 33:4308–4319, 2020.

[20] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st international conference on image processing*, volume 2, pages 168–172. IEEE, 1994.

[21] G. Chechik, V. Sharma, U. Shalit, et al. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(3), 2010.

[22] H. Chen, J. Yan, N. U. R. Junejo, et al. Sparse representation based on tunable q-factor wavelet transform for whale click and whistle extraction. *Shock and Vibration*, 2018, 2018.

[23] L. Chen, L. Wu, Z. Hu, et al. Quality-aware unpaired image-to-image translation. *IEEE Transactions on Multimedia*, 21(10):2664–2674, 2019.

[24] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[25] Y. Chen, Y. Li, T. Kong, et al. Scale-aware automatic augmentation for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9563–9572, 2021.

[26] S. Chib and E. Greenberg. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335, 1995.

[27] J. Choi, M. Jeong, T. Kim, et al. Pseudo-labeling curriculum for unsupervised domain adaptation. *arXiv preprint arXiv:1908.00262*, 2019.

[28] P. N. Chowdhury, P. Shivakumara, U. Pal, et al. A new augmentation-based method for text detection in night and day license plate images. *Multimedia Tools and Applications*, 79(43):33303–33330, 2020.

[29] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3642–3649. IEEE, 2012.

[30] P. J. Clemins, M. T. Johnson, K. M. Leong, and A. Savage. Automatic classification and speaker identification of african elephant (loxodonta africana) vocalizations. *The Journal of the Acoustical Society of America*, 117(2):956–963, 2005.

[31] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

[32] D. O. Committee. Detection, classification, localization, and density estimation (dclde) of marine mammals using passive acoustic monitoring workshop dataset, 2011.

[33] P. C. Conant, P. Li, X. Liu, et al. Silbido profundo: An open source package for the use of deep learning to detect odontocete whistles. *The Journal of the Acoustical Society of America*, 152(6):3800–3808, 2022.

[34] E. D. Cubuk, B. Zoph, D. Mane, et al. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[35] E. D. Cubuk, B. Zoph, J. Shlens, et al. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[36] F. Dadouchi, C. Gervaise, C. Ioana, et al. Automated segmentation of linear time-frequency representations of marine-mammal sounds. *The Journal of the Acoustical Society of America*, 134(3):2546–2555, 2013.

[37] P. Dai, H. Zhang, and X. Cao. Sloan: Scale-adaptive orientation attention network for scene text recognition. *IEEE Transactions on Image Processing*, 30:1687–1701, 2020.

[38] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[39] T. Dao, A. Gu, A. Ratner, et al. A kernel theory of modern data augmentation. In *International Conference on Machine Learning*, pages 1528–1537. PMLR, 2019.

[40] T. DeVries and G. W. Taylor. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*, 2017.

[41] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[42] T. Dietterich. Overfitting and undercomputing in machine learning. *ACM computing surveys (CSUR)*, 27(3):326–327, 1995.

[43] W. R. Dillon and M. Goldstein. *Multivariate analysis: Methods and applications.* New York (NY): Wiley, 1984., 1984.

[44] L. Ding and A. Goshtasby. On the canny edge detector. *Pattern recognition*, 34(3):721–725, 2001.

[45] D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017.

[46] H. Emami, M. M. Aliabadi, M. Dong, et al. Spa-gan: Spatial attention gan for image-to-image translation. *IEEE Transactions on Multimedia*, 23:391–401, 2020.

[47] S. Fang, H. Xie, Y. Wang, et al. Read like humans: autonomous, bidirectional and iterative language modeling for scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7098–7107, 2021.

[48] A. Fawzi, H. Samulowitz, D. Turaga, et al. Adaptive data augmentation for image classification. In *2016 IEEE international conference on image processing (ICIP)*, pages 3688–3692. Ieee, 2016.

[49] M. Frid-Adar, E. Klang, M. Amitai, et al. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.

[50] Y. Gal, R. Islam, and Z. Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017.

