Exploiting sensor and process characteristics to tackle label scarcity in AIoT sensing

Luo, Wenjie

2023

https://hdl.handle.net/10356/169116

https://doi.org/10.32657/10356/169116

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

EXPLOITING SENSOR AND PROCESS CHARACTERISTICS TO TACKLE LABEL SCARCITY IN AIOT SENSING

LUO WENJIE
School of Computer Science and Engineering

2023

EXPLOITING SENSOR AND PROCESS CHARACTERISTICS TO TACKLE LABEL SCARCITY IN AIOT SENSING

LUO WENJIE

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2023

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

22/03/2023 .... Date

Luo wenjie .... LUO WENJIE

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

22/03/2023 [illegible]

. . . .

Date Prof. Rui Tan

Authorship Attribution Statement

This thesis contains material from four papers published in the following peer-reviewed journals and conferences in which I am listed as an author.

Chapter 2 contains material from a conference paper and its extended journal version:

- Wenjie Luo, Zhenyu Yan, Qun Song and Rui Tan, "PhyAug: Physics-Directed Data Augmentation for Deep Sensing Model Transfer in Cyber-Physical Systems", in *The 20th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), pages 31-46, May 18-21, 2021, Nashville, USA.* DOI: 10.1145/3412382.3458255.

- Wenjie Luo, Zhenyu Yan, Qun Song and Rui Tan, "Physics-Directed Data Augmentation for Deep Model Transfer to Specific Sensor," *ACM Transactions on Sensor Networks (TOSN), 19(1):1-30, 2022.* DOI: 10.1145/3549076.

The contributions of the co-authors are as follows:

- Wenjie Luo co-designed the approach, led the experiments and drafted the paper.
- Zhenyu Yan and Qun Song participated in the research discussions and assisted in the experiment design and paper drafting.
- Prof Rui Tan led the project direction and edited the manuscript drafts.

Chapter 3 contains material from a conference paper and its extended journal manuscript:

- Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan and Guosheng Lin, "Indoor Smartphone SLAM with Learned Echoic Location Features", in *The 20th ACM Conference on Embedded Networked Sensor Systems (SenSys), 13 pages, Nov 6-9, 2022, Boston, USA*. DOI: 10.1145/3560905.3568510.

- Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan and Guosheng Lin, "Indoor Smartphone SLAM with Learned Echoic Location Features", submitted to *IEEE Transactions on Mobile Computing.*

The contributions of the co-authors are as follows:

- Wenjie Luo co-designed the approach, led the experiments and drafted the paper.
- Qun Song and Zhenyu Yan participated in research discussions and assisted in the paper drafting.
- Prof Rui Tan led the project direction and edited the manuscript drafts.
- Prof Guosheng Lin participated in the discussion and provided insights on approach design.

Chapter 1 and Chapter 4 contain material from a workshop paper:

- Wenjie Luo and Rui Tan, "Physics-informed Machine Learning Model Generalization in AIoT: Opportunites and Challenges," *The 3rd Workshop on Data-Driven and Intelligent Cyber-Physical Systems for Smart Cities (DI-CPS), 6 pages, San Antonio, May 9, 2023, Texas USA*. DOI: 10.1145/3576914.3588751

The contributions of the co-authors are as follows:

- Wenjie Luo led the experiments and drafted the paper.
- Prof Rui Tan provided the project direction and edited the manuscript drafts.

22/03/2023 Luo wenjie

. . . .

Date LUO WENJIE

Acknowledgements

Firstly, I would like to express my utmost appreciation to my advisor, Prof. Rui Tan, for his exceptional guidance and mentorship throughout my Ph.D. studies. Prof. Tan has always been a guiding light for me, consistently leading me on the right path with his unwavering support and valuable insights. His diligence, critical thinking, and self-discipline have been a true inspiration and set an excellent example for me to follow. I always look forward to our weekly discussions, as they have sparked many innovative ideas and helped me stay on track with my research. Prof. Tan's mentorship has been invaluable in reminding me not to focus solely on obtaining a certificate, but rather to become an expert in my research field. His words of wisdom have been instrumental in shaping my mindset and guiding me toward achieving my goals. I am truly honored to have had the privilege of being mentored by Prof. Tan, and words cannot express the depth of my gratitude to him. I will always remember his guidance and mentorship, and strive to apply the lessons I have learned from him in my future endeavors.

Secondly, I would like to express my gratitude to my dear family members for their steadfast support. I want to extend a special thanks to my dearest mother who was diagnosed with cancer just before I embarked on my Ph.D. journey. Even while enduring the grueling chemotherapy process, she has gone out of her way to avoid distracting me from my studies and has always been encouraging me to continue. I want to thank my father, who has been the unwavering support pillar of our family for so many years. Even during our darkest times, he never complains a word. I also want to express my appreciation to my younger sister, who has taken care well of my parents in my absence. And last but not least, I want to thank my wife, who chose to stand by my side during the most challenging times. To my dear baby, your arrival has brought us immense joy and hope.

Lastly, I would like to thank all my friends from the NTUIoT sensing group. It is been a delightful journey working with talented people like you. I would like to thank senior members for setting excellent examples for us to follow and always being available for consulting.

A special thanks to CNCL lab manager Toh Leong Teck for his willingness to go above and beyond to ensure that our experimental needs are met. His expertise and dedication have been invaluable to the success of my research projects.

To my dear family

Abstract

The Internet of Things (IoT) is a global cyber network that consists of trillions of smart objects, edge devices, fogs, and cloud computing systems. It is a fabrication of data generation infrastructures and computing infrastructures. It is estimated that there will be more than 50 billion edge devices connected to the IoT by 2025 [1]. Artificial intelligence (AI), on the other hand, is the driving force that unleashes the full potential of IoT. Recent advances in machine learning, especially in deep learning, have inspired the development of deep neural network (DNN)-based smart sensing applications in IoT. The integration of AI and IoT gives rise to the paradigm of Artificial Intelligence of Things (AIoT). AIoT has paved its way in reshaping smart wearables, smart homes, smart cities, and smart industries. The effectiveness of DNN models hinges on the availability of large, labeled datasets that can unveil valuable feature representations. The widespread use of DNN models in computer vision (CV), natural language processing (NLP), and voice sensing can be attributed to the abundance of labeled training datasets. However, in the domain of AIoT sensing, the availability of such high-quality datasets is limited. The fundamental reason arises from two aspects of AIoT sensing data. Firstly, the annotation of IoT sensing data by humans is a challenging task. Despite the abundance of IoT sensing data, the human-uninterpretable nature of AIoT sensing data makes it difficult for human assistants to understand the meaning behind it if the labeling process is separated from the data collection. Consequently, constructing labeled datasets for AIoT applications demands significant human effort. Secondly, the heterogeneity of IoT sensor hardware and/or the deployment environments of DNN models introduce domain shifts. The domain shifts cause the deviation in data distribution between the testing and training data. It leads to label scarcity for the target-domain dataset, posing challenges for domain adaptation. In summary, label scarcity stands out as a primary challenge in developing effective AIoT sensing applications.

Existing solutions for addressing the label scarcity rely on machine learning techniques, such as self-supervised learning, few-shot learning, or transfer learning.

These approaches are data-driven and fall short of exploiting the underlying first principles that govern data generation. This thesis proposes to utilize prior knowledge governing data generation to address the label scarcity in the development of machine learning algorithms for AIoT sensing applications. Specifically, it exploits the sensor and process characteristics to develop efficient machine learning models such that the demand for the labeled data can be reduced. The problems to be addressed and the proposed approaches are presented in two studies forming the main technical development of this thesis.

The first study targets the *run-time domain shift* phenomenon in AIoT sensing applications. It utilizes both the sensor and process characteristics to address the label scarcity for target domains where massive labeled datasets are not available. In physics-rich AIoT, the domain shifts are often governed by certain physical laws. For instance, the distribution of indoor temperature is governed by the fluid dynamics model and the microphone data received is determined by its frequency response curve. A comprehension of these physical laws can aid in the creation of more efficient and generalized machine learning models by incorporating physical laws as additional information during the model training. This study proposes an approach called *physics-directed data augmentation* (PhyAug) to address the run-time domain shifts caused by the sensor and/or process characteristics. Different sensors can respond to the same excitation differently, which results in data distribution deviations. The sensed data is often determined by the hardware characteristics of the sensors and can be described by certain parametric models. PhyAug leverages such parametric models for DNN model transfer. In addition, the process of sensed data collection can exhibit certain characteristics. For example, the sensor readings mounted on human bodies are often correlated with the human body's movement. PhyAug exploits the process characteristics to augment the training data. PhyAug has the following two key features. First, PhyAug augments the training data strategically by following the physical law to transfer DNNs instead of using *ad hoc* perturbations or transformations like conventional data augmentation does. Second, PhyAug only requires a small amount of target domain data for law fitting. This data requirement is less than the competing baselines that use transfer learning techniques. The study applies PhyAug to a series of case studies and compares its performance with other possible approaches. The superior performance achieved by PhyAug highlights the potential of PhyAug

in addressing a range of physics-governed domain shifts for various AIoT sensing applications.

The second study utilizes the process characteristics of data collection to improve the effectiveness of the machine learning algorithm. It addresses the label scarcity in developing DNN-based systems for new AIoT applications. Despite the limited availability of labeled training data, AIoT sensing applications are featured in a wealth of unlabeled data gathered by billions of devices. In machine learning, self-supervised learning is a new technique that can effectively extract feature representations from unlabeled sensing data. Self-supervised learning has found many successful applications in the field of CV, NLP, voice sensing, etc. Nevertheless, using machine learning techniques directly without taking into account the specific characteristics of the sensing data may result in the suboptimal performance of DNN models. This study considers the process characteristics of the sensing data collection and utilizes it to design customized machine learning algorithms. Specifically, it designs ELF-SLAM, a smartphone-based simultaneous localization and mapping (SLAM) system using the learned echoic location features (ELFs). The following challenges are tackled during the design of the system. First, to alleviate the data labeling effort, ELF-SLAM uses a smartphone to collect both inertial moving unit (IMU) data and acoustic data simultaneously. The IMU data is used to estimate user trajectory and provide label information for acoustic data. Due to the long-run drifting issue of IMU data, acoustic echoes are used to detect the loop closures for trajectory regularization. However, conventional acoustic features are ineffective for loop closure detection. To overcome this challenge, this study opts to learn an effective embedding using a DNN model. Second, to address the ineffectiveness of the machine learning algorithm on unlabeled acoustic data, this study exploits the process characteristics of acoustic data collection to design a customized contrastive learning procedure to learn ELFs that can be used to signal loop closures. The customization includes the use of the acoustic simulator for model pre-training, an effective positive/negative data pairing approach based on the spatial-temporal relationship of acoustic data and a new model finetuning based on unlabeled data for target room adaptation. ELF-SLAM exhibits satisfactory performance in both mapping and localization. This study demonstrates the necessity and effectiveness of integrating process characteristics of sensing data collection in machine learning algorithms for AIoT sensing applications.

In summary, this thesis targets the label scarcity challenge in developing machine learning algorithms for AIoT sensing applications. It identifies the two fundamental factors contributing to this label scarcity, namely, *uninterpretability* of sensed data and *run-time domain shifts*. This thesis proposes to exploit the sensor and process characteristics to address the domain shifts and reduces the machine learning algorithm's reliability on the labeled data. The effectiveness of the proposed approach is demonstrated in a set of distinct AIoT applications. Future research will continue to explore more approaches in combining prior knowledge with machine learning algorithms to develop more efficient DNN models for AIoT sensing applications.

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
IoT	The Internet of Things
AIoT	Artificial Intelligence of Things
CV	Computer Vision
CNN	Convolutional Neural Network
NLP	Natural Language Processing
SLAM	Simultaneous localization and mapping
IMU	Inertial moving unit
ELF	Echoic location feature
CL	Contrastive learning
DNN	Deep neural network
PIML	Physics-informed machine learning
ISM	Image source modeling
CDCL	Cross-domain contrastive learning
ADDA	Adversarial Discriminative Domain Adaptation
FRC	Frequency response curve
LPF	Low-pass filter
MFCC	Mel-Frequency Cepstral Coefficients
PSD	Power spectral density
MLP	Multi-layer perceptron
t-SNE	T-distributed Stochastic Neighbor Embedding
SNR	Signal-to-noise ratio
KWS	Kwyword Spotting
ASR	Automated speech recognition
ARR	Acoustics-based Room Recognition
ABS	Acoustic background spectrogram
STFT	Short-time Fourier transform

FIV	Field of view
TDoA	Time difference of arrival
SVM	Support vector machine
BART	Bayesian Algebraic Reconstruction Technique
LSQR	Least Squares with QR-factorization
DE	Differential evolution
AoA	Angle-of-arrival
RSSI	Received signal strength indicators
CSI	Channel state information
EMR	Electromagnetic radiation
DTW	Dynamic time warping
FHSS	Frequency hopping spectrum spread
ESS	Echo sequence similarity
PCA	Principal component analysis
RANSAC	Random sample consensus
AP	Access points
RIR	Room impulse response

Chapter 1

Introduction

The Internet of Things (IoT) is a global network consisting of trillions of edge sensors. According to estimates, the number of IoT devices connected via the internet is expected to reach 55 billion by 2025. These IoT devices continuously collect data from their surroundings, generating an astounding 1 billion gigabytes of data every day [1]. Artificial intelligence (AI), due to its capability in dealing with big data, can transform the big data generated by IoT into actionable insights and enhances the smart sensing capabilities of the IoT. The integration of AI with IoT has given rise to the paradigm of Artificial Intelligence of Things (AIoT). AIoT aims to create intelligent systems that can learn, reason, and make decisions based on data collected from IoT devices. As AIoT technology advances, it is expected to have a profound impact on different aspects of human daily lives. First, AIoT will enable more intelligent, efficient, and autonomous systems in various fields, including healthcare, transportation, manufacturing, and smart cities. Second, AIoT can help organizations and businesses to optimize their operations, improve customer experiences, and reduce costs by providing insights and predictions based on data analysis.

1.1 Label Scarcity in AIoT

However, developing machine learning algorithms in the field of AIoT sensing faces the label scarcity challenge, which is explained as follows.

Deep neural networks (DNNs), due to their remarkable performance in feature learning, have been widely used to handle complex and unstructured data. The performance of the DNN model is proportional to the depth of the model and the total number of neurons. The *state-of-the-art* DNN models contain billions of neurons. For example, ChatGPT is estimated to consist of more than 170 billion neurons. To achieve optimal performance, the DNN model must be trained with a vast amount of data to find the best possible parameters. During the training, the DNN model is presented with labeled examples and it learns to identify patterns and relationships in the input data that are associated with specific output labels. The model is adjusted iteratively using the gradient descent optimization algorithm until it can make accurate predictions on new and unseen data. A basic requirement for DNN model training is the availability of extensive training datasets. The success of DNNs in CV, NLP, and voice sensing can be attributed to the massively available labeled training datasets. For example, ImageNet [2] contains more than 1.4 million labeled images from over 1,000 categories; LibriSpeech corpus [3] contains more than 1,000 hours of audio recordings with transcriptions.

Despite the abundance of unlabeled AIoT sensing data, the label scarcity hinders the development of DNN models for AIoT sensing applications. The fundamental difficulty arises from two aspects of AIoT sensing applications.

First, **uninterpretable nature of most IoT sensing data** renders it challenging to construct semantic labels for DNN model training. Fig. 1.1 uses a simple example to illustrate this challenge. The left part depicts the conventional approach for creating labeled training datasets in the field of CV and NLP. The media data, such as images, audio, and texts, are interpretable by humans. Thus, human assistants can be involved in annotating the data after the data collection is completed. The separable collection and labeling nature of media data allows the construction of large amounts of labeled datasets such as ImageNet. Multiple platforms, such as Amazon Mechanical Turk [4] are established to facilitate this process. On the contrary, as shown in the right part of Fig. 1.1, the human-uninterpretable nature of IoT sensing data makes it difficult for labeled dataset construction. Human assistants cannot interpret the data if the data annotation process is performed after the data collection and therefore the context has been lost. In general, IoT data collection and labeling must be performed in tandem, which will incur much more manual effort. While the above example involves *semantic labels*, for the AIoT

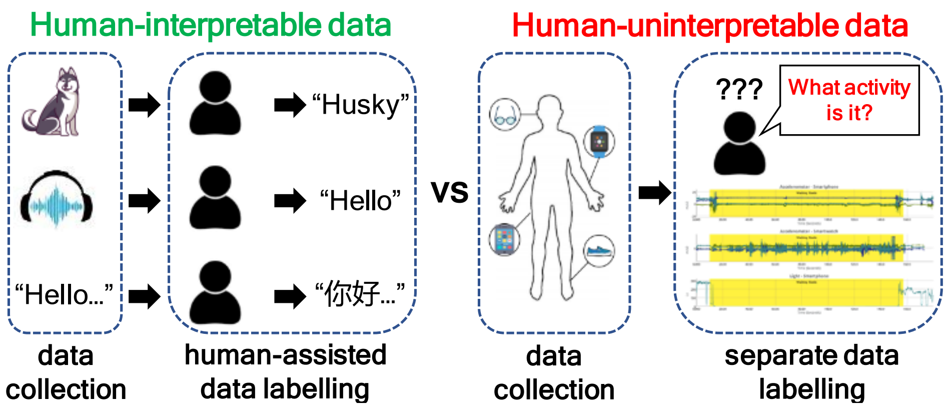


FIGURE 1.1: Left: human-interpretable media data. Right: human-uninterpretable IoT sensing data. The uninterpretability of IoT sensing data makes it difficult for human assistants to annotate data if the labeling process is separated from data collection.

applications with *quantitative labels*, e.g., patients' height, blood pressure, cholesterol level, and real-time positions, etc, obtaining such quantitative labels often require dedicated equipment thus incurring extra overhead. The above difficulties in annotating IoT data with either semantic or quantitative labels result in the scarcity of extensive and labeled training data in the field of AIoT sensing.

Second, **run-time domain shifts** are common in AIoT sensing applications. The domain shifts result in distribution deviations between the collected sensor data at the inference stage and the standard training data. A pre-trained DNN model on a standard training dataset will have degraded performance if evaluated on the collected sensor data. In the field of AIoT sensing, the cause of domain shifts can stem from differences in the sensors' hardware or the changing ambient environment where the DNN model is deployed. Domain shift can pose a significant challenge in real-world AIoT sensing applications, especially when DNN models often operate in dynamic environments where the underlying data distribution can shift over time. As a result, the run-time domain shifts need to be carefully addressed while developing DNN models for AIoT sensing applications. Addressing domain shift requires techniques such as domain adaptation or transfer learning using labeled/unlabeled data from the target domain. Data centralization approaches, such as constructing datasets for training generalized DNN models are infeasible in the field of AIoT sensing. Run-time domain shifts exacerbate the label scarcity challenge for AIoT sensing.

Table 1.1: Summary of used sensor and physical characteristics.

Used characteristics	Study	Prior knowledge	Application	Used technique	Label types
Sensor	PhyAug	Frequency response curve	Voice sensing	Data augmentation	Semantic
			Room recognition		Quantitative
		Fisheye model	Computer vision		Semantic
Process		Slowness model	Location sensing		Quantitative
	ELF-SLAM	Kinetic relationship		Contrastive learning	

1.2 Methodology

In physics-rich IoT sensing, the data generation or domain shifts are governed by certain first principles. For instance, the distribution of indoor temperature is governed by the fluid dynamics model and the microphone data received is determined by its frequency response curve. A comprehension of the first principles can aid in the creation of more efficient machine learning models while reducing the demand for labeled data. In machine learning, physics-informed machine learning (PIML) [5, 6] is recently proposed to leverage the physical knowledge obtained from the observational data to enhance the performance of machine learning models. PIML-based DNN models can outperform conventional models and require fewer training data in modeling multi-physics and multi-scale systems.

Inspired by the success of PIML, this thesis leverages two common AIoT sensing characteristics, i.e., the sensor and process characteristics to address the label scarcity challenge in AIoT sensing. **First**, sensor characteristic variation often causes domain shifts between the sensor data collected in the wild and the standard training data. It is a common phenomenon in the field of AIoT sensing and can lead to degraded performance of pre-trained DNN models when evaluated on the collected sensor data. The prevalent transfer learning techniques addressing the domain shifts can reduce the amount of training data needed in the target domain compared with training from scratch. However, they still require a substantial amount of data to achieve satisfactory performance. This thesis proposes to leverage sensor characteristics causing the domain shift to guide the adaptation of DNN models. Specifically, it uses a minimum amount of data collected by the target sensor to estimate the parameters of identified sensor models, then uses the parametric models to generate augmented target sensor data. The augmented target-sensor data are used to transfer the source-domain DNN. Compared with the conventional transfer learning approaches, the proposed approach can pinpoint the source of domain shift and only require a small amount of the target domain data to estimate the sensor characteristics. **Second**, the processes of IoT sensing data generation/collection have specific characteristics and can aid in the creation of more efficient machine learning algorithms by integrating process characteristics during model training. The proposed usage of process characteristics in this thesis can be divided into two categories. (1) The process characteristics of data generation are used to augment data samples for DNN model transfer. For example, a slowness model governing seismic wave propagation in a heterogeneous medium can be described using a parametric model. The unknown parameters can be fitted using a small amount of genuine data and then the fitted model is used to generate augmented data to adapt a DNN model that is trained on homogeneous medium data. The data augmentation using a fitted model allows a reduction of demand for target domain data. (2) The process characteristics of data collection are used in the design of machine learning algorithms. This thesis exploits the unique characteristics of acoustic data collection to design customized contrastive learning procedures for acoustic feature learning. This feature learning process does not require label information and only requires a few minutes of unlabeled data. Table 1.1 gives a detailed summary of the used sensor and process characteristics, the label types, the used techniques, and their corresponding applications in this thesis.

The following two sections in this chapter introduce the details of exploiting sensor and process characteristics to tackle label scarcity in AIoT sensing applications.

1.3 PhyAug: Physics-Directed Data Augmentation for Deep Sensing Model Transfer in Cyber-Physical Systems

Recent advances in deep learning have attracted great interest in applying it in various embedded sensing systems. The deep neural networks (DNNs), albeit capable of capturing sophisticated patterns, require significant amounts of labeled training data to realize the capability. A sensing DNN trained on a design dataset is often observed run-time performance degradations, due to *domain shifts* [7]. The shifts can be caused by the sensor and process characteristics deviations in the real deployments from those captured by the design dataset.

Transfer learning [8] has received increasing attention for addressing domain shifts. It is a cluster of approaches aiming at storing knowledge learned from one task and applying it to a different but related task. Under the transfer learning scheme, ideally, with little new training data, a DNN trained from the *source domain* (i.e., the design dataset) can be transferred to the *target domain* (i.e., data captured by a specific sensor in real deployment). Prevalent transfer learning techniques, including *freeze-and-train* [9] and *domain adaptation* [8], require substantial training data collected in the target domain. The freeze-and-train approach retrains selected layers of a DNN with new target-domain samples to implement the model transfer. Domain adaptation trains a new DNN to transform the target-domain inference data back to the source domain. For instance, the Mic2Mic [10] trains a cycle-consistent generative adversarial network (CycleGAN) to perform the translation between two microphones with distinct characteristics. The training of CycleGAN requires about 20 minutes of microphone recording from both domains for a keyword spotting task [10]. In summary, although the prevalent transfer learning techniques reduce the demands on the target-domain training data in comparison with learning from scratch in the target domain, they still need substantial target-domain data to implement the model transfer. This thesis exploits the first principles governing the domain shifts to reduce the demand on target-domain data for model transfer, vis-à-vis the *physics-regardless* approaches [8–10].

Recent studies attempt to incorporate prior knowledge in the form of commonsense [11] or physical laws [12, 13] to increase learning efficiency. The presentation of the prior knowledge to learning algorithms is the core problem of *physics-constrained machine learning*. In [12, 13], the closed-form physical laws are incorporated into the loss function of DNN training. The improved learning efficiency of physics-constrained machine learning encourages exploiting first principles to address domain shifts more efficiently. However, physics-constrained machine learning requires new DNN architectures and/or training algorithms; it is not to exploit first principles in transferring existing DNNs to address the domain shift problems.

In the domain of physics-rich AIoT, the domain shifts caused by sensors or process characteristics are often governed by first principles. For example, the performance of a microphone is often characterized by the frequency response curve [7];

a fisheye camera is characterized by the polynomial function [14], etc. This thesis proposes a new approach called *physics-directed data augmentation* (PhyAug) to use a minimum amount of data collected from the target sensor to estimate the parameters of the first principle governing the domain shift caused by a specific sensor, then use the parametric first principle to generate augmented target-domain training data. The augmented target-domain data samples are used to transfer or retrain the source-domain DNN. PhyAug has the following two key features. **First**, different from the conventional data augmentations that apply unguided *ad hoc* perturbations (e.g., noise injection) and transformations (e.g., scaling, rotation, etc) on existing training data to improve the DNNs' robustness against variations, PhyAug augments the training data strategically by following first principles to transfer DNNs. **Second**, PhyAug uses augmented data to represent the domain shifts and thus requires no modifications to the legacy DNN architectures and training algorithms. In contrast, recently proposed domain adaptation approaches based on adversarial learning [10, 15–17] update the DNNs under new adversarial training architectures that need extensive hyperparameter optimization and even application-specific redesigns. Such needs largely weaken their readiness, especially when the original DNNs are sophisticated such as the DeepSpeech2 [18] for automatic speech recognition.