[51] L. Galteri, L. Seidenari, M. Bertini, et al. Deep universal generative adversarial compression artifact removal. *IEEE Transactions on Multimedia*, 21(8):2131–2145, 2019.

[52] G. Ghiasi, Y. Cui, A. Srinivas, et al. Simple copy-paste is a strong data augmentation method for instance segmentation. *arXiv preprint arXiv:2012.07177*, 2020.

[53] A. Ghosh, H. Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

[54] D. Gillespie, M. Caillat, J. Gordon, et al. Automatic detection and classification of odontocete whistles. *The Journal of the Acoustical Society of America*, 134(3):2427–2437, 2013.

[55] R. Girshick, I. Radosavovic, G. Gkioxari, et al. Detectron, 2018.

[56] I. Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, et al. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[58] A. Graves, S. Fernández, F. Gomez, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[59] P. Gruden and P. R. White. Automated tracking of dolphin whistles using gaussian mixture probability hypothesis density filters. *The Journal of the Acoustical Society of America*, 140(3):1981–1991, 2016.

[60] P. Gruden and P. R. White. Automated extraction of dolphin whistles—a sequential monte carlo probability hypothesis density approach. *The Journal of the Acoustical Society of America*, 148(5):3014–3026, 2020.

[61] S. Gu, E. Holly, T. Lillicrap, et al. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

[62] I. Gulrajani, F. Ahmed, M. Arjovsky, et al. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[63] C.-e. Guo, S.-C. Zhu, and Y. N. Wu. Primal sketch: Integrating structure and texture. *Computer Vision and Image Understanding*, 106(1):5–19, 2007.

[64] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2315–2324, 2016.

[65] T. Haarnoja, A. Zhou, P. Abbeel, et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[66] X. C. Halkias and D. P. Ellis. Call detection and extraction using bayesian inference. *Applied Acoustics*, 67(11-12):1164–1174, 2006.

[67] D. Han, J. Kim, and J. Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5927–5935, 2017.

[68] K. Han and D. Wang. Neural network based pitch tracking in very noisy speech. *IEEE/ACM transactions on audio, speech, and language processing*, 22(12):2158–2168, 2014.

[69] R. Hataya, J. Zdenek, K. Yoshizoe, et al. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

[70] K. He, H. Fan, Y. Wu, et al. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[71] K. He, X. Zhang, S. Ren, et al. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[72] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[73] Y. He, C. Chen, J. Zhang, J. Liu, F. He, C. Wang, and B. Du. Visual semantics allow for textual reasoning better in scene text recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 888–896, 2022.

[74] D. Ho, E. Liang, X. Chen, et al. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pages 2731–2741. PMLR, 2019.

[75] G. Huang, Z. Liu, L. Van Der Maaten, et al. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[76] S.-W. Huang, C.-T. Lin, S.-P. Chen, Y.-Y. Wu, P.-H. Hsu, and S.-H. Lai. AugGAN: Cross domain adaptation with GAN-based data augmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 718–731, 2018.

[77] H. Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[78] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[79] P. Isola, J.-Y. Zhu, T. Zhou, et al. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[80] M. Jaderberg, K. Simonyan, A. Vedaldi, et al. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[81] A. K. Jain, J. Mao, and K. M. Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[82] V. M. Janik, S. L. King, L. S. Sayigh, et al. Identifying signature whistles from recordings of groups of unrestrained bottlenose dolphins (tursiops truncatus). *Marine Mammal Science*, 29(1):109–122, 2013.

[83] V. M. Janik and P. J. Slater. Context-specific use suggests that bottlenose dolphin signature whistles are cohesion calls. *Animal behaviour*, 56(4):829–838, 1998.

[84] A. Jaramillo-Legorreta, G. Cardenas-Hinojosa, E. Nieto-Garcia, et al. Passive acoustic monitoring of the decline of mexico's critically endangered vaquita. *Conservation Biology*, 31(1):183–191, 2017.

[85] J. Jeong, S. Lee, J. Kim, et al. Consistency-based semi-supervised learning for object detection. *Advances in neural information processing systems*, 32, 2019.

[86] J.-j. Jiang, L.-r. Bu, F.-j. Duan, et al. Whistle detection and classification for whales based on convolutional neural networks. *Applied Acoustics*, 150:169–178, 2019.

[87] J. M. Joyce. Kullback-leibler divergence. In *International encyclopedia of statistical science*, pages 720–722. Springer, 2011.

[88] S. Kahl, C. M. Wood, M. Eibl, and H. Klinck. BirdNET: A deep learning solution for avian diversity monitoring. *Ecological Informatics*, 61:101236, 2021.

[89] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, et al. Icdar 2015 competition on robust reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160. IEEE, 2015.

[90] D. Karatzas, F. Shafait, S. Uchida, et al. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493. IEEE, 2013.

[91] A. Karpathy, G. Toderici, S. Shetty, et al. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[92] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[93] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020.

[94] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[95] A. Kershenbaum and M. A. Roch. An image processing based paradigm for the extraction of tonal sounds in cetacean communications. *The Journal of the Acoustical Society of America*, 134(6):4435–4445, 2013.

[96] Y. Kim, J. Yim, J. Yun, and J. Kim. Nlnl: Negative learning for noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 101–110.

[97] Y. Kim, J. Yun, H. Shon, and J. Kim. Joint negative and positive learning for noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9442–9451.

[98] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[99] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[100] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[101] K. G. Larkin. Reflections on Shannon Information: In search of a natural information-entropy for images. *arXiv preprint arXiv:1609.01117*, 2016.

[102] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[103] C.-H. Lee, J.-L. Shih, K.-M. Yu, et al. Automatic music genre classification based on modulation spectral analysis of spectral and cepstral features. *IEEE Transactions on Multimedia*, 11(4):670–682, 2009.

[104] C.-Y. Lee and S. Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016.

[105] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

[106] J. Lee, S. Park, J. Baek, et al. On recognizing texts of arbitrary shapes with 2d self-attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 546–547, 2020.

[107] J. Lemley, S. Bazrafkan, and P. Corcoran. Smart augmentation learning an optimal data augmentation strategy. *Ieee Access*, 5:5858–5869, 2017.

[108] H. Li, P. Wang, C. Shen, et al. Show, attend and read: A simple and strong baseline for irregular text recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8610–8617, 2019.

[109] P. Li, X. Liu, K. Palmer, et al. Learning deep models from synthetic data for extracting dolphin whistle contours. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–10. IEEE, 2020.

[110] P. Li, X. Liu, and X. Xie. Learning sample-specific policies for sequential image augmentation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4491–4500, 2021.

[111] P. Li, M. A. Roch, H. Klinck, E. Fleishman, D. Gillespie, E.-M. Nosal, Y. Shiu, and X. Liu. Learning stage-wise gans for whistle extraction in time-frequency spectrograms. *IEEE Transactions on Multimedia*, 2023.

[112] Z. Li and D. Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[113] M. Liao, J. Zhang, Z. Wan, et al. Scene text recognition from two-dimensional perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8714–8721, 2019.

[114] T. P. Lillicrap, J. J. Hunt, A. Pritzel, et al. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[115] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32, 2019.

[116] C. Lin, M. Guo, C. Li, et al. Online hyper-parameter learning for auto-augmentation strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6579–6588, 2019.

[117] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

[118] S. Liu, M. Liu, M. Wang, et al. Classification of cetacean whistles based on convolutional neural network. In *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, pages 1–5. IEEE, 2018.

[119] W. Liu, D. Anguelov, D. Erhan, et al. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[120] W. Liu, C. Chen, K.-Y. K. Wong, et al. Star-net: a spatial attention residue network for scene text recognition. In *BMVC*, volume 2, page 7, 2016.

[121] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai. Richer convolutional features for edge detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3000–3009, 2017.

[122] V. Loginov. Why you should try the real data for the scene text recognition. *arXiv preprint arXiv:2107.13938*, 2021.

[123] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[124] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[125] J. Lu, W. Zhang, and H. Yin. Generate and purify: Efficient person data generation for re-identification. *IEEE Transactions on Multimedia*, 2021.