This thesis applies PhyAug to five case studies and quantifies the performance gains compared with other possible approaches. The first two case studies aim to adopt DNNs for keyword spotting (KWS) and automatic speech recognition (ASR) respectively to individual microphones. KWS and ASR differ in DNN model depth and complexity. The domain shifts are from the microphone's hardware characteristics. PhyAug profiles the source and the target microphones by playing a 5-second white noise, then augmenting the source-domain data to re-train the DNN for the target domain. PhyAug recovers the microphone-induced accuracy loss by 53%-72% and 37%-70% in KWS and ASR, respectively. The third case study is acoustics-based room recognition (ARR). The experiment shows that the room recognition accuracy drop can be up to 80% if the pre-trained model is evaluated using the data collected from a specific smartphone microphone. PhyAug profiles the source and the target smartphones by recording 1-minute acoustic background spectrograms in any room simultaneously, then augmenting the source smartphone's data to train the DNN for the target smartphone. PhyAug recovers the accuracy loss by 33%-80% for the target smartphone. The fourth case study focuses on fisheye image recognition (FIR). PhyAug adapts a ResNet-50 DNN designed for pinhole cameras to a specific fisheye camera using the estimated parameters. The parameters estimation for a fisheye camera only requires around 20 image samples taken on a checkerboard picture. PhyAug recovers the camera-induced object recognition accuracy loss by 72% and avoids the compute-intensive image rectification. In the last case study of seismic source localization with TDoA fingerprints, by exploiting the first principle of signal propagation in uneven media, PhyAug only requires 3% to 8% of labeled TDoA measurements used by the vanilla fingerprinting approach in achieving the same localization accuracy.

1.4 Indoor Smartphone SLAM with Learned Echoic Location Features

Lack of labeled training data is often the biggest obstacle that hinders the design of machine learning algorithms for a new AIoT sensing application. To address the *scarcity of labeled training data*, this study exploits the process characteristics of data collection to design customized contrastive learning procedures and learn a new acoustic feature using limited unlabeled data to enable a smartphone-based simultaneous localization and mapping (SLAM) system.

Location sensing is a fundamental service of mobile operating systems. More than 73% of the top 100 free Android apps on Google Play [19] require location information. In the last two decades, research has exploited various smartphone's built-in sensing modalities for indoor location sensing, which include Wi-Fi [20, 21], GSM [22], FM radio [23], visible light [24], imaging [25], acoustic background [26], and geomagnetism [27]. However, these sensing modalities have their own limiting factors. For instance, radio frequency (RF) signals are susceptible to electromagnetic noises and transceivers' automatic gain controls. Visible light sensing suffers blockage. Visual imaging may generate privacy concerns in certain spaces and times. The acoustic background only gives room-level granularity. Hence, identifying new modalities from smartphones' common built-in hardware to enrich the location sensing toolbox has been an interest of research.

Exploiting smartphone's built-in audio hardware for active location sensing receives increasing research [28–33]. The active sensing uses smartphone's loudspeaker to emit excitation sounds in the target indoor space and microphone to capture the echoes from the surroundings. The echoes carry information that depicts how a sound emitted from a specific location reverberates in the indoor space. The existing approaches can be classified into two categories. The first category, namely, *analytic approach* [31–33], estimates the smartphone's location by analyzing the processes of the sound reflections by nearby surfaces (e.g., walls). The image source modeling (ISM) is a prevailing analysis technique. However, when the surroundings are complex (e.g., irregular surfaces, many nearby objects with complex 3D structures, etc), accurate ISM becomes intractable. Thus, the existing studies following the analytic approach often make simplifying assumptions that the major reflectors are at most two nearby walls [31–33]. The second category, namely, *fingerprint approach* [28–30], uses the echoes captured by the smartphone at a certain location as the fingerprint of the location and then applies supervised learning to build a location recognition model. However, the blanket process of collecting labeled fingerprints at spatially fine-grained locations to form the training dataset imposes a high overhead. Thus, existing studies only target room-level localization [28, 30] or recognize a limited number of locations (11 closed locations in [29]).

Nevertheless, the fingerprint approach has the potential in offering good generalizability because it does not make specific assumptions about the surroundings. Moreover, the high spatial resolution achieved in [29] encourages further investigation of whether satisfactory resolutions can be maintained when the number of fingerprinted locations increases. To develop a preliminary understanding, in this thesis, a blanket process of fingerprinting 128 locations using excitation chirps with near-inaudible frequencies is conducted in a $16 \times 28\,\mathrm{m}^2$ space hosting tens of cubicles and several meeting areas. The analysis by applying supervised learning to the collected data shows that the fingerprint approach can maintain sub-meter localization accuracy. The results suggest that acoustic echo is a promising modality for designing useful indoor localization services for off-the-shelf smartphones.

To unleash the fingerprint approach from the blanket labeled training data collection process, this thesis aims to design a simultaneous localization and mapping (SLAM) system based on the smartphone's inertial measurement unit (IMU) data and the acoustic echoes. Specifically, when the user carrying the smartphone moves in the space, both IMU data and acoustic echo data are collected. When the user returns to a location visited earlier, the movement trajectory produces a *loop closure*.

If the loop closures can be accurately detected using the echo data, the IMU-based dead reckoning result, which is deviation-prone, can be rectified to yield a more accurate trajectory reconstruction. Accordingly, the reconstructed trajectory and the associated echo fingerprints form a *trajectory map*. The trajectory map only contains the echoes from the space covered by the trajectory. The extension to the whole indoor space relies on the *map superimposition* of many trajectory maps.

As a key to SLAM, loop closure detection requires an effective embedding such that any two acoustic echo samples captured at the same location are close in the embedded space, while those collected at different locations are apart. However, finding an effective embedding by which a certain similarity metric can effectively signal loop closures is challenging. The tests show that the generic acoustic features, e.g., power spectral density, spectrogram, and principal component analysis, cannot effectively discriminate locations. Thus, an alternative approach is learning an effective embedding. A straw-man proposal is to train a DNN using labeled echo data collected at sparse locations and use the output of a hidden layer as the embedding. However, such embedding for recognizing the pre-defined locations generally becomes ineffective for the locations out of the training dataset such as the arbitrary locations on the user's trajectory in SLAM. Contrastive learning (CL), on the other hand, exploits the self-supervised learning technique to learn effective representations from unlabeled data. Applied to the SLAM problem, it can learn from the unlabeled echo data collected on the user's trajectory and only requires the information of whether two echo samples are collected at close locations. Such coarse proximity information can be easily derived using heuristics. Thus, this thesis applies CL with a cosine similarity-based contrastive loss function to train an embedding DNN that outputs a representation called *echoic location feature* (ELF). As such, the cosine similarity between any two echo samples' ELFs effectively signals whether the two samples are collected at the same/close locations (i.e., loop closure).

To realize the ELF-based SLAM, this thesis makes the following three designs. First, a trajectory-level CL procedure is designed to learn the trajectory-specific ELFs for loop closure detection. It consists of the *pre-training* step, which uses extensive synthetic echoes to build a basic ELF extractor, and the *fine-tuning* step, which adapts the basic extractor to a target indoor space using unlabeled echoes collected on the user trajectory. Second, a clustering-based approach is designed to curate the loop closures. This is because, although the similarity values among the ELFs can signal most of the true positive loop closures, they can also cause false positives. The curation exploits prior knowledge about the user's movement to effectively remove the false positives. Third, a floor-level CL procedure is designed to superimpose the trajectory maps from many users to form a single *floor map*. The procedure reconciles the differences among the ELFs from different trajectory maps at the same location due to the non-negligible impact of smartphone orientation on echo data. Lastly, the echoes are used to estimate the wall distances and then the estimated distances and the rectified user trajectory are leveraged for room geometry reconstruction.

Extensive experiments show that ELF-based SLAM achieves median localization errors of 0.1 m, 0.53 m, and 0.4 m on the reconstructed trajectories in a living room, an office, and a shopping mall, and outperforms the Wi-Fi and geomagnetic SLAM systems.

1.5 Summary of Contributions

The contributions of this thesis are summarized as follows:

- Label scarcity is identified as the main challenge for developing machine learning algorithms for AIoT sensing applications. This thesis proposes to exploit the sensor and process characteristics to address the label scarcity challenge.

- This thesis proposes PhyAug to use first principles to address the run-time domain shifts in AIoT sensing. Specifically, it exploits both the sensor and process characteristics to generate augmented target-domain data to transfer a DNN model. This approach is more efficient than the physics-regardless transfer learning in terms of target-domain data sampling complexity. By using a small amount of data for first principle fitting, this thesis can avoid collecting substantial training data from an individual target domain. The data and source code for the case studies are made publicly available.[1]

[1]`https://github.com/jiegev5/PhyAug`

- This thesis presents ELF-SLAM, a smartphone-based SLAM system based on the learned echoic location features (ELFs). To address the label scarcity challenge, this thesis exploits the process characteristics of data collection and designs a customized contrastive learning procedure to learn ELFs. The learning of ELFs does not require information but a few minutes of echo data. Extensive experiments conducted in a living room, an office, and a shopping mall show that ELF-SLAM achieves sub-meter mapping and localization accuracy and outperforms WiFi and geomagnetic SLAM.
- This thesis further discusses the potential approaches of integrating known prior knowledge to improve the efficiency of machine learning algorithms in the field of AIoT sensing. The future works of this thesis include (1) exploring methods based on learning and/or inductive bias for knowledge integration, (2) exploring the approach of using DNN to model intricate physical laws, and (3) exploring physics-informed parameter tuning for online model transfer.

1.6 Outline of the Thesis

The rest of this thesis is organized as follows. Chapter 2 presents PhyAug that exploits the sensor and process characteristics to address the run-time domain shifts in AIoT sensing applications. First, it presents the approach overview and related work. Then, the chapter presents the effectiveness of the proposed approach using five case studies with distinct objectives. Chapter 3 presents ELF-SLAM, a smartphone-based indoor SLAM system based on learned ELFs. First, it presents the preliminary study of using acoustic echo for indoor location sensing. Then, the overview of ELF-SLAM is presented. The design of contrastive learning procedures to learn ELFs is further discussed. Lastly, the performance of ELF-SLAM is evaluated in three different indoor environments. Chapter 4 concludes the thesis and discusses future work directions.

Chapter 2

PhyAug: Physics-Directed Data Augmentation for Deep Sensing Model Transfer in Cyber-Physical Systems

This chapter presents PhyAug: Physics-directed data augmentation for deep sensing model transfer in cyber-physical systems [1]. The organization of this chapter is as follows. §2.1 overviews the PhyAug approach. §2.2 reviews related work. §2.4 presents the case study of keyword spotting, §2.5 presents the case study of automatic speech recognition, §2.6 presents the case study of acoustic room recognition, §2.7 present the case study of fisheye image recognition, and §2.8 present the case study of seismic localization, §2.9 discusses several issues of PhyAug. §2.10 summaries this chapter.

2.1 Approach Overview

The design of the DNN used for a sensing task is often driven by a training dataset that is either publicly available or collected by the system designer. However, when the DNN is deployed, the actual distribution of the inference data samples

[1]This chapter is partially published in [34, 35]

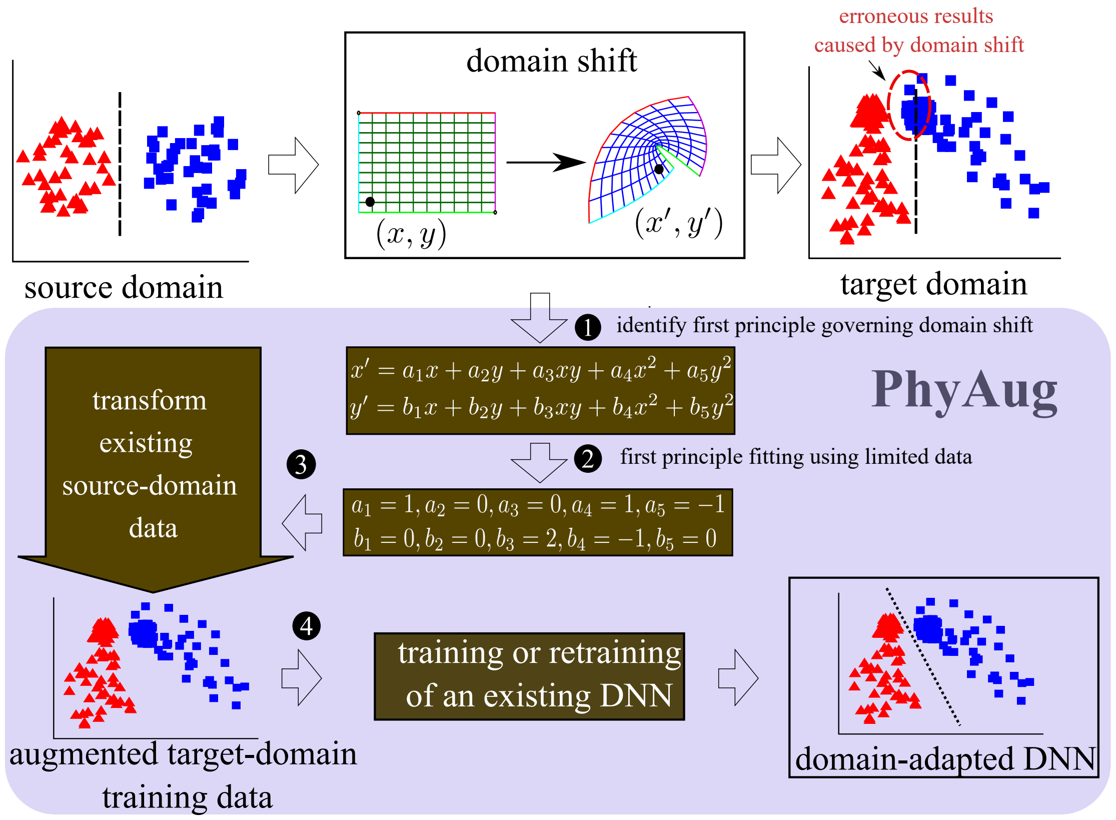

erroneous results caused by domain shift
domain shift
(x, y)
(x', y')
source domain
target domain
❶ identify first principle governing domain shift
$x' = a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$
$y' = b_1x + b_2y + b_3xy + b_4x^2 + b_5y^2$
PhyAug
❷ first principle fitting using limited data
$a_1 = 1, a_2 = 0, a_3 = 0, a_4 = 1, a_5 = -1$
$b_1 = 0, b_2 = 0, b_3 = 2, b_4 = -1, b_5 = 0$
❸ transform existing source-domain data
❹
augmented target-domain training data
training or retraining of an existing DNN
domain-adapted DNN

FIGURE 2.1: PhyAug workflow.

may be different from that of the training dataset. For instance, the sensors used for collecting the training and inference data samples can have different characteristics caused by hardware model differences and manufacturing imperfections across sensors of the same model. The basic principle of PhyAug is to obtain the sensor characteristics using low-overhead approaches and then learn the mapping from the source-domain sensor to the target-domain sensor. After the mapping is learned, one may convert the training data collected by the source-domain sensor to augmented data that are consistent with the target-domain sensor. As a result, the deep model retrained using the augmented data can work effectively on the data collected by the target-domain sensor.

Fig. 2.1 illustrates PhyAug's workflow using a simple example, where the DNN performs two-class classification based on two-dimensional data samples and the first principle governing the domain shift is a nonlinear polynomial transform. Such transform can be used to characterize camera lens distortion [36]. To simplify the discussion, this example considers class-independent domain shift, i.e., the transform is identical across all the classes. Note that PhyAug can deal with class-dependent domain shifts, which will be discussed later. The general transfer learning approaches regardless of the first principles need to draw substantial data samples from both the source and target domains. Then, they apply *domain shift learning* techniques to update the existing source-domain DNN or construct a prefix DNN [10] to address the domain shift. Extensive data collection in the target domain often incurs undesirable overhead in practice.

Differently, as shown in Fig. 2.1, PhyAug applies the following four steps to avoid extensive data collection in the target domain. (1) The system designer identifies the parametric first principle governing the domain shift. For the current example, the parametric first principle is $x' = a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$ and $y' = b_1x + b_2y + b_3xy + b_4x^2 + b_5y^2$, where (x, y) and (x', y') are a pair of data samples in the source and target domains, respectively, and a_i, b_i are unknown parameters. (2) A small amount of unlabeled data pairs are collected from the source and target sensors to estimate the parameters of the first principle. For this example, if the domain shift is perturbation-free, the minimum number of data pairs needed is the number of unknown parameters of the polynomial transform. If the domain shift is also affected by other unmodeled perturbations, more data pairs can be drawn to improve the accuracy of estimating the parameters under a least squares formulation. If the domain shift is class-dependent, the data pair sampling and parameter estimation should be performed for each class separately. (3) All the existing source-domain training data samples are transformed to the target domain using the fitted first principle, forming an augmented training dataset in the target domain. (4) With the augmented training dataset, various techniques can be employed to transfer the existing DNN built in the source domain to the target domain. For instance, the DNN model can be retrained with augmented data. The retraining can use the existing DNN as the starting point to speed up the process. For instance, for the DeepSpeech2 [18] which is a large-scale ASR model used in §2.5, the retraining only requires half of the training time compared with the training from scratch using the augmented data.

For sensing DNN design, the source domain is in general the design dataset. In such a case, the source domain cannot be excited anymore for data pair sampling in both domains simultaneously. However, the excitation can be recreated to collect the corresponding target-domain samples. For instance, a speaker can be used to play voice samples from the source-domain dataset and collect the corresponding samples from a target-domain microphone. Similarly, image samples from the source-domain dataset can be displayed and the corresponding samples can be collected using a target-domain camera that may have optical distortions.

2.2 Related Work

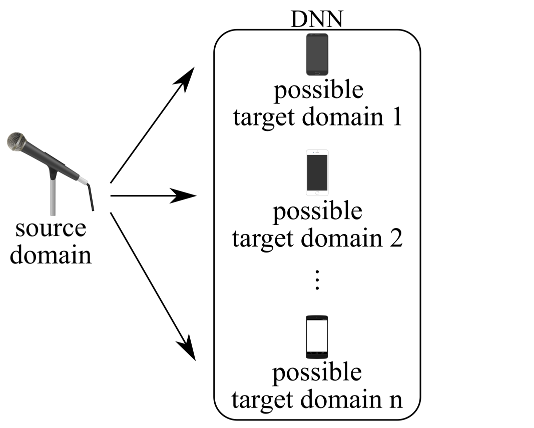
The applications of deep learning in embedded sensing systems have obtained superior inference accuracy compared with heuristics and conventional machine learning. Various approaches have been proposed to address the domain shift problems in embedded sensing [7, 10, 17, 37] and image recognition [15, 16, 39]. Table 2.1 summarizes the categorization, used techniques, and requirements of these approaches. In what follows, the important details of these approaches and their differences from PhyAug are discussed.

■ **Domain adaptation:** Few-shot Adversarial Domain Adaptation (FADA) [15] transfers the model with a limited amount of target-domain training data. It uses the *supervised adversarial learning* technique to find a shared subspace of the data distributions in the source and target domains. FADA requires labeled and paired data samples from both the source and target domains. Adversarial Discriminative Domain Adaptation (ADDA) [16] uses *unsupervised adversarial learning* to learn a feature encoder for the target domain. Although ADDA requires neither class labels nor data pairing, it demands substantial unlabeled target-domain data. TransAct in [17] considers sensor heterogeneity in human activity recognition and uses unsupervised adversarial learning to learn stochastic features for both domains. It requires hundreds of unlabeled target-domain data samples. Mic2Mic [10] applies CycleGAN, which is also an adversarial learning technique, to map the target-domain audio recorded by a microphone "in the wild" back to the source-domain microphone for which the DNN is trained. Mic2Mic requires about 20 minutes of speech recording from both microphones, which represents an overhead. It performs one-to-one translations. Experiment results in §2.4 and §2.5 show that CycleGAN may underperform when the source domains are publicly available speech datasets collected using numerous microphones in diverse environments. Cross-domain contrastive learning (CDCL) exploits contrastive self-supervised learning for domain adaptation. Specifically, it minimizes the distance of the cross-domain data samples from the same classes via contrastive loss, such that the trained model is invariant

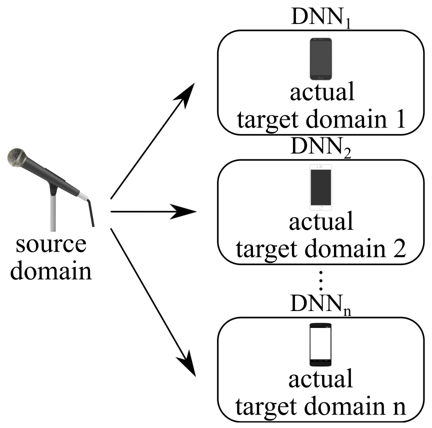
TABLE 2.1: Categorization, used techniques, and requirements of various solutions to address domain shifts.

Category	**Used technique**	**Solution**	**Applications in publication**	**Requirements**				
				Source domain label	**Target domain label**	**Paired label data**	**First principle**	**Target domain data volume***
Domain adaptation (Model transfer)	Adversarial learning	FADA [15]	computer vision	Yes	Yes	Yes	–	low
		ADDA [16]	computer vision	–	–	–	–	high
		TransAct [17]	activity sensing	–	–	–	–	medium
		Mic2Mic [10]	voice sensing	–	–	–	–	high
	Meta learning	MetaSense[37], Onefi [38]	voice & motion	Yes	Yes	–	–	low
	Contrastive learning	CDCL [39]	computer vision	Yes	–	–	–	medium
	Data augmentation	**PhyAug**	voice sensing	–	–	–	Yes	low
			room recognition	–	–	–	Yes	low
			computer vision	–	–	–	Yes	low
Model robustness	Data augmentation	CDA [7]	voice and activity sensing	–	–	–	Yes	medium

* The bars represent oracle scales partially based on the reported numbers in respective publications. Fully comparable scales are difficult to obtain because the solutions are designed for different applications. PhyAug is compared with FADA, Mic2Mic, and CDA in the evaluation sections of this thesis. Reasons for excluding other approaches from the comparison will be discussed in the case studies.



(A) Data augmentation for model robustness (e.g.,[7]).



(B) Data augmentation for model transfer (PhyAug).

FIGURE 2.2: Different purposes of data augmentation illustrated using voice sensing. Note that the source domain may contain many microphones used to collect training samples.

to the domain difference. As shown in the §2.4, this approach requires a significant amount of target domain data to achieve a satisfactory result. PhyAug is a domain adaptation approach. Compared with ADDA [16], TransAct [17], and Mic2Mic [10] that are based on unsupervised adversarial learning and thus require substantial target-domain training data, PhyAug exploits the first principle governing the domain shift to reduce the demand on target-domain data. Although FADA [15] aims at reducing the demand for target-domain data, it requires extensive other information such as class labels in both domains. In contrast, PhyAug requires unlabeled data only. Different from Mic2Mic [10] that requires the source domain to be a single microphone, PhyAug admits a source-domain dataset collected via many (and even unknown) microphones in the KWS and ASR case studies. This enlarges the application scope because the datasets used to drive the design of DNNs for real-world applications often consist of recordings from diverse sources.

MetaSense [37] and Onefi [38] employ meta-learning technique [40–43] to rapidly adapt a model to the target user's condition with few shots. Specifically, MetaSense uses data collected from multiple source domains to train a base model that can adapt to a target domain related to the source domains. However, it requires substantial training data from both domains and class labels from each source domain. For voice sensing, MetaSense cannot use unlabeled source-domain datasets collected via many microphones. But PhyAug can.

■ **Model robustness via data augmentation:** Data augmentation has been widely adopted for enhancing model robustness. As illustrated in Fig. 2.2a, a conventional scheme presumes a number of domain shifts (e.g., scaling, rotation, noise injection, etc) and follows them to generate augmented training samples. Then, the original and the augmented data samples are used to train a single DNN. During the serving phase, this DNN remains robust to the domain shift resembling the presumption. However, should the actual domain shift be out of the presumption, the robustness is lost. The study [7] adopts the above conventional data augmentation (CDA) approach to mitigate the impact of sensor heterogeneity on DNN's accuracy. Specifically, it estimates the probability distribution of sensors' heterogeneity characteristics from a *heterogeneity dataset* and then uses the characteristics sampled from the estimated distribution to generate augmented training data. As the dataset needs to cover heterogeneity characteristics, its collection in practice incurs a considerable overhead. The heterogeneity dataset in [7] consists of 2-hour recordings of 20 different microphones placed equidistant from an audio speaker. If the characteristic of a microphone "in the wild" is out of the estimated characteristic distribution (i.e., a missed catch), the enhanced DNN may not perform well. Since CDA uses sensor characteristics, it can be viewed as an approach directed by first principles. Different from CDA's objective of enhancing model robustness, PhyAug uses data augmentation to transfer a model to a specific target domain (i.e., sensor). Fig. 2.2b illustrates this in the context of voice sensing, where microphones' unique characteristics create domains. PhyAug constructs a dedicated DNN for each target domain. Thus, PhyAug is free of the missed catch problem.

2.3 Methodology

As this thesis proposes PhyAug which is a domain adaptation approach, it is desirable to show PhyAug's applicability to multiple applications and its scalability to address different levels of pattern sophistication. Therefore, PhyAug is applied to different applications, i.e., KWS, ASR, ARR, FIR and seismic localization. Although KWS and ASR are two specific human voice sensing tasks, they have significantly different complexities. ARR is a mobile sensing application. The benchmark results presented in this thesis show that ARR performance is greatly affected by smartphone heterogeneity. FIR is a visual sensing application where the fisheye camera produces non-linear distortion on the captured images, causing domain shift from the pinhole camera. Different from the aforementioned case studies where domain shifts are mainly caused by sensor characteristics, seismic event localization concerns the domain shift caused by variations of the monitored physical process. For each case study, PhyAug is compared with multiple existing approaches to show the advantages and performance gains of PhyAug.