[126] W. T. Lu, L. Su, et al. Vocal melody extraction with semantic segmentation and audio-symbolic domain transfer learning. In *ISMIR*, pages 521–528, 2018.

[127] S. M. Lucas, A. Panaretos, L. Sosa, et al. Icdar 2003 robust reading competitions: entries, results, and future directions. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3):105–122, 2005.

[128] C. Luo, Y. Zhu, L. Jin, et al. Learn to augment: Joint data augmentation and network optimization for text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13746–13755, 2020.

[129] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey. Normalized loss functions for deep learning with noisy labels. In *International conference on machine learning*, pages 6543–6553. PMLR.

[130] E. Maggiori, Y. Tarabalka, G. Charpiat, et al. High-resolution aerial image labeling with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7092–7103, 2017.

[131] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transactions on image processing*, 17(1):53–69, 2007.

[132] A. Mallawaarachchi, S. Ong, M. Chitre, et al. Spectrogram denoising and automated extraction of the fundamental frequency variation of dolphin whistles. *The Journal of the Acoustical Society of America*, 124(2):1159–1170, 2008.

[133] G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi. BAGAN: Data augmentation with balancing GAN. *arXiv preprint arXiv:1803.09655*, 2018.

[134] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[135] D. McClosky, E. Charniak, and M. Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, 2006.

[136] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 33(11):2259–2272, 2011.

[137] D. K. Mellinger, S. W. Martin, R. P. Morrissey, et al. A method for detecting whistles, moans, and other frequency contour sounds. *The Journal of the Acoustical Society of America*, 129(6):4055–4061, 2011.

[138] G. Meng, T. Dai, S. Wu, et al. Sample-aware data augmentor for scene text recognition. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3978–3985. IEEE, 2021.

[139] B. Met-Montot, S. Cabon, G. Carrault, et al. Spectrogram-based fundamental frequency tracking of spontaneous cries in preterm newborns. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 1185–1189. IEEE, 2021.

[140] P. Mi, J. Lin, Y. Zhou, et al. Active teacher for semi-supervised object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14482–14491, 2022.

[141] A. Mishra, K. Alahari, and C. Jawahar. Scene text recognition using higher order language priors. In *BMVC-British Machine Vision Conference*. BMVA, 2012.

[142] T. M. Mitchell et al. *Machine learning*, volume 1. McGraw-hill New York, 2007.

[143] V. Mnih, K. Kavukcuoglu, D. Silver, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[144] D. Mu, W. Sun, G. Xu, et al. Random blur data augmentation for scene text recognition. *IEEE Access*, 9:136636–136646, 2021.

[145] J. Mu, W. Qiu, G. D. Hager, et al. Learning from synthetic animals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12386–12395, 2020.

[146] S. Mun, S. Park, D. K. Han, and H. Ko. Generative adversarial network based acoustic scene training set augmentation and selection using SVM hyper-plane. *Proceedings of the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 93–97, 2017.

[147] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013.

[148] M. Ngo, S. Karaoglu, and T. Gevers. Self-supervised face image manipulation by conditioning gan on face decomposition. *IEEE Transactions on Multimedia*, 2021.

[149] K. Nishi, Y. Ding, A. Rich, et al. Augmentation strategies for learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8022–8031, 2021.

[150] E.-M. Nosal. Methods for tracking multiple marine mammals with wide-baseline passive acoustic arrays. *The Journal of the Acoustical Society of America*, 134(3):2383–2392, 2013.

[151] O. Nuriel, S. Benaim, and L. Wolf. Permuted adain: Enhancing the representation of local cues in image classifiers. *arXiv preprint arXiv:2010.05785*, 2020.

[152] O. Nuriel, S. Fogel, and R. Litman. Textadain: Fine-grained adain for robust text recognition. *arXiv preprint arXiv:2105.03906*, 2021.

[153] Y. Oh, B. Kim, and B. Ham. Background-aware pooling and noise-aware loss for weakly-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6913–6922, 2021.

[154] T. Osakabe, M. Tanaka, Y. Kinoshita, et al. Cyclegan without checkerboard artifacts for counter-forensics of fake-image detection. In *International Workshop on Advanced Imaging Technology (IWAIT) 2021*, volume 11766, page 1176609. International Society for Optics and Photonics, 2021.

[155] J. N. Oswald, J. Barlow, and T. F. Norris. Acoustic identification of nine delphinid species in the eastern tropical pacific ocean. *Marine mammal science*, 19(1):20–037, 2003.

[156] S. Pandey, P. R. Singh, and J. Tian. An image augmentation approach using two-stage generative adversarial network for nuclei image segmentation. *Biomedical Signal Processing and Control*, 57:101782, 2020.

[157] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[158] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952.

[159] T. Q. Phan, P. Shivakumara, S. Tian, et al. Recognizing text with perspective distortion in natural scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 569–576, 2013.

[160] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition.* Prentice-Hall, Inc., 1993.

[161] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[162] A. J. Ratner, H. R. Ehrenberg, Z. Hussain, et al. Learning to compose domain-specific transformations for data augmentation. *Advances in neural information processing systems*, 30:3239, 2017.

[163] S. Reddy, M. Mathew, L. Gomez, et al. Roadtext-1k: Text detection & recognition dataset for driving videos. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11074–11080. IEEE, 2020.

[164] J. Ren, X. Jiang, J. Yuan, et al. Sound-event classification using robust texture features for robot hearing. *IEEE Transactions on Multimedia*, 19(3):447–458, 2016.

[165] P. Ren, Y. Xiao, X. Chang, et al. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

[166] X. Ren and S.-i. Kamata. Data augmentation for ancient characters via blend-font net. In *Thirteenth International Conference on Digital Image Processing (ICDIP 2021)*, volume 11878, page 1187806. International Society for Optics and Photonics, 2021.

[167] Z. Ren, X. Wang, N. Zhang, et al. Deep reinforcement learning-based image captioning with embedding reward. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 290–298, 2017.

[168] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.

[169] A. Risnumawan, P. Shivakumara, C. S. Chan, et al. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications*, 41(18):8027–8048, 2014.

[170] A. Rizzi, M. Antonelli, and M. Luzi. Instrument learning and sparse nmd for automatic polyphonic music transcription. *IEEE Transactions on Multimedia*, 19(7):1405–1415, 2017.

[171] M. A. Roch, T. Scott Brandes, B. Patel, et al. Automated extraction of odontocete whistle contours. *The Journal of the Acoustical Society of America*, 130(4):2212–2223, 2011.

[172] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[173] I. Sato, H. Nishimura, and K. Yokoi. Apac: Augmented pattern classification with neural networks. *arXiv preprint arXiv:1505.03229*, 2015.

[174] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[175] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[176] O. Serra, F. Martins, and L. R. Padovese. Active contour-based detection of estuarine dolphin whistles in spectrogram images. *Ecological Informatics*, 55:101036, 2020.

[177] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang. Deepcontour: A deep convolutional feature learned by positive-sharing loss for contour detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3982–3991, 2015.

[178] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016.

[179] B. Shi, X. Wang, P. Lyu, et al. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4168–4176, 2016.

[180] B. Shi, M. Yang, X. Wang, et al. Aster: An attentional scene text recognizer with flexible rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2035–2048, 2018.

[181] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[182] A. Shrivastava, T. Pfister, O. Tuzel, et al. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[183] D. Silver, A. Huang, C. J. Maddison, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[184] D. Silver, J. Schrittwieser, K. Simonyan, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[185] P. Y. Simard, D. Steinkraus, J. C. Platt, et al. Best practices for convolutional neural networks applied to visual document analysis. In *Icdar*, volume 3. Citeseer, 2003.

[186] L. Sixt, B. Wild, and T. Landgraf. Rendergan: Generating realistic labeled data. *Frontiers in Robotics and AI*, 5:66, 2018.