2.4 Case Study 1: Keyword Spotting (KWS)

Human voice sensing is important for human-computer interactions in many Internet of Things (IoT) applications. At present, the DNN for a specific human voice sensing task is often trained based on a *standard dataset*. However, as IoT microphones are often of small form factors and low cost, their recordings often suffer degraded and varied voice qualities. In addition, the environment that an IoT microphone resides in can also affect its recording. For instance, the echo patterns in indoor spaces of different sizes can be distinct. Such run-time variations may be poorly captured by the standard dataset. As a result, the DNN yields reduced accuracy after the deployment.

In this thesis, two human voice sensing functions are considered: KWS and ASR. PhyAug is applied to address the domain shift problem. Specifically, it starts with a swift process of profiling the IoT microphone's frequency response curve (FRC) with the help of a smartphone. Then, the FRC is used to transform the standard dataset. Finally, the DNN is retrained using the transformed dataset to obtain a personalized DNN for the IoT microphone.

In the case studies of KWS (§2.4) and ASR (§2.5), **source domain** is the standard dataset originally used to train the DNN; **target domain** is the dataset of voice samples captured by a specific deployed microphone; **first principle** is the microphone's FRC induced by the microphone hardware and its ambient environment.

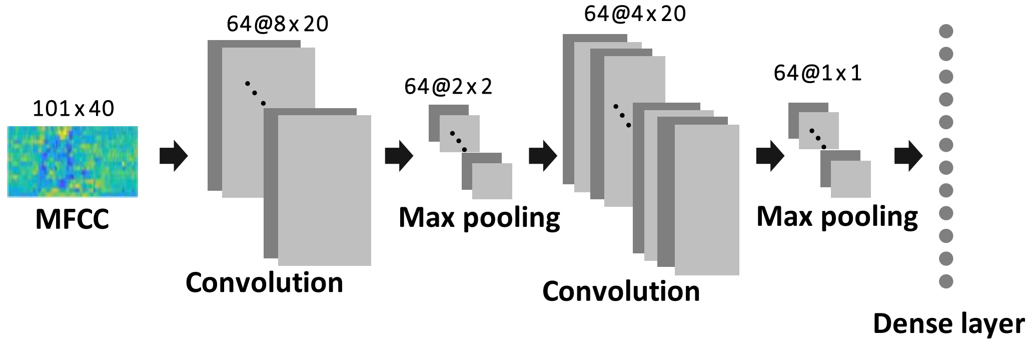


FIGURE 2.3: CNN structure used in KWS case study.

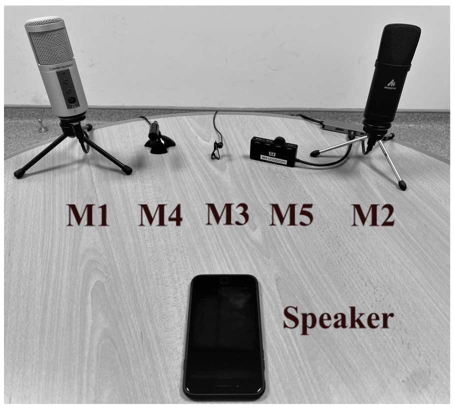


FIGURE 2.4: Microphones & experiment setup.

2.4.1 Problem Description

A set of preliminary experiments is conducted to investigate the impact of diverse microphones on KWS accuracy. Based on the results, the problem is stated.

2.4.1.1 Standard dataset and DNN

Google Speech Commands Dataset [44] is used as the standard dataset in this case study. It contains 65,000 one-second utterances of 30 keywords collected from thousands of people. Audio files are sampled at 16-kilo samples per second (ksps). The voice samples are pre-processed as follows. First, a low-pass filter (LPF) is applied with a cutoff frequency of 4 kHz on each voice sample, because the human voice's frequency band ranges from approximately 0.3 kHz to 3.4 kHz. Then, for each filtered voice sample, 40-dimensional Mel-Frequency Cepstral Coefficients (MFCC)

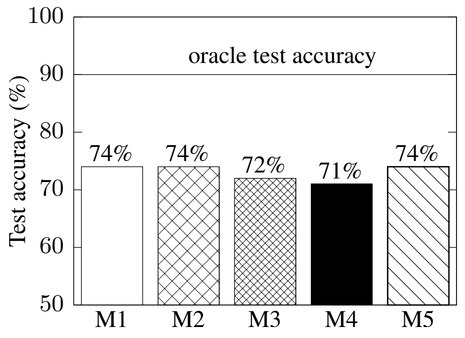


FIGURE 2.5: KWS accuracy on microphones. Horizontal line is accuracy on standard dataset.

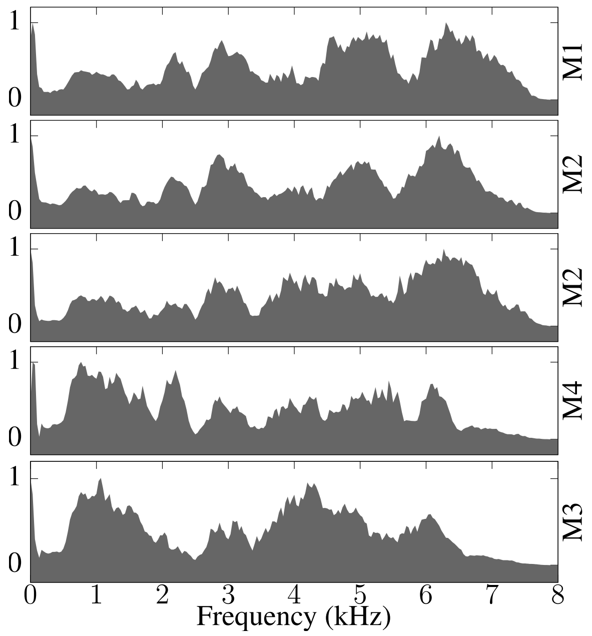


FIGURE 2.6: The five microphones' FRCs. The y-axis of each sub-figure is normalized amplitude.

frames are generated using a 30-millisecond window size and a 10-millisecond window shift. The z-score normalization is applied on each MFCC frame. Eventually, each voice sample is converted to a 101×40 MFCC tensor. The dataset is randomly split into training, validation, and testing sets following an 8:1:1 ratio.

A CNN is implemented to recognize 10 keywords, i.e., "yes", "no", "left", "right", "up", "down", "stop", "go", "on", and "off". Two more classes are also added to represent *silence* and *unknown keyword*. Fig. 2.3 shows the structure of the CNN. It achieves 90% test accuracy, which is similar to that in [45] and referred to as the *oracle test accuracy*.

2.4.1.2 Impact of microphone on KWS performance

This section demonstrates that the CNN has performance degradation as a result of microphone heterogeneity. The CNN is tested on samples captured by five different microphones named M1, M2, M3, M4, and M5 as shown in Fig. 2.4 that have list prices from high ($80) to low ($3.5). M1 and M2 are two high-end desktop cardioid condenser microphones, supporting sampling rates of 192 ksps at 24-bit depth and 48 ksps at 16-bit depth, effective frequency responses of [30 Hz, 16 kHz] and [30 Hz, 15 kHz], respectively. M3 is a portable clip-on microphone with an effective frequency response range of [20 Hz, 16 kHz]. M4 and M5 are two low-cost mini microphones without detailed specifications. Fig. 2.4 shows the placement of the microphones. For fair comparison and result reproducibility, An Apple iPhone 7 is used to play the original samples of the test dataset through its loudspeaker, with all microphones placed at equal distances away.

The samples recorded by each microphone are fed into the KWS CNN for inference. Fig. 2.5 shows the test accuracy for each microphone. Compared with the oracle test accuracy of 90%, there are 14% to 19% absolute accuracy drops due to domain shifts. By inspecting the spectrograms of the original test sample and the corresponding ones captured by the microphones, the differences can be observed. This explains the distinct accuracy drops among microphones. From the above experiment results, the research questions addressed in this case study are as follows. First, how to profile the characteristics of individual microphones with low overhead? Second, how to exploit the profile of a particular microphone to recover KWS's accuracy?

2.4.2 PhyAug for Keyword Spotting

PhyAug for KWS consists of two procedures: *fast microphone profiling* and *model transfer via data augmentation*.

2.4.2.1 Fast microphone profiling

A microphone can be characterized by its frequency response consisting of magnitude and phase. Only the magnitude component is considered, because the information of a voice signal is largely represented by the energy distribution over frequencies, with little/no impact from the phase of the voice signal in the time domain. Let $X(f)$ and $Y(f)$ denote the frequency-domain representations of the considered microphone's input and output. The FRC to characterize the microphone is $H(f) = \frac{|Y(f)|}{|X(f)|}$, where $| \cdot |$ represents the magnitude.

A fast microphone profiling approach that estimates $H(f)$ in a short time is proposed. It can be performed through a factory calibration process or by the user after the microphone is deployed. Specifically, a loudspeaker placed close to the target microphone emits a band-limited acoustic white noise $n(t)$ for a certain time duration. The frequency band of the white noise generator is set to be the band that is desired to be profiled. Meanwhile, the target microphone records the received acoustic signal $y_n(t)$. Thus, the FRC is estimated as $H(f) = \frac{|\mathcal{F}[y_n(t)]|}{|\mathcal{F}[n(t)]|}$, where $\mathcal{F}[\ \cdot\ ]$ represents the Fourier transform. As $n(t)$ has a nearly constant power spectral density (PSD), this approach profiles the microphone's response at all frequencies in the given band. In experiments, the iPhone 7 shown in Fig. 2.4 is used to emit the white noise. The frequency band of the noise generator is set to be $[0, 8\,\text{kHz}]$, which is the Nyquist frequency of the microphone. Fig. 2.6 shows the measured FRCs of the five microphones used in experiments. Each FRC is normalized to $[0, 1]$. The microphones exhibit distinct FRCs. In addition, the two low-end microphones M4 and M5 are observed to have lower sensitivities to the higher frequency band, i.e., 5 kHz to 8 kHz, compared with the microphones M1, M2, and M3.

2.4.2.2 Model transfer via data augmentation

Training samples are augmented in the target microphone's domain by transforming the original training samples using FRC. The procedure for transforming a sample $x(t)$ is as follows: (1) Apply the pre-processing LPF on $x(t)$ to produce $x'(t)$; (2) Conduct short-time Fourier transform using 30-millisecond sliding windows with an offset of 10 milliseconds on $x'(t)$ to produce 101 Fourier frames, i.e., $X_i(f)$, $i = 1, 2, \ldots 100$; (3) Multiply the magnitude of each Fourier frame with the

FRC to produce $|Y_i(f)| = H(f) \cdot |X_i(f)|$; (4) Generate the MFCC frame from each PSD $|Y_i(f)|^2$; (5) Concatenate all 101 MFCC frames to form the MFCC tensor. Lastly, PhyAug retrains the CNN with augmented data samples for the microphone using the pre-trained CNN as the starting point.

2.4.3 Performance Evaluation

2.4.3.1 Alternative approaches

The performance evaluation employs the following alternative approaches.

■ **Data calibration:** At run time, it uses the measured FRC to convert the target-domain data back to the source-domain data and then applies the pre-trained CNN on the converted data. Specifically, let $Y_i(f)$ denote the ith Fourier frame after the microphone applies the LPF and short-time Fourier transform on the captured raw data. Then, it estimates the corresponding source-domain PSD as $|X_i(f)|^2 = \left(\frac{|Y_i(f)|}{H(f)}\right)^2$ and generates the MFCC frame from $|X_i(f)|^2$. The MFCC tensor concatenated from the MFCC frames over time is fed to the pre-trained CNN.

■ **Conventional data augmentation (CDA) [7]:** This alternative captures the essence of the approach in [7] following the conventional data augmentation scheme illustrated in Fig. 2.2a. Specifically, one out of the five microphones, e.g., M1, is designated as the testing microphone. The remaining four, e.g., M2 to M5, are used to generate a *heterogeneity dataset* [7]. The *heterogeneity generator* [7] is constructed as follows. For each microphone in the heterogeneity dataset, FRC is measured multiple times with the fast profiling process. At any frequency f, the FRC value is modeled by a Gaussian distribution. A Gaussian mixture is formed by the four heterogeneity-dataset microphones' Gaussian distributions with equal weights. The Gaussian mixtures for all frequencies form the heterogeneity generator. Then, each source-domain training sample is transformed by an FRC sampled from the heterogeneity generator into an augmented sample. Lastly, the DNN is retrained with the augmented training samples and tested with the samples captured by the testing microphone.

■ **CycleGAN (essence of [10]):** Mic2Mic [10] trains a CycleGAN using unlabeled and unpaired data samples collected from two microphones A and B. Then, CycleGAN can translate a sample captured by A to the domain of B, or vice versa. Following [10], a CycleGAN is trained to translate the samples captured by a target microphone to the source domain of Google Speech Commands Dataset. To measure the test accuracy, a test sample collected by a microphone is converted by the corresponding CycleGAN to the source domain and fed into the pre-trained CNN.

Compared with PhyAug that requires a single 5-second profiling data collection process for each microphone, CDA repeats the profiling process many times for each heterogeneity microphone to construct the heterogeneity generator; the training of CycleGAN requires 15 minutes of data collected from each target microphone. Thus, both alternative approaches have higher overheads.

■ **FADA** [**15**]: It trains a feature encoder and classifier in the source domain. Then, it combines source-domain and target-domain data to train a domain-class discriminator. Finally, the weights of the feature encoder and classifier are updated to the target domain through adversarial learning using the domain-class discriminator. To apply FADA for KWS, the architecture in [15] is used and the KWS model in Fig. 2.3 is modified by adding a fully-connected layer before the last dense layer. Thus, the model has a feature encoder (CNN layers) and a classifier (fully-connected layers).

■ **CDCL** [**39**]: CDCL comprises three steps for domain adaptation. First, a domain-invariant feature encoder is trained to minimize the distance between the source-domain data and the target-domain data via the contrastive learning. The procedure in [39] is followed and the positive and negative data samples are constructed as follows. The data samples from the different domains but in the same class are viewed as positive samples. The data samples that are in different classes from the same or different domains are viewed as negative samples. Second, the trained feature encoder is freezed and applied to the labeled source-domain data to train a classifier. Third, the trained feature encoder and the classifier are applied to the target-domain data to evaluate the domain adaptation performance. In this paper, the used feature encoder is a ResNet-18 model and the classifier is a multi-layer perceptron (MLP) consisting of 4 layers. The numbers of neurons in the four layers of the MLP are 512, 1024, 1024 and 12.

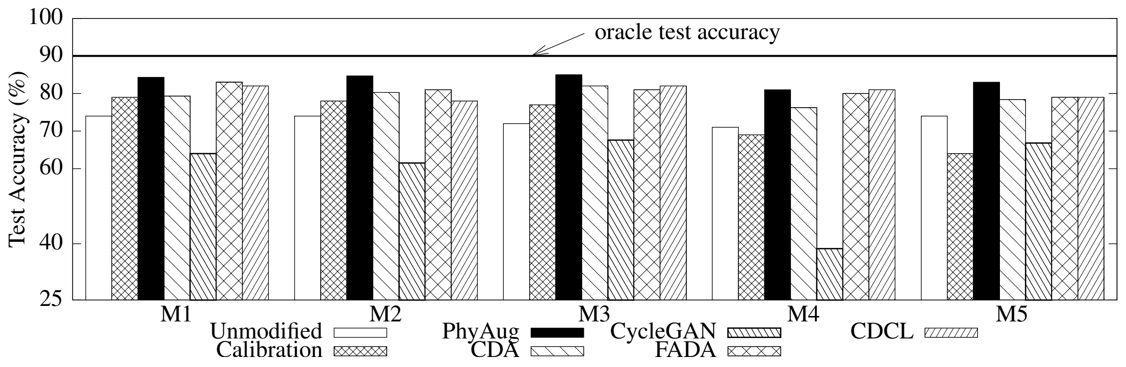


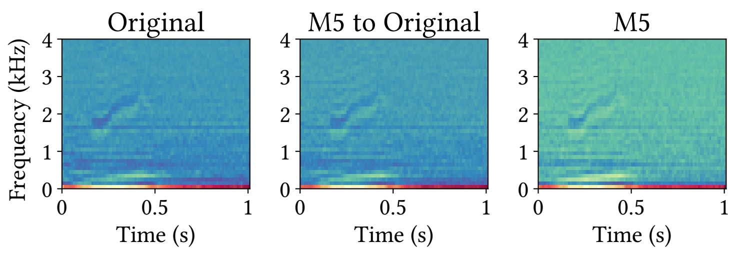
FIGURE 2.7: KWS test accuracy using various approaches on tested microphones. Compared with the unmodified baseline, PhyAug recovers the accuracy losses by 64%, 67%, 72%, 53%, and 56% respectively for the five microphones toward the oracle test accuracy.

The MetaSense, ADDA and TransAct reviewed in §2.2 are excluded from the baselines for the following reasons. MetaSense cannot be applied to the unlabeled source-domain dataset collected via many microphones. Unsatisfactory results are obtained for ADDA in the adversarial training with hours of target-domain training data and extensive hyperparameter tuning. It is suspected that the amount of target-domain training data is still insufficient for ADDA. Note that PhyAug only requires five seconds of unlabeled target-domain data as shown shortly. TransAct is customized for activity recognition that differs from human voice sensing.

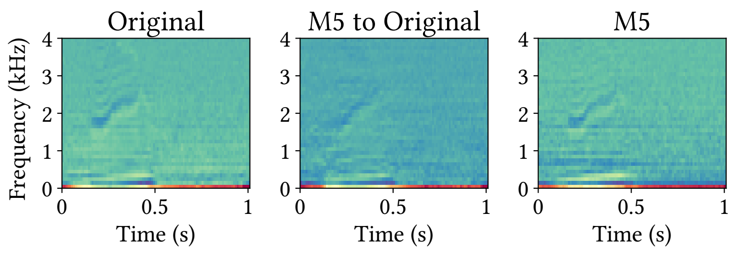
2.4.3.2 Evaluation results

PhyAug and the alternatives are applied for the five microphones in Fig. 2.4. The test accuracies are shown in Fig. 2.7. The bars labeled "unmodified" are the results from Fig. 2.5, for which no domain adaptation technique is applied. They are included as the baseline. The results are explained in detail as follows.

■ **Data calibration:** It brings test accuracy improvements for M1, M2, and M3. The average test accuracy gain is about 4%. For the cheap microphones M4 and M5, it results in test accuracy deteriorations. The reason is as follows. Its back mapping uses the reciprocal of the measured FRC (i.e., $1/H(f)$), which contains large elements due to the near-zero elements of $H(f)$. The larger noises produced by the low-end microphones M4 and M5 are further amplified by the large elements of $1/H(f)$, resulting in performance deteriorations. Thus, although this approach may bring performance improvements, it is susceptible to noises.



(A) M5 $\longrightarrow$ M1



(B) M5 $\longrightarrow$ Original

FIGURE 2.8: CycleGAN translation results (mid column). (a) Translation from M5 to M1. High similarity between first and second columns shows effectiveness of CycleGAN. (b) Translation from M5 to the domain of Google Speech Commands Dataset. Dissimilarity between first and second columns shows ineffectiveness of CycleGAN.

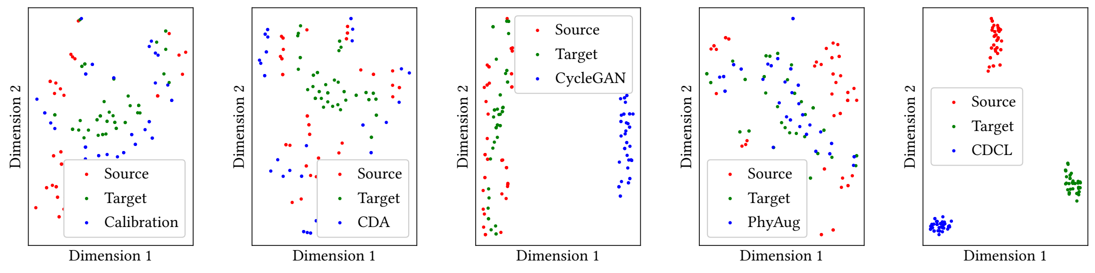
■ **PhyAug:** The black bars in Fig. 2.7 show PhyAug's results. Compared with the unmodified baseline, PhyAug recovers the test accuracy losses by 64%, 67%, 72%, 53%, and 56% for the five microphones. PhyAug cannot fully recover the test accuracy losses. This is because PhyAug only addresses the deterministic distortions due to microphones; it does not address the other stochastic factors such as the environmental noises and the microphones' thermal noises.

■ **CDA:** It recovers certain test accuracy losses for all microphones. This is because for any target microphone, there is at least one heterogeneity dataset microphone giving a similar FRC as the target microphone. Specifically, from Fig. 2.6, M1, M2, and M3 exhibit similar FRCs; M4 and M5 exhibit similar FRCs (i.e., they have good responses in lower frequencies). However, PhyAug consistently outperforms CDA. In addition, CDA introduces larger overhead than PhyAug as discussed in §2.4.3.1.

■ **CycleGAN:** It leads to test accuracy deteriorations for all five target microphones. Although CycleGAN is effective in translating the domain of a microphone to that of another microphone, which is the basis of Mic2Mic [10]. Howerver, CycleGAN is ineffective in translating a certain microphone to the source domain of a dataset that consists of recordings captured by many microphones. This is illustrated using an example of CycleGAN translated audio spectrogram. First, a CycleGAN is trained to translate M5 to M1. The first and the third columns of Fig. 2.8a show the spectrograms captured by M1 and M5 for the same sample played by the smartphone in the setup shown in the paper. It is observed that there are discernible differences. The mid column shows the output of the CycleGAN, which is very similar to the first column. This result suggests that CycleGAN is effective for device-to-device domain translation. Then, the same approach is applied to train a different CycleGAN to translate M5 to the domain of the Google Speech Commands Dataset. Fig. 2.8b shows the results. The third column is the spectrogram captured by M5 when a dataset sample shown in the first column is played by the smartphone in the setup shown in the paper. The mid column is CycleGAN's translation result, which has discernible differences from the first column, suggesting the ineffectiveness of CycleGAN. An intuitive explanation is that the CycleGAN shown with samples captured by many microphones during the training phase is confused and caters into no single microphone. Due to the discrepancy between CycleGAN's output and the dataset, the pre-trained CNN fed with CycleGAN's outputs yields low test accuracy.

■ **FADA:** When the number of labeled target-domain samples per class (LTS/C) is set to 10 for FADA training, it recovers the accuracy loss for the five microphones by 56%, 38%, 47%, 47%, and 37%, respectively, as shown in Fig. 2.7. The performance of FADA increases with LTS/C. When LTS/C is increased to 20, PhyAug still outperforms FADA. Note that PhyAug requires a single unlabeled target-domain sample only. In addition, from the experience, FADA is sensitive to hyperparameter setting.

■ **CDCL:** The amount of the used target-domain data per class for contrastive learning is 200. As shown in Fig. 2.7, CDCL recovers the accuracy loss for the five microphones by 50%, 25%, 56%, 53%, and 27%, respectively. However, the performance of CDCL is sensitive to the amount of target-domain data used for contrastive feature learning. PhyAug outperforms CDCL even when number of the used target-domain data samples per class increases up to 400.



(A) $r = 7$ (B) $r = 0.2$ (C) $r = 1$ (D) $r = 10$ (E) $r = 0.4$

FIGURE 2.9: t-SNE visualization of different domain data. The r value reported in the sub-figure caption characterizes the effectiveness of the approach. It is the ratio of the source-translation and target-translation distances.

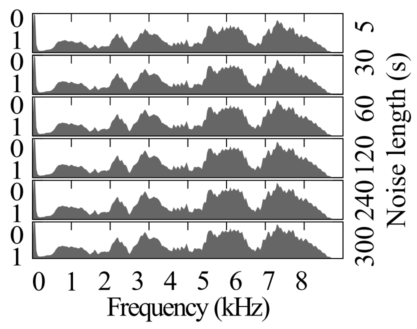
2.4.4 In-depth Analysis

2.4.4.1 Data translation performance of different approaches

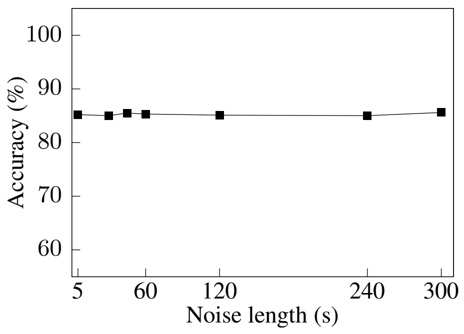
The data translation performance is investigated for each approach using the T-distributed Stochastic Neighbor Embedding (t-SNE) [46]. t-SNE is a dimensionality reduction technique to effectively visualize the high-dimensional data in the low-dimensional feature space. As shown in Fig. 2.9, the red dots labeled "Source" represent the source-domain data, i.e., the original keyword spotting data. The green dots labeled "Target" represent the target-domain data. The target-domain data presented is collected by the microphone M4. The blue dots in each subplot represent the translated data using different approaches. Ideally, the data samples of the same color should cluster together. Moreover, the data translated by an approach from the source-domain data should be close to the target-domain data. To simplify the characterization of the translation effectiveness, a metric $r = \frac{d_s}{d_t}$ is defined, where d_s is the average distance between the source-domain data points and the corresponding translated data points in the t-SNE space and d_t is the average distance between the target-domain data points and the corresponding translated data points. If the value of r is less than 1, the translated data is closer to the source-domain data; otherwise, the translated data is closer to the target domain. Fig. 2.9d shows the data translation performance for PhyAug. PhyAug applies the learned FRC to translate the source-domain data to the target domain. Thus, the translated data via PhyAug is expected to be closer to the target-domain data. The distance ratio r for PhyAug is 10, indicating that the translated data via PhyAug is closer to the target-domain data. Thus, when the model trained using the translated data is applied to the target-domain data, performance improvement is obtained. Fig. 2.9a shows the data translation performance for the Data calibration approach. Data calibration uses the learned FRC to calibrate the target-domain data back to the source domain. Thus the translated data via Data calibration is expected to be closer to the source domain. The distance ratio r for Data calibration is 7, indicating that the calibrated data are closer to the target-domain data. When the trained DNN from the source-domain data is applied to the calibrated data, the performance improvement is limited. Fig. 2.9b shows the data translation performance for CDA. CDA uses a *heterogeneity generator* to construct the augmented data that can contain the target-domain data. Thus the augmented data is expected to be close to the target-domain data. The distance ratio r for CDA is 0.2, indicating that the augmented data are closer to the source-domain data. When the DNN trained using the augmented dataset is applied to the target-domain data, the improvement is limited. Fig. 2.9c shows the data translation performance for CycleGAN. CycleGAN trains a data translation model that tries to map the data between the source domain and the target domain. As the data is translated from the target domain, the translated data is expected to be closer to the source domain. However, it is observed that the translated data are far from both the source-domain and the target-domain data. Thus, CycleGAN is observed the performance drop on microphone M4 in the KWS case study. Fig. 2.9e shows the data translation performance for CDCL. This approach applies the contrastive learning to learn the feature representation such that the domain distance between the source-domain data and the target-domain data is minimized. The translated data is expected to be close to the source-domain data. The distance ratio r for CDCL is 0.4, indicating that the translated data is closer to the source-domain data. Thus, the learned feature representation can reduce the domain difference to a certain extent. However, CDCL requires substantial target-domain data in order to train the DNN model. In summary, PhyAug outperforms the competing baselines in terms of the data translation quality, and it achieves the best results for domain adaptation.