[187] B. L. Sjare and T. G. Smith. The relationship between behavioral activity and underwater vocalizations of the white whale, delphinapterus leucas. *Canadian Journal of Zoology*, 64(12):2824–2831, 1986.

[188] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[189] P. Sprechmann and G. Sapiro. Dictionary learning and sparse coding for unsupervised clustering. In *2010 IEEE international conference on acoustics, speech and signal processing*, pages 2042–2045. IEEE, 2010.

[190] H. Steinhaus et al. Sur la division des corps matériels en parties. *Bull. Acad. Polon. Sci*, 1:801–804, 1956.

[191] Z. Su, W. Liu, Z. Yu, D. Hu, Q. Liao, Q. Tian, M. Pietikäinen, and L. Liu. Pixel difference networks for efficient edge detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5117–5127.

[192] C. Tan, J. Xia, L. Wu, et al. Co-learning: Learning from noisy labels with self-supervision. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1405–1413, 2021.

[193] D. Tanaka, D. Ikami, T. Yamasaki, et al. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5552–5560, 2018.

[194] R. Tanno, A. Saeedi, S. Sankaranarayanan, D. C. Alexander, and N. Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11244–11253.

[195] A. G. Taruski. The whistle repertoire of the north atlantic pilot whale (globicephala melaena) and its relationship to behavior and environment. In *Behavior of marine animals*, pages 345–368. Springer, 1979.

[196] F. Thomsen, D. Franck, and J. K. Ford. On the communicative significance of whistles in wild killer whales (orcinus orca). *Naturwissenschaften*, 89(9):404–407, 2002.

[197] Y. Tokozume, Y. Ushiku, and T. Harada. Between-class learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5486–5494, 2018.

[198] T. Tran, T. Pham, G. Carneiro, et al. A bayesian data augmentation approach for learning deep models. *arXiv preprint arXiv:1710.10564*, 2017.

[199] F. van Beers, A. Lindström, E. Okafor, et al. Deep neural networks with intersection over union loss for binary image segmentation. In *ICPRAM*, pages 438–445, 2019.

[200] J. E. Van Engelen and H. H. Hoos. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440, 2020.

[201] H. Van Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[202] K. Vijayan, X. Gao, and H. Li. Analysis of speech and singing signals for temporal alignment. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1893–1898. IEEE, 2018.

[203] A. Waheed, M. Goyal, D. Gupta, et al. Covidgan: data augmentation using auxiliary classifier gan for improved covid-19 detection. *Ieee Access*, 8:91916–91923, 2020.

[204] A. Wali, Z. Alamgir, S. Karim, et al. Generative adversarial networks for speech processing: A review. *Computer Speech & Language*, 72:101308, 2022.

[205] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *2011 International Conference on Computer Vision*, pages 1457–1464. IEEE, 2011.

[206] T. Wang, D. J. Wu, A. Coates, et al. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 3304–3308. IEEE, 2012.

[207] X. Wang, Y. Hua, E. Kodirov, and N. M. Robertson. Imae for noise-robust learning: Mean absolute error does not treat examples equally and gradient magnitude's variance matters. *arXiv preprint arXiv:1903.12141*, 2019.

[208] X. Wang, J. Jiang, F. Duan, et al. A method for enhancement and automated extraction and tracing of odontoceti whistle signals base on time-frequency spectrogram. *Applied Acoustics*, 176:107698, 2021.

[209] Y. Wang, G. Huang, S. Song, et al. Regularizing deep networks with semantic data augmentation. *arXiv preprint arXiv:2007.10538*, 2020.

[210] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 322–330.

[211] Y. Wang, X. Pan, S. Song, et al. Implicit semantic data augmentation for deep networks. *Advances in Neural Information Processing Systems*, 32:12635–12644, 2019.

[212] Y. Wang, J. Peng, and Z. Zhang. Uncertainty-aware pseudo label refinery for domain adaptive semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9092–9101, 2021.

[213] Y. Wang, H. Xie, S. Fang, et al. Petr: Rethinking the capability of transformer-based language model in scene text recognition. *IEEE Transactions on Image Processing*, 31:5585–5598, 2022.