2.4.4.2 Amount of target-domain data needed by each approach

In this section, the amount of target-domain data needed by each approach in order to achieve satisfactory performance is investigated.



(A) FRCs of M1 measured with various noise emission times.



(B) PhyAug test accuracy with various noise emission times.

Figure 2.10: The impact of white noise emission time on the effectiveness of PhyAug.

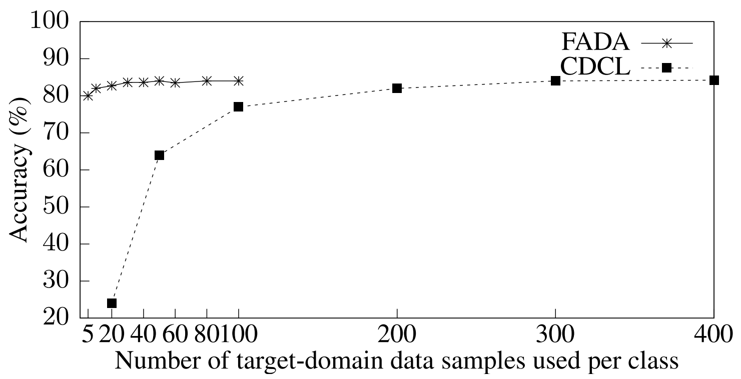


Figure 2.11: Evaluation of the amount of the target-domain data needed for CDCL and FADA.

CDA, Data calibration and PhyAug use white noise to profile the microphones. They do not require target-domain data. In the previous experiments, the microphone profiling uses a 5-minute noise. Experiments are conducted to investigate the impact of shorter noise emission durations on the performance of CDA, Data calibration and PhyAug. Since they use the same FRC to perform data translation, a specific microphone, M1 is used. Fig. 2.10 shows the test accuracy of PhyAug using the M1's FRCs measured with various noise emission times. It is observed that a noise emission time of five seconds is sufficient. This result shows that a minimum of 5-second white noise is sufficient to profile a microphone. Thus, Data calibration and PhyAug incur little overhead. CDA is different from Data calibration and PhyAug as it requires a *heterogeneity generator* to generate augmented training data. The construction of the *heterogeneity generator* requires 10-minute white noise from each microphone.

CDCL, FADA and CycleGAN require both the source-domain and the target-domain data for model training. The performance of CDCL and FADA is investigated when the amount of used target-domain data varies. The plot labeled "CDCL" in Fig. 2.11 shows the CDCL's test accuracy with respect to the used target-domain data. The horizontal axis represents the number of the target-domain data samples used per class; the vertical axis shows the test accuracy. It is observed that CDCL's performance increases when the used target-domain data amount increases. Its performance stabilizes when the number of used data samples per class in the target domain is greater than 200, which is around 5% of the available training data. The plot labeled "FADA" in Fig. 2.11 shows the FADA's test accuracy with respect to the used target-domain data. It is observed that FADA achieves good performance with 20 data samples used in each class, which is around 0.5% of the available training data. Despite that CDCL and FADA only require a small portion of target-domain data to achieve good results. PhyAug is preferred for model transfer as it only requires a short noise emission time and does not require the target-domain data.

2.4.5 Application Considerations

From the above results, PhyAug is desirable for KWS on virtual assistant systems. It is envisaged that more home IoT devices (e.g., smart lights and smart kitchen appliances, etc.) will support KWS. To apply PhyAug, the appliance manufacturer can offer the microphone profiling function as a mobile app and the model transfer function as a cloud service. Thus, the end user can use the app to obtain the FRC, transmit it to the cloud, and receive the customized KWS DNN. As the KWS DNN is not very deep and PhyAug is a one-time effort for each device, the model retraining in the cloud is an acceptable overhead to trade for better KWS accuracy over the entire device lifetime.

2.5 Case Study 2: Automatic Speech Recognition (ASR)

ASR models often have performance degradation after deployments. This section shows the impact of the microphone on ASR and applies PhyAug to mitigate the impact.

2.5.1 Impact of Microphone on ASR

LibriSpeech [47] is used as the standard dataset in this case study. It contains approximately 1,000 hours of English speech corpus sampled at 16 ksps. Each sample is an utterance for four to five seconds. An implementation [18] of Baidu DeepSpeech2 is used, which is a DNN-based end-to-end ASR system exceeding the accuracy of Amazon Mechanical Turk human workers on several benchmarks. The used DeepSpeech2 model is pre-trained with LibriSpeech training dataset and achieves 8.25% word error rate (WER) on LibriSpeech test dataset. This 8.25% WER is referred to as *oracle WER*. Note that the input to DeepSpeech2 is the spectrogram of a LibriSpeech sample, which is constructed from the Fourier frames using 20-millisecond window size and 10-millisecond window shift.

DeepSpeech2 has 11 hidden layers with 86.6 million weights. It is far more complicated than the KWS CNN. Specifically, DeepSpeech2 is 175 times larger than the KWS CNN in terms of the weight amount. All the existing studies (e.g., Mic2Mic [10], MetaSense [37], and CDA [7]) that aimed at addressing domain shift problems in voice sensing only focused on simple tasks like KWS and did not attempt a sophisticated model such as DeepSpeech2.

The performance of the pre-trained DeepSpeech2 is tested on the five microphones M1 to M5 used in §2.4. Same test methodology as presented in §2.4.1.2 is followed. In Fig. 2.12, the histograms labeled "unmodified" represent the WERs of the pre-trained DeepSpeech2 on the test samples recorded by the five microphones. The horizontal line in the figure represents the *oracle WER*. It is observed that the microphones introduce about 15% to 35% WER increases. In particular, the two low-end microphones M4 and M5 incur the highest WER increases. This result is consistent with the intuition. From the above test results, this section investigates

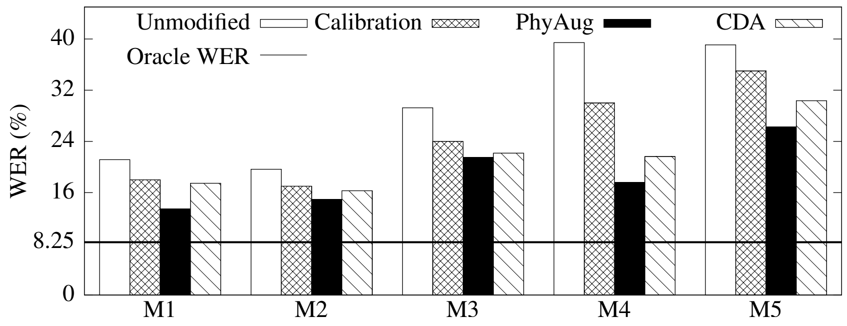


FIGURE 2.12: WERs using various approaches on tested microphones. Compared with the unmodified baseline, PhyAug reduces WER by 60%, 41%, 37%, 70%, and 42% respectively for the five microphones toward the oracle WER. As CycleGAN gives high WERs (about 90%), it is not shown.

whether PhyAug described in §2.4 for KWS is also effective for ASR. Different from the KWS CNN that takes MFCC tensors as the input, DeepSpeech2 takes the spectrograms as the input. Thus, in this case study, PhyAug does not need to convert spectrograms to MFCC tensors in the data augmentation.

2.5.2 Performance Evaluation

2.5.2.1 Comparison with alternative approaches

Data calibration, CDA [7], and CycleGAN (i.e., essence of [10]) described in §2.4.3.1 are used as the baselines. FADA [15] cannot be readily applied to DeepSpeech2, because FADA requires class labels while DeepSpeech2 performs audio-to-text conversion without the concept of class labels. Differently, PhyAug and the three used baselines transform data without needing class labels.

■ **Data calibration:** Its results are shown by the histograms labeled "calibration" in Fig. 2.12. Compared with the unmodified baseline, this approach reduces some WERs.

■ **PhyAug:** Among all tested approaches, PhyAug achieves the lowest WERs for all microphones. Compared with the unmodified baseline, PhyAug reduces WER by 60%, 41%, 37%, 70%, and 42%, respectively, for the five microphones toward the oracle WER.

■ **CDA [7]:** It performs better than the data calibration approach but worse than PhyAug. As PhyAug is directed by the target microphone's actual characteristics, it outperforms CDA that is based on the *predicted* characteristics that may be inaccurate.

■ **CycleGAN:** A 3.5-hour speech dataset is recorded and used to train a CycleGAN to translate samples captured by a target microphone to the source domain of LibriSpeech dataset. Unfortunately, DeepSpeech2's WERs on the data translated by CycleGAN from the microphones' samples are higher than 90%. A possible reason is as follows. Unlike the KWS task studied in Mic2Mic [10] and §2.4 of this thesis, which discriminates a few target classes only, end-to-end ASR is much more complicated. CycleGAN may require much more training samples than required to achieve good performance.

2.5.2.2 Impact of various factors on PhyAug

The impact of the following three factors is evaluated on PhyAug: the indoor location of the microphone, the distance between the microphone and the sound source, and the environment type. The evaluation methodology is adopted as follows. When the impact of a factor is evaluated, the remaining two factors are fixed. For a certain factor, let X and Y denote two different settings of the factor. PhyAug(X,Y) is used to denote the experiment in which the microphone profiling is performed under the setting X and then the transferred model is tested under the setting Y. Thus, PhyAug(X,X) evaluates *in situ* performance; PhyAug(X,Y) evaluates the sensitivity to the factor.

■ **Impact of microphone location:** Microphones at different locations of an indoor space may be subject to different acoustic reverberation effects. Experiments are setup at three spots, namely, A, B, and C, in a $7 \times 4\,\mathrm{m}^2$ meeting room. Spot B is located at the room center; Spots A and C are located at two sides of B, about 1 m apart from B along the room's long dimension. Fig. 2.13 shows the results of the unmodified baseline approach tested at three spots, as well as PhyAug's *in situ* performance and location sensitivity. PhyAug's *in situ* WERs (i.e., PhyAug(A, A), PhyAug(B, B), PhyAug(C, C)) are consistently lower than those of the unmodified baseline. The WERs of PhyAug(A, B) and PhyAug(A, C) are slightly higher than PhyAug(B, B) and PhyAug(C, C), respectively.

Similarly, the impact of the microphone locations is evaluated on CDA and Data calibration approaches. Fig. 2.14 and Fig. 2.15 show the results of CDA and Data

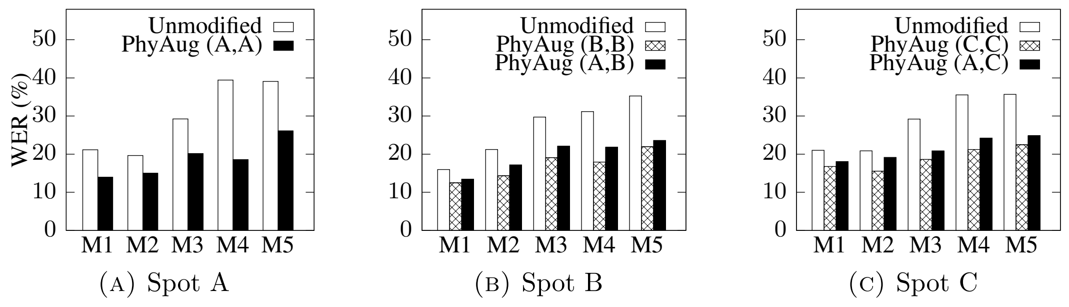


FIGURE 2.13: PhyAug's *in situ* performance and location sensitivity evaluated at three spots in a $7 \times 4\,\mathrm{m}^2$ meeting room.

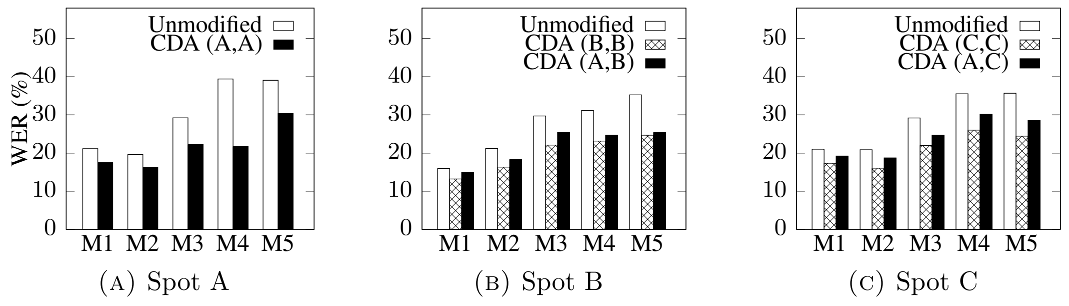


FIGURE 2.14: CDA's *in situ* performance and location sensitivity evaluated at three spots in a $7 \times 4\,\mathrm{m}^2$ meeting room.

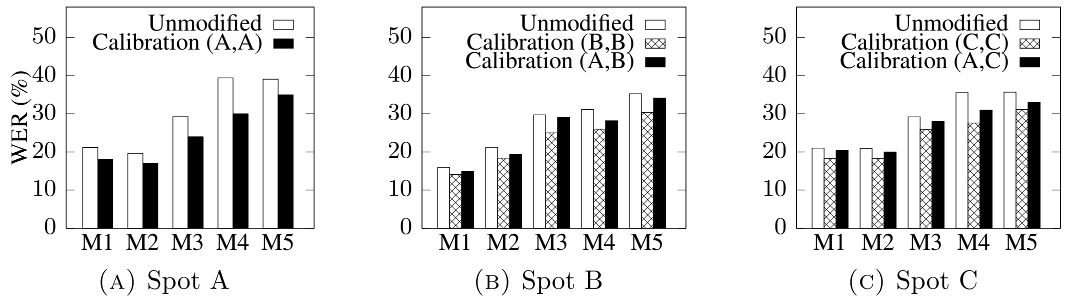
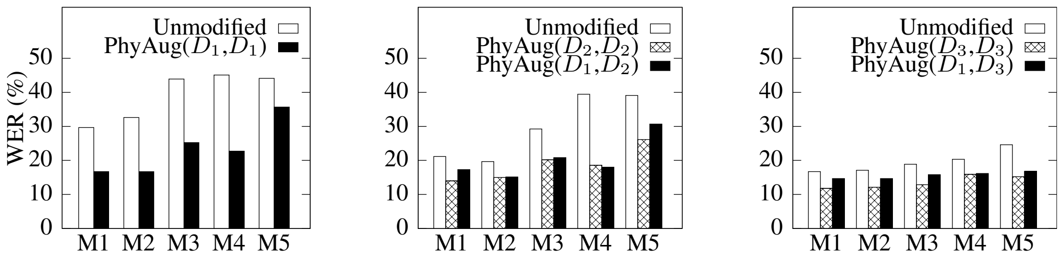


FIGURE 2.15: Data calibration's *in situ* performance and location sensitivity evaluated at three spots in a $7 \times 4\,\mathrm{m}^2$ meeting room.

calibration, respectively. Similar to PhyAug, it is observed that the WERs are consistently lower than the unmodified results at three tested locations for both approaches, and the WERs of (A, B) and (A, C) are slightly higher than (B, B) and (C, C). These results show that location affects the performance of a certain ASR model transferred by all evaluated approaches, but not much. Thus, CDA and Data calibration also exhibit a similar robustness trend as PhyAug.

■ **Impact of microphone-speaker distance:** The distance affects the signal-to-noise ratio (SNR) received by the microphone and thus ASR performance. With the setup at the aforementioned Spot C, the distance between the microphones and



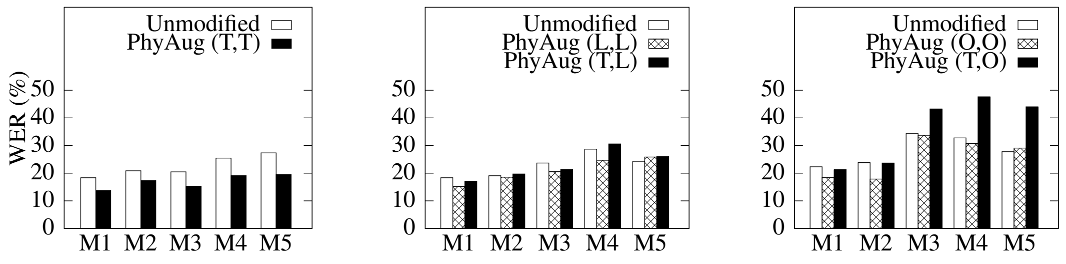
(A) 75 cm (D_1) (B) 45 cm (D_2) (C) 15 cm (D_3)

FIGURE 2.16: PhyAug's *in situ* performance and microphone-speaker distance sensitivity evaluated with three distances.

the iPhone 7 used to play test samples is varied to be 75 cm, 45 cm, and 15 cm (referred to as D_1, D_2, and D_3). Fig. 2.16 shows the results. The unmodified baseline's WERs become lower when the microphone-speaker distance is shorter, due to the increased SNR. PhyAug's *in situ* WERs (i.e., PhyAug(D_1,D_1), PhyAug(D_2,D_2), and PhyAug(D_3,D_3)) are consistently lower than those of the unmodified baseline. The performance gain is better exhibited when the distances are longer. This suggests that *in situ* PhyAug improves the resilience of DeepSpeech2 against weak signals. In most cases, the WERs of PhyAug(D_1,D_2) and PhyAug(D_1,D_3) are slightly higher than those of PhyAug(D_2,D_2) and PhyAug(D_3,D_3), respectively. This shows that the microphone-speaker distance affects the performance of a certain model transferred by PhyAug, but not much. Thus, PhyAug for DeepSpeech2 is insensitive to the microphone-speaker distance.

Another related factor is the speaker's azimuth with respect to the microphone that can affect the quality of the recorded signal due to the microphone's polar-pattern characteristic. For a certain microphone, the different azimuths of the speaker create multiple target domains. If the speaker's azimuth can be sensed (e.g., by a microphone array), PhyAug can be applied. However, as the five microphones used in this thesis lacks speaker azimuth sensing capability, the application of PhyAug to address the domain shifts caused by the speaker's azimuth is skipped.

■ **Impact of environment:** Different types of environments in general have distinct acoustic reverberation profiles, which may affect the microphone's signal reception. The experiment setup is deployed in three distinct types of environments: a small tutorial room (T), a large lecture theatre (L), and an outdoor open area (O). Fig. 2.17 shows the results. The unmodified baseline approach has similar results in T and L. Its WERs become higher in O, because O has a higher level of background noise. PhyAug's *in situ* WERs in T, i.e., PhyAug(T,T), are consistently lower than



(A) Tutorial room (B) Lecture theater (C) Open area

FIGURE 2.17: PhyAug's *in situ* performance and environment sensitivity evaluated in three types of environment, namely, small tutorial room (T), large lecture theater (L), and outdoor open area (O).

those of the unmodified baseline. PhyAug(L,L) and PhyAug(O,O) reduce WERs compared with the unmodified baseline, except for the low-quality microphone M5. As M5 has higher noise levels, the microphone profiling process may not generate fidelity FRCs for M5, leading to increased WERs. As shown in Figs. 2.17b and 2.17c, the WERs of PhyAug(T,L) and PhyAug(T,O) are higher than those of the unmodified baseline. The above results show that PhyAug for DeepSpeech2 may have degraded performance on low-quality microphones. In addition, PhyAug for DeepSpeech2 is sensitive to various environments.

2.5.3 Application Considerations

From results presented in §2.5.2, PhyAug suits ASR systems deployed at fixed locations, such as residential and in-car voice assistance systems, as well as minutes transcription systems installed in meeting rooms. PhyAug can also be applied to the *ad hoc* deployment of ASR and automatic language translation for a multilingual environment.

2.6 Case Study 3: Acoustics-based Room Recognition (ARR)

Smartphone indoor localization without using extra sensors and infrastructure is desirable. Recent studies exploit the smartphone's built-in audio system for infrastructure-free room-level indoor localization [26, 30]. Specifically, they use a smartphone to sense a room's acoustic background spectrogram (ABS) [26] or the room's reverberation in response to a probe sound emitted by the smartphone [30]. They follow supervised learning to train a model using labeled data samples collected from multiple rooms. Then, the smartphone with the model can recognize which room it is located in using the ABS or room reverberation sensed by the smartphone. Different from the previous two case studies (KWS, ASR) that aim at interpreting the voices, ARR uses acoustic signals to sense the environment. Since ARR uses a smartphone microphone as the sensor, presumably, its performance can be affected by the heterogeneity of the smartphones' microphones. Specifically, if the target smartphone deployed with the trained ARR model is different from the smartphones or specialized acoustic devices used to collect the training data samples, the performance of the ARR model may drop. Conventional data augmentation approaches [7] may fail to capture such device variability because of a lack of target sensors' domain knowledge. Data translation approaches, e.g., mic2mic [10] only address the single device-to-device data translation. In addition, it requires a translation module installed on each device, hindering the generality of the approach.

In this section, it is validated that the main cause of the ARR model performance drop is microphone variability. To address this issue, PhyAug is applied to recover the performance degradation of the ABS-based ARR model when being applied on a specific smartphone. Specifically, the smartphone's ABS profile is exploited to perform data translation from a source smartphone to the target smartphone. Then, the transfer learning technique is applied to obtain a domain-adapted ARR model for the target smartphone.

In this case study, **source domain** is the dataset collected from the smartphone's microphone used to train the ARR model; **target domain** is the dataset captured from a different smartphone; **first principle** is the microphone's FRC.

2.6.1 Problem Description

In this section, the procedures for ABS feature extraction and the DNN model used for room recognition are described. Then, the impact of smartphone microphone variability on the pre-trained ARR model's accuracy is measured. Finally, PhyAug is applied to recover the model accuracy loss and compare PhyAug with baseline approaches.

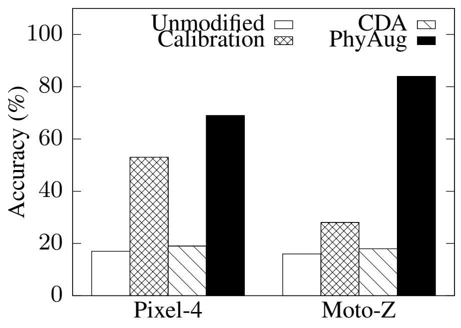
2.6.1.1 ABS feature extraction and DNN model design

The acoustic signal preprocessing steps described in [26] are followed to extract ABS features. An ABS feature is extracted as follows: First, the smartphone records a 1-second long background sound within a room at a sampling rate of 44.1 kHz. Second, short-time Fourier transform (STFT) is applied to the signal by sliding a 1,024-point hamming window with 512 points of overlap to obtain the ABS. Third, the frequencies that are greater than 7kHz are discarded and the values in each frequency bin are sorted. Lastly, the 5th percentile of the sorted values in each frequency bin is selected to form a one-dimensional vector with 163 elements as the ABS feature. The sorting and selection make sure the feature characterizes the background sound, rather than the transient foreground sound.

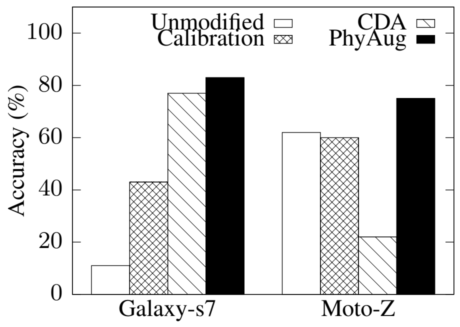
Different from the ABS-based ARR system in [26] that uses nearest-neighbor classification, a 5-layer MLP model that takes the ABS feature as input is adopted to perform room recognition. The number of neurons in the five layers is 163, 256, 512, 1024, and N, where N represents the number of rooms. During training, a 0.4 dropout rate is adopted between any two hidden layers to prevent overfitting. The model is implemented using Pytorch [48].

2.6.1.2 Impact of smartphone microphone variability on ARR

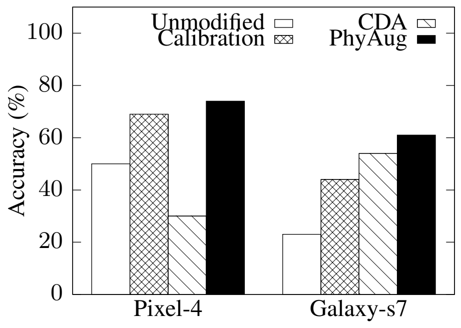
The impact of smartphone microphone on a pre-trained ARR model is investigated. Three smartphones of different models (Samsung Galaxy S7, Motorola Moto Z, and Google Pixel 4) are used to collect 20 rooms' ABS features. For each room, 10-minute training data and 2-minute testing data are collected using each smartphone. The DNN model is trained with data collected using a specific smartphone and then test the trained model with data collected using all smartphones. The results are shown by the histograms labeled "Unmodified" in Fig. 2.18a, 2.18b, and 2.18c, respectively. Taking Fig. 2.18a as an illustration, the source device used to train the DNN model is Galaxy S7. The oracle accuracy numbers reported in the sub figure captions are obtained by training and testing the model on the data collected from the same smartphone, which are 98% for Galaxy S7 and Pixel 4, and 99% for Moto Z. Thus, the ABS-based ARR can achieve high accuracy on recognizing different rooms if the source and target devices are identical. However, from Fig. 2.18a, when applying the model trained on Galaxy S7 to Pixel 4 and



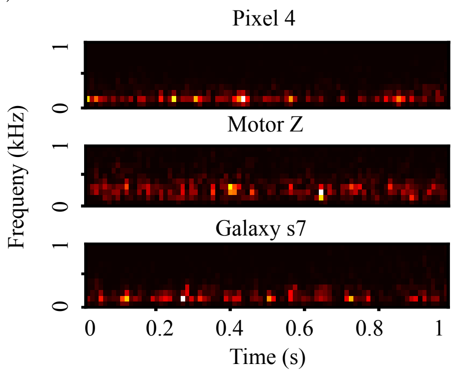
(A) Source device: Galaxy S7 (oracle accuracy: 98%)



(B) Source device: Pixel 4 (oracle accuracy: 98%)



(C) Source device: Moto Z (oracle accuracy: 99%)



(D) ABSes captured by different phones at the same time

FIGURE 2.18: Impact of smartphone microphone variability on ABS-based ARR and comparison of different approaches.

Moto Z, the DNN model's accuracy drops to 17% and 16%, respectively. Similar substantial accuracy drops can be observed when the source smartphone is Pixel 4 or Moto Z. These results show that the ABS-based ARR is highly sensitive to smartphone microphone variability.

To visualize the differences between the acoustic traces collected from different smartphones in the same room, Fig. 2.18d plots the spectrograms of the phones' 1-second data traces collected at the same time. It shows that different smartphones record different ABSes, which is caused by the microphone heterogeneity. From this observation, the research question is how to exploit the microphone characteristics to recover the ARR DNN's accuracy loss?

2.6.2 PhyAug for Acoustics-based Room Recognition

Similar to §2.4.2, PhyAug for ARR consists of *fast microphone profiling* and *model transfer via data augmentation.*

For *fast microphone profiling*, a new approach that is different from but related to that in §2.4.2 is adopted. Specifically, instead of using a speaker to playback the white noise, the smartphone is placed in a *profiling room* to record a 1-minute ABS for phone characterization. The profiling room can be different from those to be recognized. The procedure in §2.6.1.1 is followed to obtain the spectrogram over a 1-minute window. Then, the values in each frequency bin are sorted and the average of the values is computed from the first percentile to the 20th percentile. The resulting averages for all the frequency bins form the smartphone's *ABS profile* that is specific to the profiling room. The ABS profiles of the source smartphone and the target phone are obtained, which are denoted by $ABS_s(f)$ and $ABS_t(f)$, respectively.

For *model transfer via data augmentation,* the source-domain training data is transformed into the target domain. The source-target transfer function is $H(f) = \frac{ABS_t(f)}{ABS_s(f)}$. Then, the labeled source-domain ABS features are multiplied with $H(f)$ to generate augmented target-domain ABS features. Finally, the ARR model is re-trained with the augmented data.

2.6.3 Performance Evaluation

PhyAug is compared with two alternative approaches: data calibration and CDA. The evaluation results are shown in Fig. 2.18a, 2.18b and 2.18c, which use Galaxy S7, Pixel 4, and Moto Z as the source device, respectively.

■ **Data calibration:** This approach converts the target domain data back to the source domain, then applies the pre-trained model on the converted data. The histograms labeled "Calibration" show the results of this approach. Compared with the "Unmodified" results, this approach recovers certain amount of the accuracy loss for most source-target phone combinations. However, it leads to lower accuracy when the source and target phones are Pixel 4 and Moto Z. The accuracy drops from 62% to 60%.

■ **CDA:** The scheme presented in §2.4.3.1 is followed and target smartphones' ABS profiles are used to generate a *heterogeneity dataset*. A DNN model is trained on this dataset and evaluated on target smartphones. The histograms labeled "CDA" show the results. Compared with the "Unmodified" results, considerable accuracy recovery is observed when transferring from Pixel 4 to Galaxy S7 and from Moto Z to Galaxy S7. However, CDA underperforms when transferring between Pixel 4 and Moto Z. The reason is as follows. From the "Unmodified" results, any source-target pair involving Galaxy S7 has poor result. For example, the DNN model's absolute accuracy drops between Pixel 4 with Moto Z is around 30% - 40%, whereas the accuracy drops between Galaxy S7 with Pixel 4 or Galaxy S7 with Moto Z are more than 70%. This implies that Galaxy S7's ABS profile is significantly different from those of Pixel 4 and Moto Z. Under the CDA approach, when the transfer is between Pixel 4 and Moto Z, the Galaxy S7 is used as one of the two phones in the heterogeneity dataset. As a result, the heterogeneity dataset has a complex pattern, which adversely affects the dataset's representativeness.

■ **PhyAug:** PhyAug can recover the accuracy loss for any source-target smartphone pair. In addition, PhyAug achieves the best performance recovery among all evaluated approaches. Specifically, PhyAug recovers 24% to 72% absolute accuracy degradations on different smartphone pairs. In particular, when the source device is Galaxy S7, the "unmodified" accuracies on Pixel 4 and Moto Z are 17% and 16%. PhyAug can recover 52% and 68% absolute accuracy losses, outperforming significantly over the data calibration and CDA approaches.

2.6.4 Summary

The DNN based mobile application generally suffers from performance degradation due to the heterogeneity of mobile sensors. This case study applies PhyAug to recover the ARR performance loss caused by smartphone microphone variations. PhyAug only requires smartphones to record 1-minute ABS profiles in a certain room and achieves significant accuracy recovery. This case study shows that PhyAug can be used to address the sensor heterogeneity issue in DNN-based mobile sensing applications.

2.7 Case Study 4: Fisheye Image Recognition

DNN-based visual sensing can be found in many IoT applications, including video surveillance [49], augmented reality [50], and autonomous driving [51]. Many such applications use fisheye cameras. The fisheye camera is different from the normal pinhole camera with rectilinear mapping. The fisheye camera produces images with a wide field of view (FOV) while creating strong distortions due to the non-linear mapping of optical lens systems.

Prevalent image datasets consist of samples obtained using pinhole cameras. Standard DNNs that achieve state-of-the-art performance are also trained and tested on such pinhole camera datasets [52, 53]. They may not perform well on the images collected by fisheye cameras. Image rectification is a conventional approach that applies inverted fisheye models to fix distorted images. However, image rectification has two main limitations [53]. First, as fisheye images contain greatly distorted peripheries, mapping the limited pixels from the image periphery to a larger region leads to information loss. Second, the image rectification requires additional processing time for every image before the image classification. The processing time grows drastically as the image resolution increases. Thus, the image rectification approach is not suitable for applications that impose both deadline and high-resolution requirements, e.g., visual sensing-based pedestrian detection on a moving vehicle.

In this section, PhyAug is applied to recover DNN's performance degradation caused by fisheye camera distortion without performing image rectification. First, a non-linear polynomial model is used to augment the original dataset to the images with fisheye distortions. Then, a transfer learning technique is applied to obtain a domain-adapted DNN model for fisheye images.

In this case study, **source domain** is the image dataset that is captured by pinhole cameras and used to train the DNN; **target domain** is the fisheye camera dataset for specific IoT applications; **first principle** is the fisheye camera model described by a non-linear polynomial function.

2.7.1 Problem Description

First, the original dataset captured by pinhole cameras is introduced. Then, the camera model used to generate synthesized fisheye images is described. Finally, the performance of PhyAug, CDA, and the image rectification approach is compared.

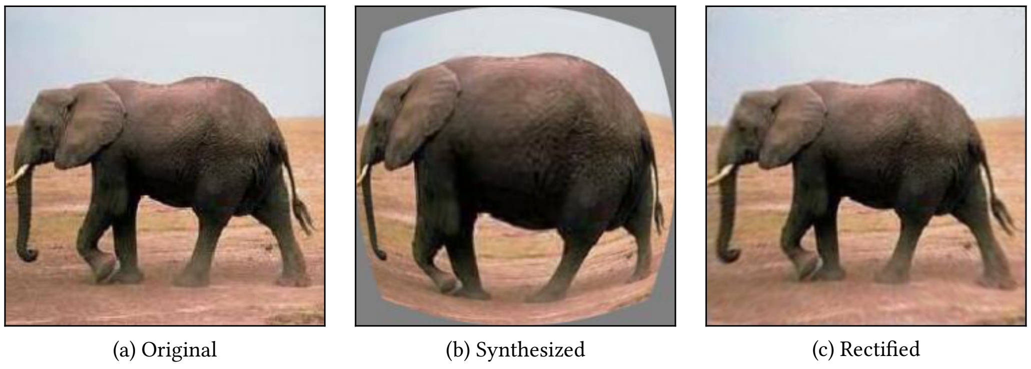


FIGURE 2.19: Fisheye image construction and rectification using the parameterized model. (a) original image; (b) synthesized fisheye image; (c) rectified image.

2.7.1.1 Fisheye camera model

For a given camera, distortions occur when the scenes deviate from the rectilinear projection. The most common source of image distortion is the radial distortion caused by the camera's optical lens system. Fisheye cameras produce strong radial distortions on images. Several models have been proposed to characterize a fisheye camera [54]. In this thesis, a generic fisheye model [14] is adopted, which is a fourth-order polynomial function:

$$R_{src} = r \cdot \left(A \cdot r^3 + B \cdot r^2 + C \cdot r + D\right), \tag{2.1}$$

where r is the destination image radius and R_{src} is the source pixel. In this model, image radius is normalized, so that $r = 1$ refers to the half minimum width or height of the input image. A, B, C represent the distortion of the image. When the three values are positive, the image contains barrel distortion. When they are negative, pincushion distortion occurs in the image. D describes the linear scaling of the image. The values of A, B, C, D are fixed for a given camera and different across camera models. In this thesis, the fisheye model is applied with various parameters to a public dataset, enabling the construction of data collected by different camera models. Subsequently, the domain adaptation performance of different approaches is evaluated.

2.7.1.2 Dataset and deep neural network model

In this case study, experiments use Caltech-101 dataset [55]. It consists of image objects in 101 classes. Each class contains 40 to 800 images, with a total of around 9,000 images. The dataset is split into training, validation, and testing sets at 70%, 10% and 20% ratios.

The ResNet-50 CNN model [56] is used to perform multi-class classification. ResNet-50 consists of 48 convolutional layers along with one max-pooling and one average-pooling layer. The base model is pre-trained on the ImageNet dataset [2]. The model is customized by reducing the output layer size from 1,000 neurons to 101 neurons, to align with Caltech-101's class number. The *freeze-and-train* is used to transfer the pre-trained base model for Caltech-101 dataset. In particular, weights in feature encoders are frozen and weights in the fully connected layers are updated. In this case study, the model training and evaluation are implemented using PyTorch.

2.7.1.3 Impact of image distortion on DNN model

The performance of a DNN model trained on a standard dataset captured by pinhole cameras is investigated on images captured by fisheye cameras. Due to the lack of publicly available fisheye image datasets, a synthetic dataset is used for experiments. The images are distorted using the model depicted in Eq. (2.1). Fig. 2.19 shows a sample. Fig. 2.19(a) is the original image. Fig. 2.19(b) is the corresponding distorted image with a positive parameter set of $A = 0.2, B = 0.2, C = 0.01, D = 0.59$, where a strong radial distortion effect can be observed. The same parameters set are applied on Caltech-101 to generate the synthetic fisheye image dataset. The fisheye model is implemented using the Wand package [57] in Python. The pre-trained model is tested on both the original and distorted datasets. As shown in Fig. 2.20, the pre-trained ResNet-50 model achieves 90% oracle accuracy on the original test set. The accuracy on the distorted dataset is 72%. Thus, there is an 18% absolute accuracy drop. Many fisheye cameras can produce images with FOV greater than 180°, resulting in stronger non-linear distortions. Hence, significant accuracy drops can be observed. Based on the experiment results, the research question for this case study is: how to exploit the

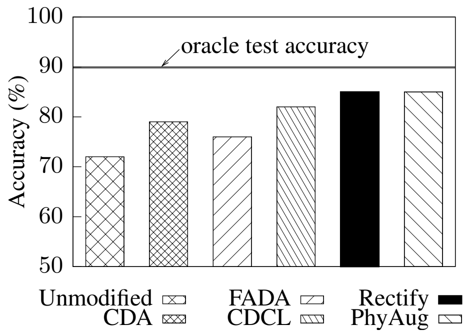


FIGURE 2.20: ResNet-50 test accuracy on Caltech-101 dataset using different approaches.

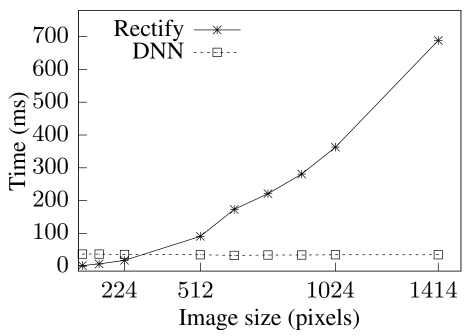


FIGURE 2.21: Execution time vs. image size on an NVIDIA Jetson Nano.

first principle of fisheye camera models to recover the image classification DNN's accuracy loss?

2.7.2 PhyAug for Fisheye Image Recognition

A generic fisheye model is used, which is described by a fourth-order polynomial function (cf. Eq. (2.1)). Note that a fourth-order polynomial function can well represent the model of fisheye cameras, while higher orders provide no additional benefit in terms of accuracy [53]. A system designer can reconstruct the camera model based on lens parameters stored in the image's metadata. In case that the metadata is missing, a standard calibration procedure can be applied to estimate a camera's intrinsic parameters [58]. Specifically, the camera can capture photos of a printed image with a known pattern, e.g., a chessboard, at different angles. In general, 20 to 30 pictures from a fisheye camera are sufficient to obtain distortion parameters. Such calibration tools are available in OpenCV [59] and Matlab [58].

PhyAug for FIR has the following two steps. First, the parameters of Eq. (2.1) for a specific fisheye camera are estimated and the fisheye model is applied to the original images to generate augmented samples in the target domain. Then, the DNN model is trained with the augmented data for the fisheye camera. In this case study, it is assumed that the parameters of the target camera are known.

2.7.3 Performance Evaluation

2.7.3.1 Baseline approaches.

PhyAug performance is evaluated against the following baseline approaches.

■ **CDA:** This approach follows the conventional data augmentation scheme to build a DNN model robust to camera lens distortions. Specifically, it randomly generates a set of potential fisheye camera models using Eq. (2.1). Subsequently, CDA applies these camera models to augment the training dataset. In the evaluation, 10 sets of parameters are randomly generated and used to augment the Caltech-101 training dataset.

■ **FADA:** This approach follows the adversarial domain adaptation as presented in §2.4 to train a domain-invariant feature encoder using paired source-domain and target-domain images. The feature encoder used is a ResNet-50 model and the discriminator used is a 4-layer MLP with the neurons in the four layers are 1024, 1024, 1024 and 101. The number of images in each class used for feature encoder training is set to 100 due to the relatively complex feature space for this case study.

■ **CDCL:** This approach aims to train a domain-invariant feature encoder by applying contrastive learning. Similar procedures as presented in §2.4 are followed to construct the positive and negative samples for training. The number of images in each class used for contrastive feature training is 400.

■ **Image rectification:** This approach follows the conventional image rectification scheme to correct image distortions. Once the camera lens distortion parameters are determined, one can tune the same model as in Eq. (2.1) to rectify the image. Fig. 2.19(c) shows the rectified result by applying the inverted parameters on the distorted image. In the evaluation, the estimated parameters are applied to rectify distorted images and then evaluate the pre-trained model on rectified images.

2.7.3.2 Evaluation results.

PhyAug and the baseline approaches are applied to Caltech-101 dataset. The results are presented as follows:

■ **CDA:** As shown in Fig. 2.20, CDA achieves 79% accuracy on the target fisheye dataset. This approach achieves higher accuracy than the unmodified result, which directly applies the pre-trained model to the target test data. However, there is still an 11% accuracy gap towards the oracle accuracy. The result shows that the model trained with CDA mitigates the impact of fisheye camera distortion.

■ **FADA:** As shown in Fig. 2.20, FADA achieves 76% accuracy on the fisheye images. It only gives 4% absolute accuracy gain compared with the unmodified result. The performance increase is subtle even the used target-domain images are increased for model training. FADA does not perform well on FIR compared to the KWS. The reasons are two-fold. First, the data complexity of FIR dataset is higher than KWS dataset. The size of a fisheye image is $3 \times 224 \times 224$, whereas the size of KWS MFCC is $1 \times 101 \times 40$. Second, the model complexity used is much higher. The ResNet-50 model is used for FIR and the ResNet-18 is used for KWS. Thus, it is more difficult to adapt the DNN for FIR using limited data. It is concluded that FADA does not generalize well on the complex tasks for domain adaptation.

■ **CDCL:** As shown in Fig. 2.20, CDCL achieves 82% test accuracy on the fisheye images, representing a 10% absolute accuracy increase. The result shows that CDCL can effectively learn a feature embedding for both the source-domain and the target-domain data. However, the performance of CDCL is sensitive to the number of the target-domain data used for contrastive feature training.

■ **Image Rectification:** The pre-defined fisheye model in Eq. (2.1) is used to rectify the fisheye image, then apply the pre-trained model on rectified images. As shown in Fig. 2.20, image rectification achieves 85% test accuracy. This shows that image rectification can effectively recover information loss caused by Eq. (2.1). However, as the image rectification algorithm is applied on every image, thus will incur extra computing time. The evaluation of image rectification time on a resource-constrained device is presented in the next section.

■ **PhyAug:** The re-trained model is applied to the distorted image dataset. The test accuracy is 85%. PhyAug can effectively recover the accuracy loss caused by camera distortions. However, there is still a 5% gap compared with the oracle test accuracy. This is because of the information loss when applying the fisheye camera distortion model on the original image data. As shown in Fig. 2.19, the occupied region of the distorted image (b) is smaller than the original image (a). The gray area in image (b) represents the amount of lost content caused by the distortion. Therefore, a certain amount of accuracy loss is expected.

The execution time of image rectification and DNN inference are evaluated on an NVIDIA Jetson Nano that can execute DNN models. It is equipped with a quad-core Cortex-A57 CPU, a 128 core Maxwell GPU and 4GB RAM. Results are shown in Fig. 2.21. The curve labeled "Rectify" is the time taken to rectify an image with respect to the image resolution. The horizontal axis represents the length of a squared image in pixels, e.g., point 224 refers to a 224×224 RGB image. The solid line in Fig. 2.21 shows that image rectification requires more processing time when the resolution increases. It takes around 18 ms to rectify an image of size 224×224. When processing an image of 1414×1414 pixels, the rectification time increases up to 688 ms. The curve labeled "DNN" is DNN inference time on images with different resolutions. Note that The image rectification algorithm is run on CPU as there is a lack of GPU version. In practice, there will incur overhead to create GPU compilable program for every edge device. The dotted line in Fig. 2.21 shows that the inference time is consistent across all tested images, which is around 36 ms. This is because the floating-point operation for a pre-defined neural network is generally fixed and regardless of input image size. The experiment shows that ResNet-50 network can work effectively on images with different input sizes. In time-critical applications like autonomous driving, large delays are unacceptable. As PhyAug directly adapts the DNN model to the target domain, it has the advantage of avoiding the time-consuming rectification on resource-constrained devices.

2.7.4 Summary

This case study applies PhyAug to a visual sensing application. Applying DNNs trained using standard pinhole camera image datasets on fisheye images suffers performance degradation. Despite the prevalent use scenarios of fisheye cameras in visual sensing applications, few publicly fisheye datasets are available for training customized DNNs. It is identified that the main contributor to the performance drop is the non-linear mapping of the fisheye camera. The parameterized fisheye model is applied to transfer existing DNNs to a specific fisheye camera via guided data augmentation. The results show that PhyAug can significantly recover the accuracy loss, while requiring no data collection effort in the target domain of the fisheye camera. The experiment on NVIDIA Jetson Nano shows that PhyAug requires less computational overhead than the conventional image rectification approach.

2.8 Case Study 5: Seismic Source Localization

Estimating the location of a seismic event source using distributed sensors finds applications in earthquake detection [60], volcano monitoring [61], footstep localization [62], and fall detection [63]. TDoA-based localization approaches have been widely employed in these applications. The TDoA measurement of a sensor is the difference between the time instants at which the signal from the same event arrives at the sensor and a reference sensor. In the source domain where the medium density is spatially homogeneous, the seismic signal propagation velocity is also spatially homogeneous. To address measurement noises, the TDoA-based multilateration problem is often solved under a least squares formulation. However, in practice, the medium density is often spatially heterogeneous. This case study aims to deal with the target domain where the medium density is unknown and uneven. For instance, the density of the magma beneath an active volcano varies with depth. As such, seismologists need a *slowness model* that depicts the seismic wave propagation speeds at different depths before hypocenter estimation can be performed [64]. In footstep localization and fall detection, the heterogeneity of the floor materials affects the seismic wave propagation speed and degrades the performance of the simplistic multilateration formulation. Unfortunately, directly measuring the slowness model is tedious or even unfeasible in many cases.

To cope with heterogeneous media, the fingerprinting approach can be employed. Specifically, when a seismic event with a known location is triggered, the TDoA measurements by the sensors form a fingerprint of the known location. With the fingerprints of many locations, a seismic event with an unknown location can be localized by comparing the sensors' TDoA measurements with the fingerprints. The fingerprints can be collected by triggering controlled events at different locations, e.g., controlled explosions in seismology [65] and hammer excitations in structure health monitoring [66]. Under the fingerprinting approach, determining the location of an event source can be formulated as a classification problem, in which the

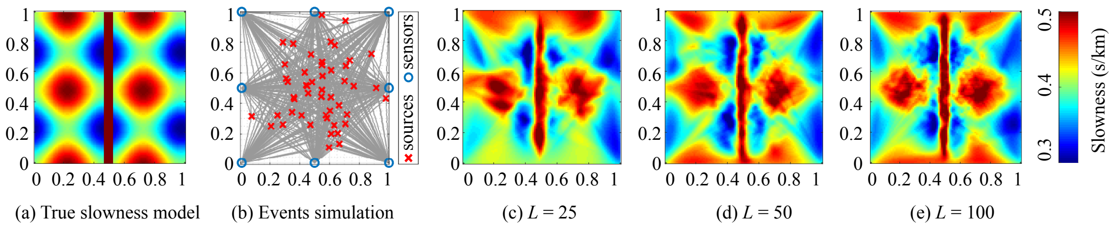


FIGURE 2.22: The $1 \times 1\,\text{km}^2$ 2D field considered in the seismic source localization case study. (a) The ground-truth slowness model with 100×100 grids. (b) Seismic event source locations and their ray paths to sensors. (c)-(e) The estimated slowness models with 25, 50, and 100 seismic events that occur at random positions in the 2D field as the training samples, respectively.

fingerprint is the input data and the corresponding location is the class label. To achieve a high localization accuracy, a laborious blanket process of fingerprinting many/all locations is generally required. In this case study, it is shown that by exploiting the first principle of seismic wave propagation in an uneven medium, the number of fingerprints can be significantly reduced whereas achieving a certain level of localization accuracy. Note that, from §2.8.3.2, even with a homogeneous medium, the fingerprinting approach outperforms the least squares approach in terms of response time, while offering comparable localization accuracy.

In this case study, **source domain** is the homogeneous medium for seismic signals; **target domain** is the heterogeneous medium for seismic signals; **first principle** is the slowness model characterizing seismic signal propagations in heterogeneous media.

2.8.1 Problem Description

Consider a 2D field divided into $W_1 \times W_2$ grids, where W_1 and W_2 are integers. Thus, the field has a total of $N = W_1 \cdot W_2$ grids. Each grid is associated with a slowness value in seconds per kilometer (s/km), which is the reciprocal of the seismic wave propagation speed in the grid. It is assumed that the slowness at any position in a grid is constant, while the slowness in different grids can be distinct. Thus, the slowness model is a matrix (denoted by $\mathbf{S} \in \mathbb{R}^{W_1 \times W_2}$) with the grids' slowness values as the elements. In this case study, a slowness model from [67] as shown in Fig. 2.22(a) is adopted, which is a $1 \times 1\,\text{km}^2$ square field with a wavy pattern and a barrier stripe in the middle. The pattern and the barrier create challenges to the event localization and will also better exhibit the effectiveness of PhyAug in addressing heterogeneous medium.

There are a total of M seismic sensors deployed in the field. When there is an event occurring in the field, the propagation path of the seismic wave front from the event source to any sensor follows a straight ray path. For instance, Fig. 2.22(b) shows the ray paths for the eight sensors considered in this case study. Note that this case study can be also extended to address the refraction of the seismic wave at the boundary of any two grids by using a ray tracing algorithm [64] to determine the signal propagation path. In Fig. 2.22(b), the deployment of the sensors at the field boundary is consistent with the practices of floor event monitoring [62] and volcano activity monitoring [61]. The seismic event locations follow a Gaussian distribution centered at the field center.