[214] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[215] P. White and M. Hadley. Introduction to particle filters for tracking applications in the passive acoustic monitoring of cetaceans. *Can. Acoust*, 36(1):146–152, 2008.

[216] C. Wigington, S. Stewart, B. Davis, et al. Data augmentation for recognition of handwritten words and lines using a cnn-lstm network. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 639–645. IEEE, 2017.

[217] S. C. Wong, A. Gatt, V. Stamatescu, et al. Understanding data augmentation for classification: when to warp? In *2016 international conference on digital image computing: techniques and applications (DICTA)*, pages 1–6. IEEE, 2016.

[218] B. Wursig and W. F. Perrin. *Encyclopedia of marine mammals*. Academic Press, 2009.

[219] X. Xia, T. Liu, B. Han, et al. Robust early-learning: Hindering the memorization of noisy labels. In *International conference on learning representations*, 2020.

[220] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama. Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, 32, 2019.

[221] C. Xie, M. Tan, B. Gong, et al. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

[222] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

[223] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang. Object contour detection with a fully convolutional encoder-decoder network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 193–202, 2016.

[224] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010.

[225] L. Yang, W. Zhuo, L. Qi, et al. St++: Make self-training work better for semi-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4268–4277, 2022.

[226] X. Yang, D. He, Z. Zhou, et al. Learning to read irregular text with attention mechanisms. In *IJCAI*, volume 1, page 3, 2017.

[227] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.

[228] C. Yi and Y. Tian. Assistive text reading from complex background for blind persons. In *International workshop on camera-based document analysis and recognition*, pages 15–28. Springer, 2011.

[229] D. Yoo and I. S. Kweon. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 93–102, 2019.

[230] W. A. Yost. Fundamentals of hearing: An introduction, 2001.

[231] K. Yu, C. Dong, L. Lin, et al. Crafting a toolchain for image restoration by deep reinforcement learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2443–2452, 2018.

[232] S. Yun, J. Choi, Y. Yoo, et al. Action-decision networks for visual tracking with deep reinforcement learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2711–2720, 2017.

[233] S. Yun, D. Han, S. J. Oh, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.

[234] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[235] J. Zeng, Q. Chen, Y. Liu, et al. Strokegan: Reducing mode collapse in chinese font generation via stroke encoding. In *proceedings of AAAI*, volume 3, 2021.

[236] H. Zhang, M. Cisse, Y. N. Dauphin, et al. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[237] Q. Zhang, R. Cong, C. Li, et al. Dense attention fluid network for salient object detection in optical remote sensing images. *IEEE Transactions on Image Processing*, 30:1305–1317, 2020.

[238] S. Zhang, S. Zhang, T. Huang, et al. Speech emotion recognition using deep convolutional neural network and discriminant temporal pyramid matching. *IEEE Transactions on Multimedia*, 20(6):1576–1590, 2017.

[239] X. Zhang, Y. Ge, Y. Qiao, et al. Refining pseudo labels with clustering consensus over generations for unsupervised object re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3436–3445, 2021.

[240] Z. Zhang and M. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[241] Z. Zhong, L. Zheng, G. Kang, et al. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13001–13008, 2020.

[242] Z. Zhong, L. Zheng, Z. Zheng, et al. Camstyle: A novel data augmentation method for person re-identification. *IEEE Transactions on Image Processing*, 28(3):1176–1190, 2018.

[243] T. Zhou, S. Wang, and J. Bilmes. Robust curriculum learning: from clean label detection to noisy label self-correction. In *International Conference on Learning Representations*, 2020.

[244] J.-Y. Zhu, T. Park, P. Isola, et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[245] S.-C. Zhu, C.-e. Guo, Y. Wu, and Y. Wang. What are textons? In *Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part IV 7*, pages 793–807. Springer, 2002.

[246] B. Zoph, E. D. Cubuk, G. Ghiasi, et al. Learning data augmentation strategies for object detection. In *European Conference on Computer Vision*, pages 566–583. Springer, 2020.

[247] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.