In what follows, the seismic signal propagation process for the lth event is modeled. For the mth sensor, denote the propagation ray path by $p_{l,m}$; denote the *ray tracing* matrix by $\mathbf{A}_{l,m} \in \mathbb{R}^{W_1 \times W_2}$, where its (i,j)th element is the length of $p_{l,m}$ in the (i,j)th grid. If $p_{l,m}$ does not go through the (i,j)th grid, the corresponding element of $\mathbf{A}_{l,m}$ is zero. Let $\mathbf{a}_{l,m} \in \mathbb{R}^{1 \times N}$ denote a row vector flattened from $\mathbf{A}_{l,m}$ in a row-wise way. Therefore, the ray tracing matrix for all sensors in the lth event, denoted by $\mathbf{A}_l \in \mathbb{R}^{M \times N}$, is $\mathbf{A}_l = [\mathbf{a}_{l,1}; \mathbf{a}_{l,2}; \ldots; \mathbf{a}_{l,M}]$. Let $t_{l,m}$ denote the time for the seismic wave front to propagate from the lth event's source to the mth sensor. Denote $\mathbf{t}_l = [t_{l,1}; t_{l,2}; \ldots; t_{l,M}] \in \mathbb{R}^{M \times 1}$. Let $\mathbf{s} \in \mathbf{R}^{N \times 1}$ denote a column vector transposed from the row vector that is the row-wise flattening of the slowness model $\mathbf{S}$. Thus, the first principle governing the propagation times is

$$\mathbf{t}_l = \mathbf{A}_l \mathbf{s}. \tag{2.2}$$

Note that the flattened slowness model $\mathbf{s}$ is identical for all events. Denote by $\widetilde{\mathbf{t}}_l = [\widetilde{t}_{l,1}; \widetilde{t}_{l,2}; \ldots; \widetilde{t}_{l,M}]$ the measurements of the propagation times. $\widetilde{\mathbf{t}}_l = \mathbf{t}_l + \boldsymbol{\epsilon}$ is assumed, where the measurement noise $\boldsymbol{\epsilon} \in \mathbb{R}^M$ is a random variable following an M-dimensional Gaussian distribution $\mathcal{N}(\mathbf{0}_M, \sigma_\epsilon^2 \mathbf{I}_M)$. In the numerical experiments, $\sigma_\epsilon = \xi \cdot \bar{\mathbf{t}}_l$ is set, where ξ is called *noise level* and $\bar{\mathbf{t}}_l$ is the average value of the elements in $\mathbf{t}_l$. In the evaluation experiments, the default noise level is $\xi = 2\%$.

In the TDoA-based fingerprinting approach, a target-domain training data sample consists of the position of the triggered event as the label and the TDoA measurements as the feature. Specifically, if the first sensor is chosen to be the reference, the feature of the lth event is

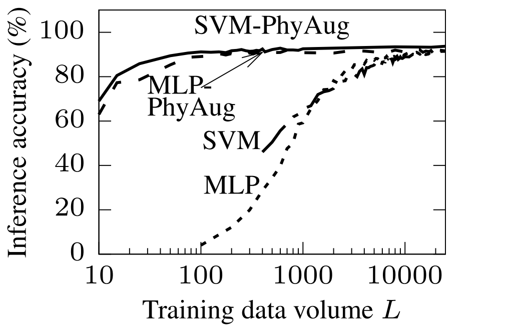
$$\mathbf{f}_l = [\widetilde{t}_{l,2} - \widetilde{t}_{l,1}; \widetilde{t}_{l,3} - \widetilde{t}_{l,1}; \ldots ; \widetilde{t}_{l,M} - \widetilde{t}_{l,1}] \in \mathbf{R}^{(M-1)\times 1}. \tag{2.3}$$

A support vector machine (SVM) or DNN can be trained based on a training dataset and then used to localize an event at run time. The research questions addressed in this case study are as follows. First, how to exploit the first principle in Eq. (2.2) to augment the training dataset? Second, to what extent the demand on actual training data samples can be reduced by applying PhyAug?

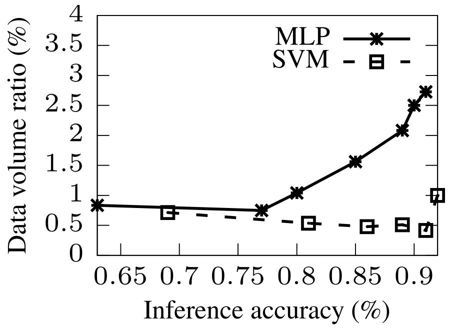
2.8.2 PhyAug for Seismic Source Localization

To use the first principle in Eq. (2.2) to augment the training dataset, the flattened slowness model $\mathbf{s}$ needs to be estimated using some training data samples. This tomography problem can be solved by the Bayesian Algebraic Reconstruction Technique (BART) or Least Squares with QR-factorization (LSQR) algorithm [68]. In this work, BART is applied to generate an estimated slowness model denoted by $\hat{\mathbf{s}}$ based on a total of L training samples collected by triggering events with known positions in the field. The details of BART are omitted here due to space constraint and can be found in [69]. Figs. 2.22(c)-(e) show $\hat{\mathbf{s}}$ when $L = 25$, $L = 50$, and $L = 100$, respectively. It is observed that when more seismic events are used, the $\hat{\mathbf{s}}$ is closer to the ground truth shown in Fig. 2.22(a). The above tomography process uses L labeled target-domain data samples. Thus, PhyAug for this case study requires target-domain class labels as indicated in Table 2.1. As PhyAug can significantly reduce the amount of needed target-domain data samples as shown shortly, the related overhead is largely mitigated.

With the estimated slowness model $\hat{\mathbf{s}}$, a large amount of augmented fingerprints can be generated to extend the training dataset. Specifically, to generate the xth augmented fingerprint denoted by $\mathbf{t}_x$, a position is randomly and uniformly drawn from the 2D field as the event source location and then computes the ray tracing matrix $\mathbf{A}_x$ and the fingerprint $\mathbf{t}_x = \mathbf{A}_x\hat{\mathbf{s}}$. Lastly, the SVM or DNN is trained using



(A) Inference accuracy vs. training data volume for SVM and MLP with or without PhyAug.



(B) Ratio of training data volumes with and without PhyAug vs. required inference accuracy.

FIGURE 2.23: Impact of PhyAug on SVM/MLP-based fingerprinting approaches (number of grids: 400; $\xi = 2\%$).

the extended training dataset consisting of the L genuine training samples and X augmented training samples.

With the above approach, any number of augmented training samples can be generated. In this case study, the following approach is adopted to decide the volume of augmented training samples. Initially, $X = 100 \times N$ is set, where N is the number of grids, and the SVM/DNN is trained with the augmented training dataset. The volume of the augmented training samples is doubled (i.e., $X = 2 \times X$) until the validation accuracy of the trained SVM/DNN saturates.

2.8.3 Performance Evaluation

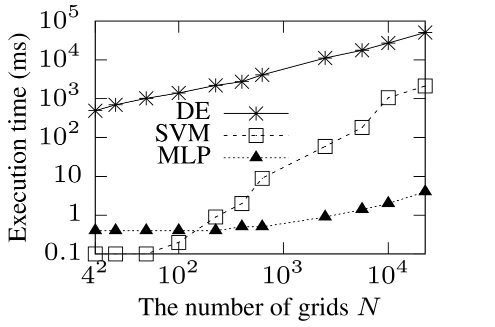
Both SVM and multilayer perceptron (MLP) are used for finger-print-based source localization. SVM is implemented using LIBSVM 3.24 [70]. It uses a radial basis function kernel with two configurable parameters C and γ. During training, grid search is applied to optimize the settings of C and γ. In addition, a 5-layer MLP is constructed. The numbers of neurons in the layers are M, 1024, 1024, 512, and N, respectively. For training, a 0.2 dropout rate is used between any two hidden layers to prevent overfitting. Cross-entropy is used as the loss function at the output layer as the training feedback.

2.8.3.1 Advantages brought by PhyAug to SVM/MLP-based fingerprinting approach

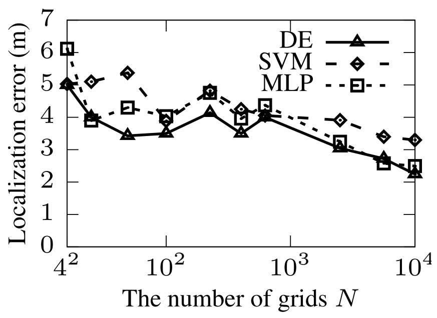
$N = 20 \times 20$ is set. The grid-wise inference accuracy is used as the evaluation metric. Fig. 2.23a shows the inference accuracy of SVM and MLP, without and with PhyAug, versus the training data volume L. First, the results of SVM and MLP without PhyAug are discussed. It is observed that the inference accuracy of SVM and MLP increases with L. When more than 8,000 training samples are provided, SVM and MLP achieve more than 92% inference accuracy. When less than 11,000 training samples are provided, SVM outperforms MLP; otherwise, MLP outperforms SVM. This observation is consistent with the general understanding that deep learning with "big data" outperforms the traditional machine learning approaches. Second, the results of SVM and MLP with PhyAug are discussed. The inference accuracy of SVM-PhyAug and MLP-PhyAug also increases with L. With more training samples, the estimated slowness model $\hat{\mathbf{s}}$ is more accurate. As a result, the augmented data samples will be of higher quality, thus helping the SVM/MLP achieve higher test accuracy. From Fig. 2.23a, it is observed that PhyAug boosts the inference accuracy of SVM and MLP when the training data volume is limited. Fig. 2.23b shows the ratio of the training data volumes required by a classifier with/without PhyAug to achieve a specified inference accuracy. With PhyAug, only less than 3% training samples are needed. This shows that PhyAug is very effective in reducing the demand on training data.

2.8.3.2 SVM/DNN vs. Least Squares Method for Seismic Source Localization

The estimated slowness model $\hat{\mathbf{s}}$ can be directly used to estimate the source location at run time by a least squares method. In the least squares method, Differential evolution (DE) is applied, which is a population-based metaheuristic search algorithm, to perform grid-granular search and iteratively improve a candidate solution p with $\|\mathbf{f} - f\left(\mathbf{A}_p\hat{\mathbf{s}}\right)\|_{\ell_2}^2$ as the error metric. In the above error metric, the $\mathbf{f}$ is the feature vector of TDoA measurements given by Eq. (2.3) for the run-time event; the $\mathbf{A}_p$ is the ray tracing matrix of the candidate position p; the $f(\,\cdot\,)$ is a function converting the seismic propagation times to the feature vector of TDoA measurements. Fig. 2.24 compares the performance of DE, SVM, and MLP in terms of



(A) Inference time vs. the number of grids (both the x- and y-axes are in log scale).



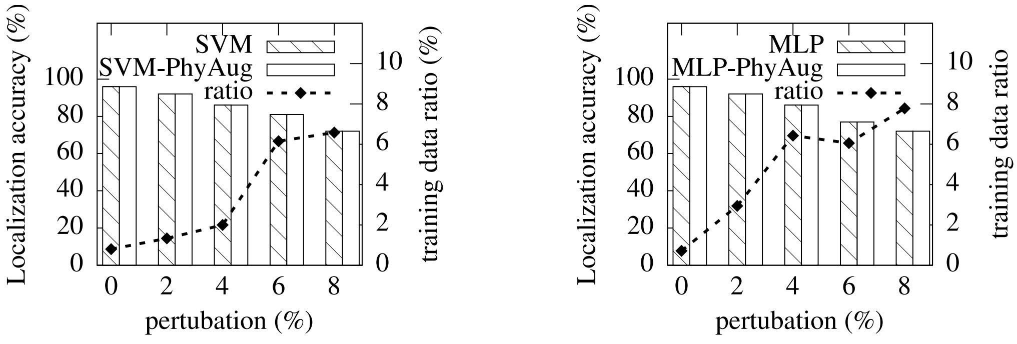
(B) Localization error vs. the number of grids (the x-axis is in log scale).

FIGURE 2.24: Performance comparison of differential evolution (DE), SVM, and MLP.

average execution time over 100 events. In the evaluation, the number of grids N is increased for finer inference granularity. Fig.2.24a shows the execution time versus N. From a regression analysis on the results, DE has a time complexity of $O(N^{0.45})$. It's execution time is several orders of SVM and MLP. For instance, when $N = 22500$, DE's execution time is 50.73 s, which is about 23x and 12,500x longer than SVM's and MLP's, respectively. The long response delays make DE unsuitable for a range of time-critical applications such as earthquake early warning [60]. From Fig. 2.24a, the execution time of ML is within 10 ms when N is up to 22,500. Fig. 2.24b shows the average localization error in terms of Euclidean distance versus N. It is observed that three approaches give comparable localization accuracy. From the above results, SVM and MLP are superior to DE due primarily to response times.

2.8.3.3 Impact of noise level

This section contains experiment results on the impact of noise level ξ on the performance of PhyAug for seismic source localization. As defined in §2.8.1, the TDoA measurement contains a random noise following $\mathcal{N}(\mathbf{0}_M, (\xi\bar{\mathbf{t}}_l)^2\mathbf{I}_M)$. The histograms in Fig. 2.25 show the grid-wise localization accuracy of SVM, MLP, and their PhyAug-assisted variants when ξ increases from 0% to 8%. The dashed curve in Figs. 2.25a and 2.25b shows the ratio between the volumes of actual training data required by SVM/MLP with and without PhyAug. The SVM/MLP approach uses the same amount of training data for all ξ settings, whereas the amount of the actual training data used for the PhyAug-assisted variant is adjusted



(A) Classifier is SVM.

(B) Classifier is MLP.

FIGURE 2.25: Impact of TDoA measurement noise level on PhyAug's effectiveness.

to achieve the same grid-wise localization accuracy as the SVM/MLP approach. From the figure, the localization accuracy decreases with ξ. This is consistent with intuition because larger noise levels lead to more classification errors. In addition, the ratio of the actual training data amounts required by SVM/MLP with and without PhyAug increases with ξ. For example, MLP-PhyAug only requires about 1% of the training data needed by MLP without PhyAug to achieve the same 96% accuracy when $\xi = 0\%$; this ratio increases to about 8% to achieve the same 77% accuracy when $\xi = 8\%$. This is because PhyAug needs more actual training data to estimate a good slowness model when the noise level is higher. Nevertheless, PhyAug reduces the demand for actual training data by a factor of more than 10 when ξ is up to 8%.

2.8.3.4 Summary

Different from the KWS and ASR case studies that use PhyAug to recover recognition accuracy loss mainly caused by sensor hardware characteristics, this case study uses PhyAug to reduce the demand for actual training data in dealing with the complexity of the sensed physical process. Although this case study is primarily based on numerical experiments, the results provide a baseline understanding of the advantages brought by PhyAug.

2.9 Discussions

In many sensing systems, the domain shifts are often governed by first principles. The illustrating case studies have demonstrated the advantages of exploiting the first principles in dealing with domain shifts that are often experienced by deployed sensing systems. In practice, the complexity of the identified first principles and the amount of data available for first principle fitting vary from application to application. The quality of the fitted models affects the performance of the domain adaptation. Though it is desirable to develop the theoretical analysis to describe how much the fitted first principle can capture the true relations between the source-domain data and the target-domain data, such analysis will need to be based on certain assumptions that are application-specific. Intuitively, the first principle described with a more complex parametric model will require more data for fitting. The focus of this thesis is to establish the steps to exploit the physics governing the domain shifts for domain adaptation and show its applicability to a number of case studies.

For applications that lack useful first principles, one may fall back to the existing physics-regardless transfer learning approaches. However, the fallback option should not discourage us from being discerning on the exploitable first principles in the pursuit of advancing and customizing deep learning-based sensing in the domain of physics-rich cyber-physical systems.

2.10 Summary

This chapter describes PhyAug, an efficient data augmentation approach to deal with domain shifts governed by first principles. The applications of PhyAug to five case studies are presented. They have distinct task objectives and require deep models with quite different architectures and scales. The extensive and comparative experiments showed that PhyAug can recover significant portions of accuracy losses caused by sensors' characteristics and reduce target-domain training data sampling complexity in dealing with the domain shifts caused by the variations of the dynamics of the sensed physical process.

Chapter 3

Indoor Smartphone SLAM with Learned Echoic Location Features

This chapter presents ELF-SLAM [1], a smartphone-based SLAM system based on the learned echoic location features (ELFs) using contrastive learning. The organization of this chapter is as follows. §3.1 reviews related work. §3.2 presents preliminaries. §3.3 presents the measurement study. §3.4 presents the design of ELF-SLAM. §3.6 presents the evaluation results. §3.7 discusses several issues. §3.8 summarizes this chapter.

3.1 Related Work

■ **Acoustics-based indoor localization and mapping:** The ubiquity of audio speakers and microphones on consumer electronics has attracted research interest in acoustics-based indoor localization. The infrastructure-based approaches deploy dedicated sound beacons or receivers to localize a mobile receiver or beacon. However, the overhead of deploying the infrastructure is undesirable. The review focuses on infrastructure-free approaches, which are summarized in Table 3.1. The *analytic approach* analyzes the acoustic signal propagation processes. It passively senses the sounds from the source or actively emits acoustic signals and analyzes the echoes. VoLoc [73] uses a smart speaker to localize a user based on the angle-of-arrival (AoA) of the user's voice and the wall reflection. In EchoSpot [33], a

[1]This chapter is partially published in [71, 72]

TABLE 3.1: Infrastructure-free acoustic indoor localization and mapping.

Approach	**Study**	**Objective**	**Presumption on reflectors**	**Labeled training data**	**Resolution**	**Sensing technique**		
						Mode	**Sensing signal**	**Duration**
Analytic	VoLoc [73]	Localization	Yes	No	Decimeters	Passive	Human voice	15 cmds
	EchoSpot [33]					Active	18-23kHz FMCW[1]	0.2 s
	BatMapper [32]	Mapping					8-16kHz chirps	0.04 s
	SAMS [31]						11-21kHz FMCW	0.03 s
Fingerprint	SurroundSense [74]	Localization	No	Yes	Semantic locations	Passive	Acoustic background	60 s
	Batphone [26]				(Sub-)room-level			10 s
	RoomSense [28]					Active	0-24kHz MLS[2]	0.68 s
	DeepRoom [30]						20kHz tone	0.1 s
	EchoTag [29]	Location tagging			Centimeter		11-22kHz chirp	0.42 s
	ELF-SLAM	Localization & mapping		No	Sub-meter		15-20kHz chirp	0.1 s

[1] Frequency-modulated continuous-wave [2] Maximum Length Sequence

device emits near-inaudible acoustic signals and analyzes the times of flight of the signals reflected off the human body and a nearby wall for user spotting. These studies [33, 73] require presumption on the sound reflectors for triangulation.

The analytic approach can be also applied for indoor mapping by estimating the distances of the smartphone to the nearby surfaces (e.g., sidewalls, ceilings, and floors) [31, 32]. To build the wall contour map, the studies [31, 32] require the user to walk straight along the walls. They detect up to two walls simultaneously. In this work, maps are aimed to be built on arbitrary trajectories with loop closures and complete floor maps in indoor spaces. In addition, it is expected to address complex spaces such as a lab with dense cubicles and a shopping mall with dense stores. The work [75] considers collated omnidirectional speaker and microphone mounted on a robot and analyzes echoes' arrival times for indoor mapping. The evaluation in [75] is only based on simulations. In the real world, complex surroundings (e.g., many nearby objects) may bring challenges to echo arrival identification.

Vis-à-vis analytic approach, the *fingerprint approach* collects training samples from different indoor locations and trains a machine learning model for online location inference [26, 28–30, 74]. Early studies [26, 28, 74] apply shallow learning and require either long recording times that may cause privacy concerns or full-spectrum recording that is susceptible to interference like human voice [30]. The work [30] applies deep learning to reduce the requirements on recording time and spectrum usage. These studies [26, 28, 30, 74] address semantic or room-level localization. The work [29] uses echoic fingerprints to tag up to 11 locations with centimeters resolution. When the fingerprint approach is applied for the whole indoor space, the blanket process of collecting labeled fingerprints at many locations causes high overhead.

Acoustic sensing has also been used for other applications like inter-smartphone ranging [76, 77], finger [78, 79] and body [80] tracking, gesture recognition [81], speaker authentication [82], breathing rate estimation [83], and lung function monitoring [84].

■ **SLAM:** SLAM constructs the map of an environment in terms of a certain signal and localizes the user device simultaneously. Any SLAM solution consists of two components: *front-end signal processing* and *back-end pose-graph optimization*. Here, existing SLAM solutions according to their used sensing modalities and the used loop closure detection methods are discussed. Radar SLAM [85] and Lidar SLAM [86] are based on point clouds generated by radar and high-profile lidar, which are unavailable on smartphones. Visual SLAM [87] uses the camera to capture images for landmark detection and map construction. The imaging may introduce privacy concerns. The imaging may introduce privacy concerns. Wi-Fi SLAM [88, 89] employs the received signal strength indicators (RSSIs) from nearby Wi-Fi access points. However, Wi-Fi RSSI can be time-varying. Geomagnetic SLAM [90] exploits the spatially varying magnetic field. Electromagnetic radiation (EMR) SLAM [91] can use the smartphone's earphone as a side-channel sensor to sense the EMR from the alternating current power network. However, the side-channel sensing may experience weak signal strength when the earphone is away from the powerlines. This thesis's evaluation will employ geomagnetic, EMR, and Wi-Fi SLAMs as the main baselines. Acoustic SLAM [92] constructs a map based on the AoAs from multiple infrastructural sound sources. ELF-SLAM is a new SLAM solution based on infrastructure-free acoustic echoes.

Different sensing modalities need specific loop closure detection methods. The DNN-based loop closure detection on Lidar and Visual SLAM [93, 94] often outperforms the hand-crafted solutions. Wi-Fi SLAM [88] applies the Gaussian process latent variable model to determine the locations of the Wi-Fi RSSI signal. Both the geomagnetic and the EMR SLAM systems employ dynamic time warping (DTW) for loop closure detection. Different from these sensing modalities, generic acoustic features are ineffective to determine locations. Thus, CL is applied to learn a new acoustic feature embedding called ELF to effectively signal loop closures.

3.2 Preliminaries

■ **Excitation signal:** The commonly used excitation signals include single tone, frequency hopping spectrum spread (FHSS), and chirp [95]. Single tone facilitates detecting Doppler shift and is usually adopted in Doppler ranging and tracking [81]. However, in this work, a multi-frequency excitation signal is preferred such that the echoes carry richer information about the surroundings for location fingerprinting. FHSS signal has a rapid hopping frequency in a wide band. As it better copes with self-interference, it is suitable for short-range active sensing [79]. When exciting sizable indoor spaces, self-interference can be easily avoided by limiting

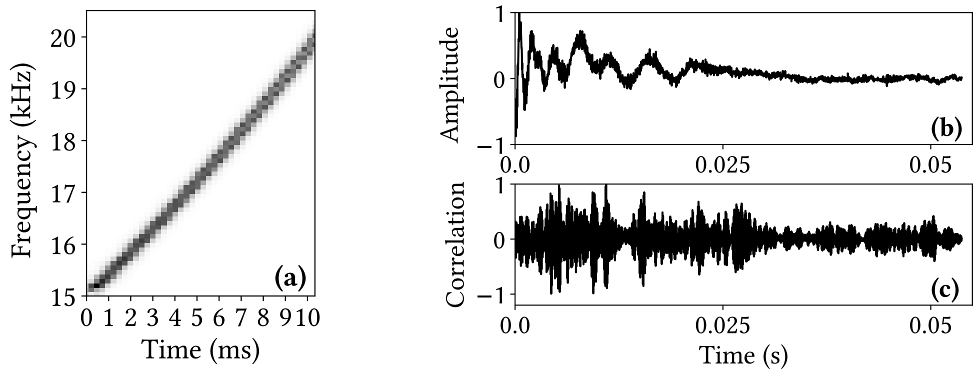


FIGURE 3.1: (a) Excitation chirp's spectrogram; (b) Received signal in time domain; (c) Pearson correlation between received signal and excitation signal template.

the excitation signal duration. Therefore, in this thesis, chirp with frequency varying with time continuously that can induce information-richer echoes is adopted, compared with FHSS.

As commodity smartphones support audio sampling at 44.1 or 48 ksps, they can capture acoustic frequencies up to 22.05 or 24 kHz. To reduce annoyance to human users, the excitation signal should be in the inaudible or near-inaudible frequency range. Although nominal audible frequency is up to 20 kHz, the audible limit of average adults is usually 15 to 17 kHz [96]. Signals with frequencies higher than 20 kHz often suffer from drastic distortions due to the smartphone audio hardware's nonlinearity in the inaudible band [30]. In this thesis, a near-inaudible logarithmic chirp sweeping the 15–20 kHz band within 10 ms is adopted as the excitation signal. Fig. 3.1a shows the spectrogram of the excitation chirp. The chirp uses a relatively wide band (i.e., 5 kHz) for the benefit of pulse compression [97], which helps capture fine-grained spatial features.

■ **Echo extraction:** A program is developed to use the smartphone's loudspeaker to emit the excitation signal and microphone to record the echo at 44.1 ksps for 100 ms. It is assumed that the smartphone is held around 30 to 40 cm in front of the chest. Note that the data collection cannot be unobtrusive, e.g., put the smartphone in a pocket. In the received data, the first 10 ms is discarded due to the propagation via the direct path from the loudspeaker to the microphone. The subsequent 1 ms data is also discarded, which usually contains the first reflection from the human body that is around $30-40$ cm away from the microphone. The

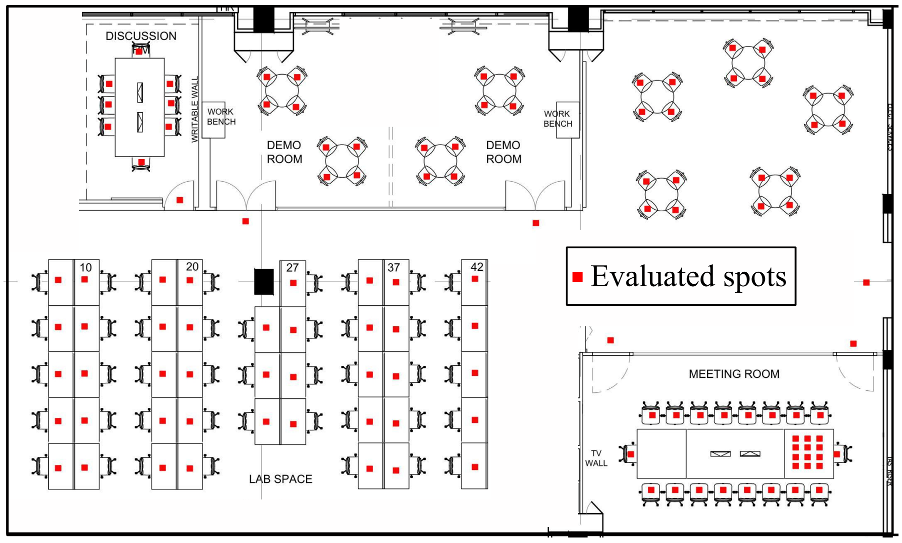


FIGURE 3.2: Floor plan of the lab space.

subsequent 50 ms data, which are collectively referred to as *echo trace* and illustrated in Fig. 3.1b, are used for SLAM. Fig. 3.1c shows the Pearson correlation between the received signal shown in Fig. 3.1b and the chirp template. The peaks in Fig. 3.1c indicate echoes.

3.3 Measurement Study

To gain insights into the system design, a set of measurement experiments is conducted in a $16 \times 28\,\mathrm{m}^2$ lab space as shown in Fig. 3.2. A Google Pixel 4 smartphone is used to excite the space at 128 spots indicated by red squares in Fig. 3.2 and collect 1,700 echo traces at each spot. This section presents the analysis results for these traces.

3.3.1 Spatial Distinctness of Echoes

The short-time Fourier transform (STFT) is applied to the echo data to extract the spectrogram. Specifically, a 96-point Hann window with 48 points of overlap is slided on echo data, resulting in a 49×48 spectrogram. The frequency bins below 15 kHz are further discarded, yielding a 12×48 image as the final result. Fig. 3.3 shows the echo spectrograms collected at three different spots. The frequency

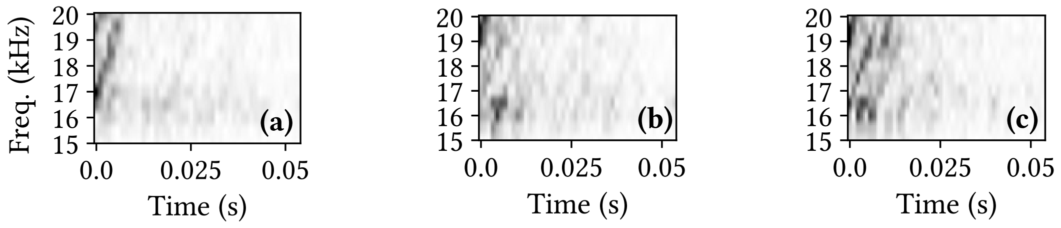
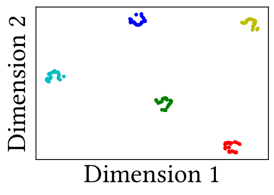
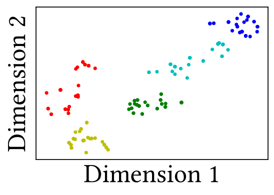


FIGURE 3.3: Echo spectrograms obtained at three locations. (a) center of the room, (b) the left side of the room, and (c) the right side of the room. The color bar indicates the normalized energy intensity at each frequency bin.



(A) Chirp excitation



(B) Single-tone excitation

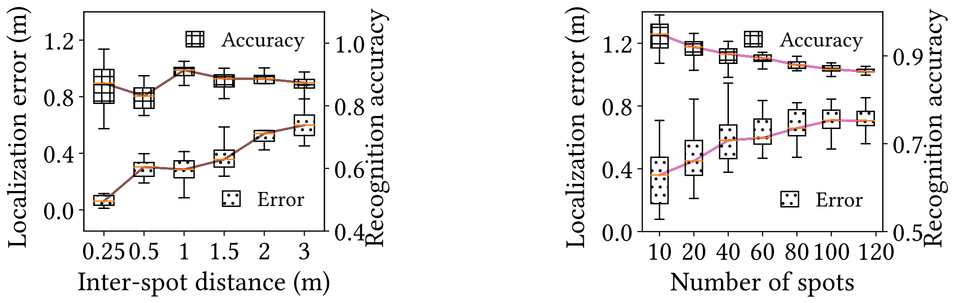
FIGURE 3.4: The t-SNE features of the echo spectrograms at five locations. Color represents locations.

dimension of a spectrogram is from 15 to 20 kHz. The differences among the spectrograms suggest that echoes vary across locations. The t-distributed stochastic neighbor embedding (t-SNE) [46] is applied to reduce the dimension of the spectrograms collected at five spots with 1 m separation in a linear topology. Fig. 3.4 shows the results when chirp and single-tone excitation signals are used. With the t-SNE features, spatial distinctness can be visualized. From the figure, compared with single-tone excitation, the chirp excitation leads to more individually-compact and mutually-separated t-SNE feature clusters. This suggests that the chirp excitation brings more spatial distinctness of echo.

To understand the distinctness limit, the task of supervised learning-based localization is used to investigate the achievable spatial resolution and scalability with respect to the number of spots.

3.3.1.1 Spatial resolution.

For each location, the spectrograms of the 1,700 echo traces are divided into 1,500 training samples and 200 test samples. The ground truth labels correspond to the spot's location. To understand how the inter-spot distance affects localization accuracy, 128 spots are divided into multiple groups with different densities. As



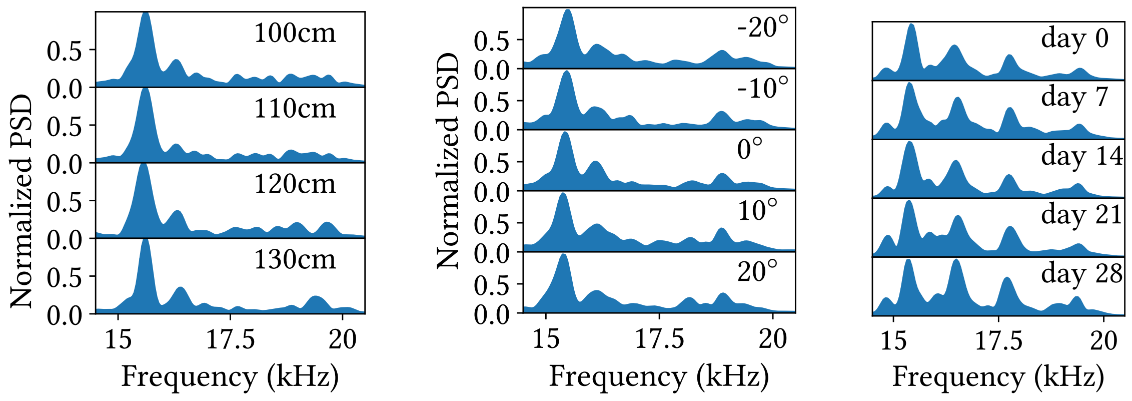
(A) Resolution (B) Scalability

FIGURE 3.5: Distinctness limit of acoustic echoes.

a result, the average inter-spot distances of the groups range from 0.25 m to 3 m. For each group, echo spectrograms are used to train a ResNet-18 DNN to classify the spots. Both the spot recognition accuracy and the mean localization error are measured. Fig. 3.5a shows the measured results versus the average inter-spot distance. For each group, the evaluation process is repeated 20 times and plot the error bars. The recognition accuracy remains at around 90%. The mean localization error increases with the inter-spot distance and remains at the sub-meter level. The results suggest that the acoustic echoes can achieve sub-meter spatial resolution.

3.3.1.2 Scalability.

The number of spots handled by a single ResNet-18 model (denoted by k) is increased to understand the scalability. For each setting of k, k spots are randomly drawn from the 128 spots to train and test a ResNet-18 model. Note that the selected spots become denser for a larger k. The process is repeated 20 times for each k setting. Fig. 3.5b shows the results. The recognition accuracy gradually decreases with k and becomes flat when k exceeds 100. This result is consistent with the intuition that the complexity of learning using a DNN increases with the number of classes. The mean localization error increases with k but remains under 1 m and thus at the level of the spatial resolution limit shown in §3.3.1.1. The above results suggest that the ResNet-18 model does not present a bottleneck when the number of spots is up to 128.



(A) Altitude (B) Orientation (C) Data age

FIGURE 3.6: Power spectral densities (PSDs) of the acoustic echoes when several factors vary.

3.3.2 Robustness of Acoustic Echoes

This section investigates the impacts of several potential affecting factors on acoustic echoes.

3.3.2.1 Altitude.

A researcher holds the phone at different altitudes to simulate the cases where users of different heights hold the phone with natural arm gestures for indoor navigation. As typical adult heights are within 150–194 cm [98], altitudes from 100 cm to 130 cm (i.e., two-thirds of user height) are tested. Fig. 3.6a shows the power spectral densities (PSDs) of the acoustic echoes at different altitudes of the same spot. It is observed that the altitude variations of less than 30 cm introduce little impact. Hence, the acoustic echoes are insensitive to variations in user height and hand altitude.

3.3.2.2 Phone orientation.

As a smartphone's loudspeaker and microphone are not perfectly omnidirectional, phone orientation may affect the received signal at a spot. The stability of the received echoes when phone orientation varies is evaluated. Fig. 3.6b shows the echo PSDs when the phone has an orientation deviation of $-20°$ to $20°$ from a certain direction. The results show that the echoes change slightly with less than $40°$ orientation deviation. When the orientation deviation is larger, the acoustic echoes exhibit larger differences. This suggests that the impact of phone orientation needs to be properly dealt with when there are multiple echo traces collected at the same spot but in different phone orientations. The solution to this issue will be presented in §3.4.5.

3.3.2.3 Temporal stability.

A smartphone is placed at a fixed location to emit and receive the chirp signal in a meeting room. 10 minutes of acoustic echoes are collected every day for one month. During this period, the indoor layout has no significant changes. To compare the echoes, FFT is applied to the raw echo data to extract PSDs, Fig. 3.6c shows the PSDs on different days. It is observed that the echo PSDs are consistent over the period. In practice, the constructed floor map can be updated whenever a user contributes a trajectory map. The continuous update helps address the potential aging issue over longer periods. In §3.3, the impact of significant changes in the indoor space layout (e.g., furniture re-arrangement) will be evaluated on the system and a mitigation approach beyond map update will be proposed.

3.4 Design of ELF-SLAM

From the measurement study, the acoustic echoes exhibit sub-meter spatial distinctness, which is the basis of the fingerprint approach. To unleash the fingerprint approach from laborious labeled training data collection, this section presents the design of ELF-SLAM based on acoustic echoes and IMU data captured by a smartphone during movements. In this section, §3.4.1 overviews the design of ELF-SLAM. §3.4.2 presents the graph-based SLAM formulation. §3.4.3 introduces ELF for loop closure detection. §3.4.4 presents a clustering-based approach for loop closure curation. §3.4.5 presents the trajectory map superimposition. §3.4.6 presents ELF-based localization.

3.4.1 Approach Overview

As illustrated in Fig. 3.7, the mapping phase of ELF-SLAM consists of two major components: *trajectory map construction* and *trajectory map superimposition*,

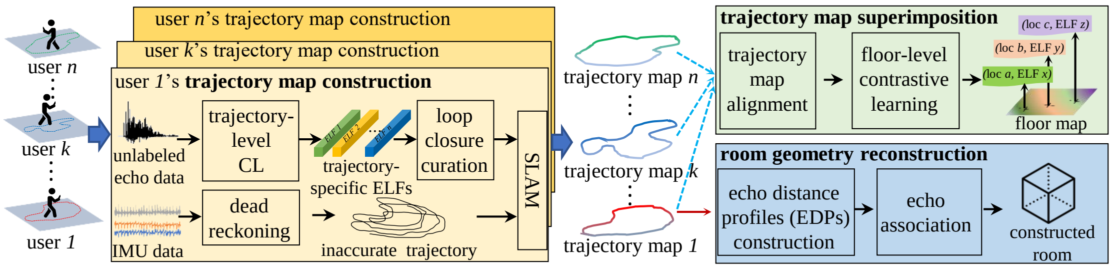


FIGURE 3.7: Overview of the mapping phase of ELF-SLAM.

where the former focuses on a single trajectory and the latter combines all available trajectories. The trajectories only cover a portion of the indoor space. However, when sufficient trajectories are collected, the combined map can cover the popular locations in the indoor space.

■ **Trajectory map construction:** A user holds a smartphone and moves within the target indoor space to collect the acoustic echoes and IMU data simultaneously on the movement trajectory. The IMU data is used to reconstruct the user's trajectory via dead reckoning, while the acoustic echo data is used to detect loop closures on the user's trajectory. The detected loop closures provide critical regulation for the dead reckoning to deal with its long-run drifting problem. Since the generic acoustic features, such as PSD, spectrogram, etc, are ineffective for loop closure detection, ELF-SLAM applies CL to learn a custom and trajectory-specific ELF for loop closure detection. This trajectory-level CL consists of model *pre-training* using synthetic data from a room acoustics simulator and *fine-tuning* using real data collected by the user from the target indoor space. ELF-SLAM detects loop closures based on a custom similarity metric called echo sequence similarity (ESS) between two sequences of ELF traces. Then, a clustering-based approach removes the false positive detection results to curate the loop closures. Lastly, a graph-based SLAM algorithm constructs an accurate trajectory map of ELFs for the user.

■ **Trajectory map superimposition:** When multiple trajectory maps are available (e.g., through crowdsensing), they are superimposed to generate a floor map. The superimposition reconciles different trajectory maps' ELFs that are collected at the same spot but in different phone orientations. To achieve this, different users' trajectory maps are first aligned into a common coordinate system. The alignment can be achieved based on known initial positions of the users' trajectories (e.g., the entrance of the space) and/or prior knowledge about the accessible passages of the target indoor space [99]. Then, the floor-level CL is applied to train a floor-wide ELF extractor using the acoustic data from all the trajectory maps. The floor map consists of the floor-level ELFs at all spots covered by all the trajectory maps, where each spot is associated with a single floor-level ELF.

Once a map is available, a smartphone can be localized based on its captured echoes in response to a few chirps emitted by the phone. Specifically, the smartphone extracts the ELFs of the echoes and compares them against the map to estimate its location.

3.4.2 Graph-based SLAM Formulation

Graph-based SLAM [100] constructs a graph whose nodes represent the mobile's poses and edges represent the kinetic constraints relating two poses. When using IMU data to establish the kinetic constraints, loop closure is a vital regulation in combating the long-run drifting problem of IMU-based odometry [101]. In this paper, by letting $\mathbf{x}_k$ denote the node (i.e., location) corresponding to the k^{th} detected footstep, the acoustic echo trace captured between the k^{th} and $(k+1)^{\text{th}}$ footsteps is the measurement associated with the node $\mathbf{x}_k$ and used to detect whether $\mathbf{x}_k$ is at the same location as any previous node (i.e., loop closure detection). The edge connecting two nodes is associated with the IMU-based odometry. The user trajectory is estimated via the graph-based optimization after the loop closures are identified. The estimation method is as follow. For a total of N detected footsteps, let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ denote the sequence of nodes that describes the user trajectory and $\mathbf{u}_{i,j}$ denote the edge constraint between nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ based on the IMU data. Let $\mathcal{C}$ denote the set of footstep index pairs of the detected loop closures. The essence of the trajectory reconstruction can be described by the following optimization problem:

$$\mathbf{X}^* = \operatorname*{argmin}_{\mathbf{X}} \sum_{\forall i \in [1,\ldots,N-1]} \|f(\mathbf{x}_i, \mathbf{u}_{i,i+1}) - \mathbf{x}_{i+1}\|^2 + \sum_{\forall \langle i,j \rangle \in \mathcal{C}} \|f(\mathbf{x}_i, \mathbf{u}_{i,j}) - \mathbf{x}_j\|^2,$$

where $\|\mathbf{x} - \mathbf{y}\|$ denotes the Euclidean distance between two locations $\mathbf{x}$ and $\mathbf{y}$, $f(\mathbf{x}_i, \mathbf{u}_{i,j})$ represents the prediction of $\mathbf{x}_j$ based on $\mathbf{x}_i$ and $\mathbf{u}_{i,j}$. In this paper, the SLAM algorithm is implemented based on the general graph optimization framework in [102], which also addresses the uncertainty of the prediction $f(\mathbf{x}_i, \mathbf{u}_{i,j})$.

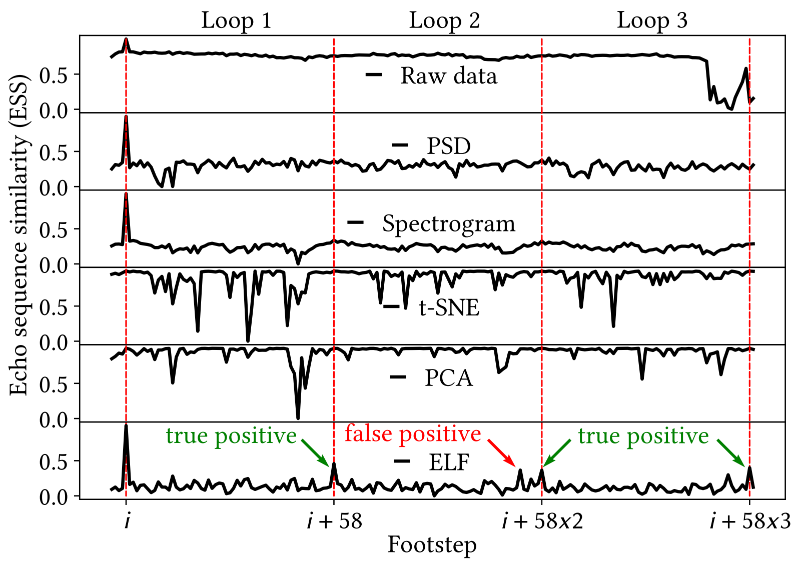


FIGURE 3.8: ESS traces with respect to footstep i. Peaks indicate loop closures at footstep $i + 58$, $i + 58 \times 2$, and $i + 58 \times 3$.

3.4.3 ELF For Loop Closure Detection

Identifying an effective feature for loop closure detection is critical to SLAM. In this section, the ineffectiveness of the generic features is demonstrated. Then, using CL to construct a learning-based feature is proposed.

3.4.3.1 Ineffectiveness of generic features.

A controlled experiment is conducted to evaluate several generic acoustic features. A researcher walks four rounds by following 58 markers pasted on the floor of the lab shown in Fig. 3.2 and uses a phone to collect acoustic data. As such, each round consists of exactly the same 58 footsteps. The following features of the echo data are computed: PSD, spectrogram, t-SNE, and principal component analysis (PCA). Then, the similarity between the features collected at footstep i in the first round with those at all footsteps in the four rounds is computed. The similarity is measured by the echo sequence similarity (ESS), which is defined as follows. For two footsteps i and j at which K_i and K_j features are collected, the ESS between them is obtained by averaging the $K_i \times K_j$ pair-wise cosine similarity among the two sets of features. Fig. 3.8 shows the resulting ESS traces with footstep i (where $i = 4$) in the first round and j being all footsteps of the four rounds sequentially. In this controlled experiment, with respect to the footstep i, loop closures are formed

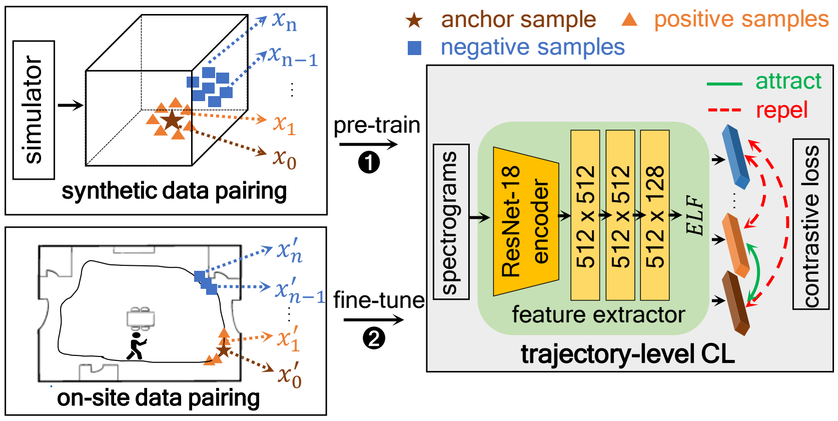


Figure 3.9: Trajectory-level CL to learn trajectory-specific ELFs.

at the footsteps $i + 58$, $i + 58 \times 2$, and $i + 58 \times 3$. If the used feature is effective, ESS peaks should be observed at these footsteps. However, from the plots in the first five rows of Fig. 3.8, no salient peaks are observed at these footsteps. This suggests that the raw data and the used generic feature extraction techniques are ineffective for our loop closure detection problem. Note that although t-SNE is effective for finding feature embeddings of clustered data [103], it is ineffective on the echo samples collected in the spatial continuum that do not exhibit clustered patterns. Note that the ESS peaks are also not observed under other similarity metrics, e.g., those related to Euclidean and Manhattan distances, etc.

3.4.3.2 Learning-based ELF.

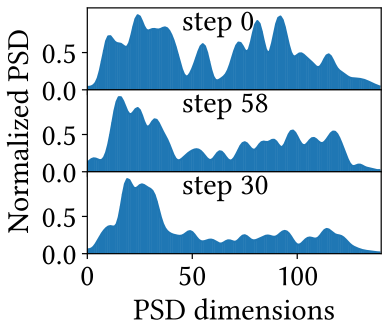
The ineffectiveness of generic features motivates us to apply CL to construct a custom feature, i.e., ELF. In what follows, CL design to construct the ELF for loop closure detection is presented. Fig. 3.9 depicts the workflow. It consists of three steps: *data pairing*, *model pre-training*, and *model fine-tuning*.

Data pairing constructs positive/negative data pairs needed by CL. In image recognition tasks, the positive samples are constructed by introducing spatial perturbations such as resizing, cropping, and blurring, which do not erase the information needed for image recognition. However, such spatial perturbations are not applicable to the echo data, because they destruct the subtle structural information embedded in the echo signal that is related to the smartphone's location. The design of our data pairing approach is based on the observation that the echoes are similar if collected at close locations and distinct if collected at locations apart.

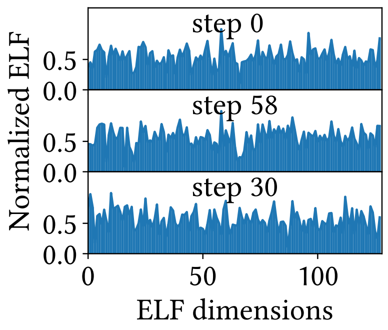
Thus, positive pairs are constructed using echoes collected at close locations and negative pairs are constructed using echoes at locations apart. Specifically, two consecutive echoes are treated as a positive pair. For each training step, a training batch of 256 such positive pairs is randomly sampled from the entire sequence of echoes collected by a certain user. According to our design in §3.3, the time gap between two consecutive echoes is 0.1 s. As the average human walking speed is 5 km/h, the locations for collecting two consecutive echoes are separated by 0.14 m on average. This average separation is slightly lower than the achievable spatial resolution of the echo modality as evaluated in §3.3. Thus, viewing two consecutive echoes during the user's movement as a positive pair is a good heuristic. Then, the cross pairing of the members of the 256 positive pairs gives the negative pairs. Note that the chance of wrongly forming a negative pair with two echo samples collected at a loop closure is low. Such limited wrong forming of negative pairs will not devastate the CL.

Model pre-training exploits self-supervised learning to build a basic ELF extractor, which will be specialized by the model fine-tuning step presented later. Self-supervised learning often requires abundant unlabeled training data for feature representation learning. To reduce the overhead of data collection in real environments, The `pyroomacoustic` [104] is used, which is a room acoustics simulator, to generate abundant synthetic training data. The `pyroomacoustic` employs the image source model [105], a widely used technique to simulate the sound propagation for each microphone and speaker location in an indoor space. Specifically, a smartphone is simulated by separating a speaker and a microphone by 15 cm. Massive echo data is generated at fine-grained grid points in various simulated rooms with different shapes and sizes. The synthetic echoes are location discriminative and can be used to train a feature extractor DNN by CL. The feature extractor architecture from [106] is used, which consists of a ResNet-18 encoder and a 3-layer projection head. The extractor maps the input spectrogram of an echo trace to a 128-dimensional ELF. As the locations of the synthetic echoes are controlled and thus known, the synthetic echoes collected at locations separated by less than 20 cm are treated as positive pairs and the rest as negative pairs. The following contrastive loss adopted from [107] is minimized during the model pre-training:

$$\ell_{i,j} = -\log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)/\tau\right)}{\sum_{k=1}^{2M} \mathbf{1}_{[k \neq i]} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/\tau\right)},$$



(A) sim(0,58)=0.40, sim(0,30)=0.42



(B) sim(0,58)=0.61, sim(0,30)=0.22

FIGURE 3.10: (a) PSDs (b) ELFs visualization at steps 0, 30, and 58.

where $\mathbf{1}_{[k \neq i]} \in \{0, 1\}$ is evaluated to 1 if and only if $k \neq i$, $\text{sim}(\,\cdot\,,\,\cdot\,)$ denotes cosine similarity, $\boldsymbol{z}$ is the extracted feature vector, i and j indicate a positive pair, M is batch size, and τ is the temperature parameter. With the above contrastive loss, the pre-training increases the feature similarity for echoes at close locations and decreases the feature similarity for those at locations apart. Thus, it builds a feature extractor that discriminates the echoes' locations under the cosine similarity metric. As a result, the loop closure detection can be implemented by comparing the ELFs produced by the feature extractor in terms of the cosine similarity. The necessity of the model pre-training will be evaluated in §3.6.

Model fine-tuning uses a small amount of unlabeled data collected by users in a specific target space to adapt the pre-trained model to capture the environment-specific characteristics. The same self-supervised CL procedure described above is followed to construct the data pairs to fine-tune the feature extractor. The resulting feature extractor can generate trajectory-specific ELFs for loop closure detection in the target space.

3.4.3.3 Loop closure detection using ELF.

The last row of Fig. 3.8 shows the ESS trace computed using ELF. It shows peaks at footstep $i+58$, $i+58\times 2$, and $i+58\times 3$ as marked by the green arrows. Fig. 3.10 shows the cosine similarities using echo features from steps 0, 30 and 58. For PSD, the cosine similarity between loop closure steps 0 and 58 is 0.4, and between the non-loop closure steps 0 and 30 is 0.42. PSD cannot differentiate the loop closure and non-loop closure data. Thus, PSD is ineffective for loop closure detection. For ELF, the cosine similarity between loop closure steps 0 and 58 is 0.61, and between the non-loop closure steps 0 and 30 is 0.22. The ELFs can see significant differences on the loop closure and non-loop closure data. Thus, ELF is effective for loop closure detection.

However, an unexpected peak close to the footstep $i + 58 \times 2$ as marked by a red arrow is also observed, which may lead to a false positive loop closure detection in Fig. 3.8. Unfortunately, the SLAM is often sensitive to false positive loop closures – some false positives can degrade the SLAM performance [91]. Thus, a loop closure curation algorithm is needed to remove the false positives.

3.4.4 Loop Closure Curation

3.4.4.1 Approach design.

A clustering-based loop closure curation approach that is based on an ESS matrix defined as follows is proposed.

ESS matrix: Consider a user's trajectory consisting of N footsteps. The pairwise ESSs between any two footsteps form a $(N-1) \times (N-1)$ ESS matrix, where the $(i, j)^{\text{th}}$ element is the ELF-based ESS between the footsteps i and j. Thus, the ESS matrix is symmetric. A large ESS suggests a high similarity between the two involved footsteps' ELFs, signaling a potential loop closure. Our extensive experiments show that a threshold value of 0.4 for ESS can identify most true positive loop closures while capturing an acceptably low number of false positives to be removed shortly. The ESS matrix is binarized using the threshold, where the positive elements represent candidate loop closures. Fig. 3.11 shows an example of the binarized ESS matrix constructed using the ELFs collected in a shopping mall. The x- and y-axis represent the footstep index. The black dots in the ESS matrix represent the positive elements.

Clustering-based approach for loop closure curation: The goal of loop closure curation is to remove the false positives from the binarized ESS matrix. Due to the user movement, the true positives in the binarized ESS matrix form trend curves. For instance, consider an ideal case in which the user walks at a constant speed, the true loop closures of footsteps 0, 1, ... , and 10 are footsteps $0+L$, $1+L$, ... , and $10+L$, where L is the loop length. As a result, the $(0, 0 + L)^{\text{th}}$,

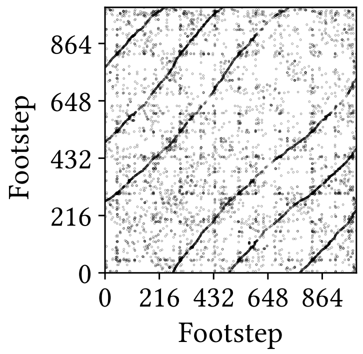


Figure 3.11: ESS matrix with true loop closures forming trend curves.

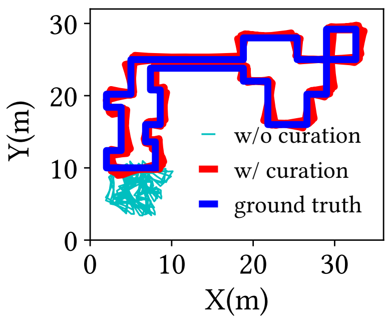


Figure 3.12: Reconstructed trajectories with/without loop closure curation.

$(1, 1 + L)^{\text{th}}$, ... , and $(10, 10 + L)^{\text{th}}$ elements of the binarized ESS matrix should be positives and form a trend line. In contrast, the false positives tend to appear at random positions in the ESS matrix, as shown in Fig. 3.11. Based on this observation, a clustering-based approach is proposed to isolate the trend curves formed by the true positives from the scattered false positives.

First, the binarized ESS matrix is divided into multiple slices, as illustrated in Fig. 3.13a. With the slicing, it is easier to identify the true positive clusters in each slice. In our implementation, the slice width is set to 16 footsteps. Then, for the positives in each slice, the DBSCAN clustering algorithm [108] is applied to identify the number of loops and divide the positives into multiple clusters. This is illustrated by Fig. 3.13b, where the clusters are differentiated by colors. Although some false positives are classified by DBSCAN as outliers, the remaining false positives close to the trend curves are still in the clusters. To remove these remaining false positives, for the points in each cluster, the random sample consensus (RANSAC) [109] linear regression algorithm is applied to detect a line approximating the trend curve segment. RANSAC is a preferred regression algorithm when many outliers are present. Concatenation of the regressed line segments across all slices gives the clean trend curves. Fig. 3.13c shows the concatenated results, in which the trend curves formed by the positives are effectively isolated from the scattered noises as shown in Fig. 3.13a. Lastly, the loop closure candidates are further curated based on the ESS matrix's symmetric property. Since the negative impact of a false positive on SLAM outweighs that of a false negative, a strategy of only retaining the positives that conform to the symmetric property is applied. Specifically, if the positive at the $(i, j)^{\text{th}}$ position of the ESS matrix has

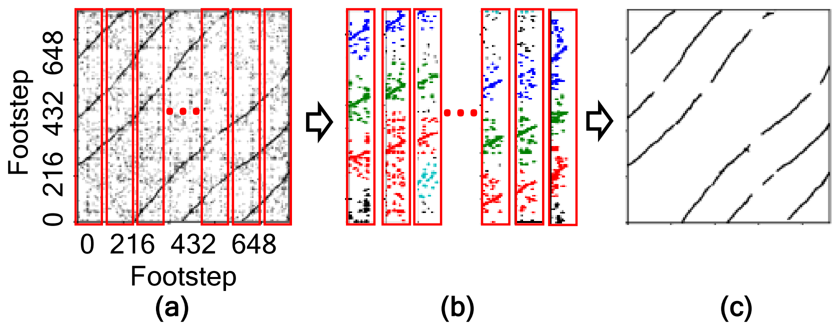


Figure 3.13: Clustering-based approach for loop closure curation: (a) ESS matrix slicing, (b) clustering in each slice, and (c) concatenated line regression results.

no counterpart positive at the $(j, i)^{\text{th}}$ position, the positive is viewed as false and excluded.

3.4.4.2 Effectiveness of loop closure curation.

To demonstrate the impact of the false positives on SLAM, all positives in Fig. 3.11 are used as the loop closure information to construct the trajectory map. The plot labeled ”w/o curation” in Fig. 3.12 shows the trajectory of the constructed map. It is observed that the false positives devastate the trajectory reconstruction. The plot labeled ”w/ curation” in Fig. 3.12 shows the reconstructed trajectory using the curated loop closures. The new trajectory highly resembles the ground truth. The result demonstrates the effectiveness of the proposed clustering-based loop closure curation.

3.4.5 Trajectory Map Superimposition

The trajectory map constructed from a single user’s data only contains the echo data on a specific trajectory. For real applications, it is desirable to combine many trajectory maps to form a floor map that covers most/all accessible locations. It is assumed that each trajectory map’s initial position relative to the indoor space is known. For instance, the SLAM mobile app may prompt the user to start the process from the entrance of the indoor space. When there are multiple entrances, location tagging [29] can be used to recognize the actual entrance. With the known initial position, the trajectory maps can be collated into a common coordinate system. However, the trajectory maps crossing a certain spot from different entering directions may have different echo data for the spot, due to the dependency of echo data on phone orientation as shown in §3.3. Thus, such differences need to be reconciled.

A floor-level CL approach is proposed to train a unified feature extractor for map superimposition. It shares the same model pre-training workflow as the trajectory-level CL except the data pairing approach for model fine-tuning. Specifically, the echo data collected at the same location regardless of the phone orientation are treated as positive pairs, whereas those collected from different locations are treated as negative pairs. The feature extractor trained via the floor-level CL generates the floor-level ELFs covering all trajectory maps. As the quality of the floor map is related to its spatial coverage, this floor-level CL approach needs to scale well with the number of locations. In §3.6, this approach is evaluated in handling 4,000 different locations with four phone orientations at each location.

When falling back to the scheme of learning a location recognition model in a supervised manner, a possible approach to mitigate the echo data's sensitivity on phone orientation is to form a training dataset with echo data and location labels (regardless of orientation) from all the trajectory maps. In §3.6, the localization performance of this supervised learning approach will be compared with the proposed approach based on the floor map.

3.4.6 Localization

Once a map (either a trajectory map or floor map) is constructed, a smartphone's location can be determined after capturing the echoes in response to the chirps. Two localization approaches are considered, i.e., *one-shot localization* and *trajectory localization*, which are suitable for the scenarios where the user stands still and moves, respectively. In the former, an ELF sequence containing multiple consecutive echoes collected at a spot is matched against the map in terms of the ESS to determine the location. In the latter, both the ELF sequence and the IMU data during the user's movement over a short time period are used for localization. Specifically, dead reckoning is applied to the IMU data to estimate the user's trajectory, and then apply a curve matching algorithm [110] to find the candidate segments in the map that resemble the user's trajectory. The candidate segment that has the largest average ESS from the captured ELF sequence is the output of the trajectory localization.

3.5 Room Geometry Reconstruction

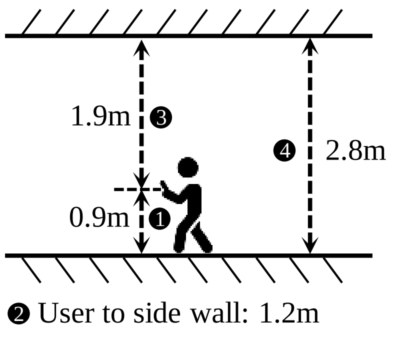
Accurate smartphone-based room geometry sensing is desirable for indoor navigation systems, virtual/augmented reality applications and network condition prediction, etc. In this section, the reconstructed user trajectory and the collected acoustic echoes are used to construct the contour of a polyhedron room with a fixed height. Specifically, the user is required to walk along the sidewalls and form a complete loop. After the IMU trajectory is rectified using the *trajectory map construction*, the wall distances are estimated using the acoustic echoes. Next, the room geometry is determined by the trajectory and the estimated wall distances. In what follows, the room reconstruction procedures are presented.

3.5.1 Wall Distance Measurement

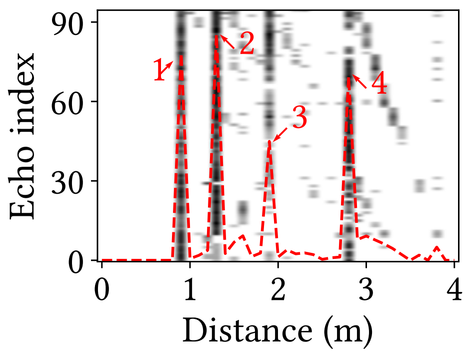
A measurement study is conducted to verify if the recorded echoes are effective for measuring the phone-wall distances. A user is asked to hold a smartphone and walk along a sidewall in a living room for a few meters. A total of 95 echo traces are collected. Fig. 3.16a shows the layout of the tested environment, where sofas, a table and a TV occupy the room. Fig. 3.14a shows the user's distances to the walls: the *phone-floor*, *phone-sidewall*, and *phone-ceiling* are 0.9 m, 1.2 m, and 1.6 m, respectively. The *room height* is 2.8 m.

Echoes extraction: The received signal is cross-correlated with the chirp template to detect the echoes reflected by the main reflectors in a room (e.g., walls, floor and ceiling). Fig. 3.15 shows an example, where the peaks represent the echoes. To detect the echoes, the envelope detector is applied to the correlated signals and searches for the local maximas. The reflectors' distance to the smartphone is calculated by $\frac{nc}{2f}$, where n is an echo's index, c is the speed of sound in air, f is the microphone's sampling rate (i.e., $44,100\,\text{Hz}$).

Echo selection: The cross-correlated signals contain many echoes generated by the nearby objects. It is difficult to associate each echo with its corresponding



(A)



(B)

FIGURE 3.14: (A) The user holds a phone and moves along a side wall at 1.2 m away. (B) Constructed echo distance profile.

reflector. However, for room reconstruction, only echoes from the main reflectors need to be identified, e.g., side walls, the ceiling, and the floor. The main reflectors can generate echoes with large amplitudes compared with smaller objects since they have larger reflecting surfaces. The correlated signals are normalized and an empirically determined threshold of 0.15 is applied to select the candidate echoes. In Fig. 3.15, the echoes marked by stars are selected and red crosses are discarded. Note that only peaks within 4m range from the smartphone are considered. This is based on the assumption that when a user walks along the sidewalls for data collection, the smartphone's distance to the sidewalls, ceiling, and floor can be maintained within 4m. In addition, the peak candidates with a distance larger than 4m are generally caused by the multi-path reflections, which are difficult for object association.

Echo distance profile: In each echo trace, the extraction and selection procedures described above are applied to generate the candidate echoes. Then, echoes selected from all echo traces are stacked to form an echo distance profile (EDP) as shown in Fig. 3.14b. The horizontal axis represents the echoes' distance to the smartphone and the vertical axis represents the echoes' indexes. The grayscale represents the echoes' amplitude, which ranges from 0.15 to 1. A darker dot has a higher amplitude. In Fig. 3.14b, four vertical lines are formed by the echoes. Lines 1, 2, and 3 located at the distances of 0.9 m, 1.2 m, and 1.6 m, respectively. These lines correspond to the distances of *phone-floor*, *phone-sidewall*, and *phone-ceiling*. Note that line 4 located at 2.8 m is also observed, whose distance is equal to the *room height*. This line is generated by the echoes that travel a full round in the vertical direction of a room (i.e., via smartphone → floor → ceiling → smartphone,

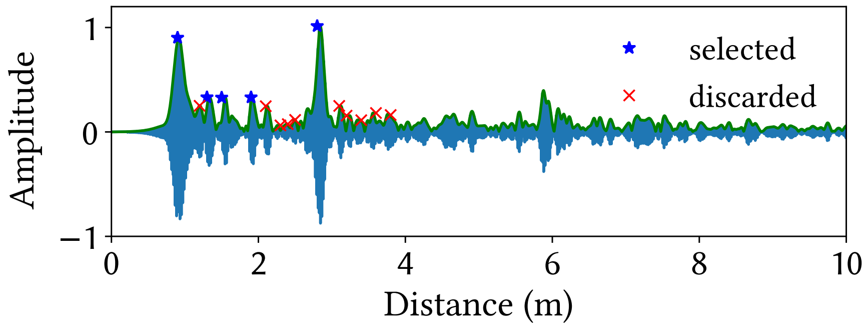


FIGURE 3.15: Echo detection on the cross-correlated echo.

or smartphone → ceiling → floor → smartphone). The red line in Fig. 3.14b represents the summation of the echoes' amplitude along the vertical axis. It is observed that the peaks corresponding to the distances of the *phone-floor*, *phone-sidewall*, *phone-ceiling*, and *room height* stand out in the EDP. The reason is that the walls' distance to the phone remains constant while a user is walking along a sidewall, while other objects' distance changes (e.g., TV, sofas, etc). As shown in Fig. 3.14b, although it is difficult to associate the peaks to the objects in a single echo trace, aggregation of echoes collected along a specific wall renders echoes from main reflectors more salient than those of the furniture inside the room. Thus, the EDP is effective to find the phone-wall distances along a sidewall.

3.5.2 Room Geometry Reconstruction Procedure

This section describes the room geometry reconstruction procedure. As shown in Fig. 3.7, the room reconstruction consists of *EDPs construction* and *echo association.*

EDPs construction: EDP is constructed for each sidewall to obtain wall distances. The wall numbers are determined based on the shape of constructed user trajectory. Note that the rectified user trajectory is used to get a more accurate approximation. The heading directions of the IMU data are tracked and the sheer direction changes are recorded as the corners between walls within a complete loop. The sidewall numbers are equal to the detected corners. Then, the echoes are split into clusters based on the timestamps of the detected corners. Since the IMU data and the echoes are collected simultaneously, each cluster contains the echo traces collected while the user walks along a specific sidewall. Echo traces in each cluster

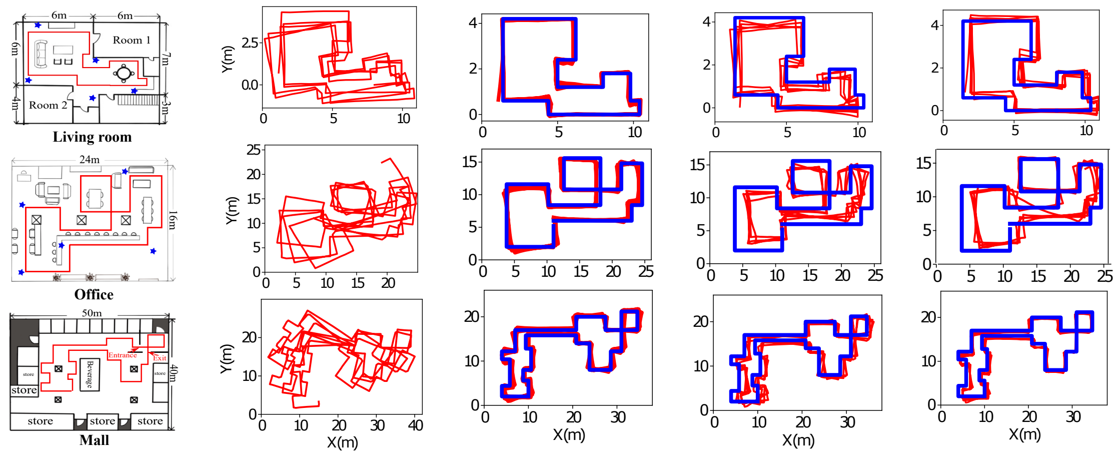
Table 3.2: Mapping error statistics (error in meters).

Modality	Living room			Office			Mall		
	$\tilde{x}$ [1]	$\bar{x}$ [2]	Q3 [3]	$\tilde{x}$	$\bar{x}$	Q3	$\tilde{x}$	$\bar{x}$	Q3
ELF	**0.10**	**0.10**	**0.14**	**0.63**	**0.63**	**0.80**	**0.45**	**0.53**	**0.69**
ELF w/o pre-train	0.73	0.82	1.25	1.69	1.68	1.94	1.16	1.14	1.42
Wi-Fi	0.44	0.45	0.55	1.52	1.54	2.06	1.24	1.26	1.54
Geomag	0.56	0.55	0.64	1.14	1.24	1.82	0.79	0.81	1.05

[1] Median error [2] Mean error [3] Third quartile of the error

are used to construct EDPs. If the echoes are correctly associated with the *phone-floor*, *phone-sidewall*, *phone-ceiling*, and the *room height* distances in EDPs, the room's geometry is also determined.

Echo association: To correctly associate the *phone-floor*, *phone-sidewall*, *phone-ceiling*, and the *room height* distances to the echoes in the constructed EDPs, the knowledge that the *room height* equals to the summation of the *phone-floor* and *phone-ceiling* distances is used. These three distances generally remain constant while a user holds the phone and moves within the room. Thus, these three distances can be determined in each EDP and then the subsequent largest peak is identified as the *phone-sidewall* distance. The procedure is as follows. First, EDPs from all sidewalls are combined to form a unified EDP (u-EDP). Since the *phone-floor*, *phone-ceiling*, and the *room height* distances remain consistent in each EDP, their appearance will be more salient in the aggregated u-EDP. In u-EDP, the echo with the largest peak is identified as *phone-floor* distance. This is because the used bottom microphone for recording is closer to the floor when held by a user. Thus, the peak amplitude at *phone-floor* distance generally has the largest value. Then, the *phone-ceiling* distance and *room height* are associated by looking for peaks that have the summation relationship with the identified *phone-floor* distance in the u-EDP. To reduce ambiguity, it is assumed the *phone-ceiling* distance is larger than *phone-floor* distance. Next, each EDP is visited and the echoes that are closest to the identified *phone-floor*, *phone-ceiling* and *room height* distances are excluded. The subsequent echo with the largest peak is identified as the *phone-sidewall* distance. Finally, the rectified trajectory and the estimated wall distances are used to determine the vertexes of the polyhedron.



(A) Floor plans (B) IMU (C) ELF (D) Wi-Fi (E) Geo

FIGURE 3.16: (A) Floor plans and ground-truth trajectory in three evaluation environments. (B) Trajectories estimated using IMU. (C)-(E) Rectified trajectories using ELF, WiFi RSSI and geomagnetic field. The blue curves are the ground truth trajectories and the red curves are the estimated trajectories.)

3.6 Performance Evaluation

3.6.1 Experiment Setup

Evaluation environments: ELF-SLAM is evaluated in three indoor environments: a living room ($60\,\mathrm{m}^2$), an office ($360\,\mathrm{m}^2$), and a shopping mall ($2,000\,\mathrm{m}^2$). The floor plans are shown in Fig. 3.16a. To conduct a comparative evaluation side by side, the SLAM systems using two smartphone's built-in sensing modalities are employed, i.e., Wi-Fi RSSI and geomagnetism, as the baselines. This is the same as the evaluation methodology adopted in [91] that studies powerline EMR SLAM. Note that the results of ELF-SLAM and EMR SLAM are also compared. To implement Wi-Fi SLAM, five Wi-Fi access points (APs) are deployed in the living room and office, as illustrated by the stars in Fig. 3.16a, that provide sufficient Wi-Fi coverage. The large shopping mall has dense APs deployed by the tenants. The number of Wi-Fi APs observable is around 5 to 10 when conducting experiments in the mall. Random people are walking in the shopping mall during the data collection.

Data collection: An Android app is developed and run on a Google Pixel 4 smartphone to collect acoustic echoes, Wi-Fi RSSI, geomagnetic field signals, and IMU data. The app uses the `WifiManager` Android API to scan the surrounding

Wi-Fi APs and collect RSSI data at a sampling rate of 0.8 sps. Wi-Fi channel state information (CSI) is not sampled, because CSI sampling requires rooting a phone [111]. The app also samples the phone's built-in magnetometer at 50 sps. During data collection, the user holds the phone with one hand around 30 to 40 cm in front of the chest and walks on a marked trajectory for multiple rounds in each of the evaluation environments. Note that the purpose of trajectory marking is to obtain the location ground truth.

Loop closure detection for baseline modalities: For Wi-Fi SLAM, the Euclidean distance between two Wi-Fi RSSI vectors is used for loop closure detection [112]. For geomagnetic SLAM, the triaxial magnetic data is normalized and then the dynamic time warping distance is used as the metric for loop closure detection [90]. The loop closure curation and graph-based optimization algorithms described in §3.4 are applied to the baseline SLAM systems.

3.6.2 Trajectory Map Construction Performance

Fig. 3.16a shows the floor plans and ground-truth trajectories. Fig. 3.16b shows the trajectories reconstructed via IMU-based dead reckoning, which deviate heavily from the ground truth due to the long-run drift problem. Fig. 3.16c, 3.16d, and 3.16e show the trajectories reconstructed by the SLAM systems using ELF, Wi-Fi RSSI, and geomagnetism. The trajectories reconstructed by ELF-SLAM are the closest to the ground truth. Table 3.2 lists the detailed mapping error statistics of the three modalities. ELF-SLAM achieves sub-meter mapping accuracy in all three environments, whereas Wi-Fi SLAM and geomagnetic SLAM's mapping errors increase in the large indoor space, i.e., office and mall. In [91], EMR SLAM using the smartphone earphone as the side-channel sensor yields about 1 m to 2 m median mapping errors in the evaluated office and lab spaces. Thus, ELF-SLAM outperforms Wi-Fi SLAM, geomagnetic SLAM, and EMR SLAM in map construction.

3.6.3 Effectiveness of CL Pre-training

The necessity of the CL pre-training is investigated by comparing the SLAM performance using the ELF extractors learned with and without the model pre-training.

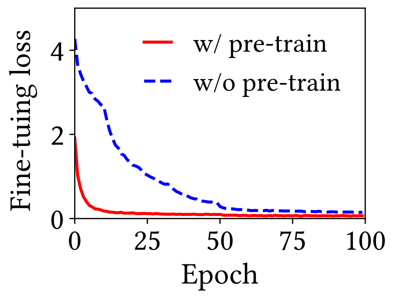


FIGURE 3.17: Trend of model finetuning loss.

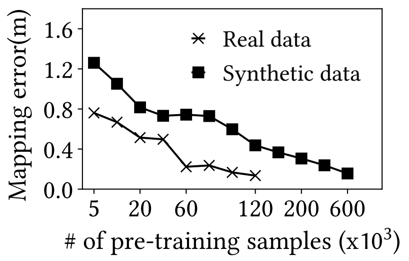


FIGURE 3.18: Impact of type and amount of pre-training data.

The row "ELF w/o pre-train" in Table 3.2 shows the case without model pre-training. Compared with the result with model pre-training (row "ELF"), the median errors increase to 0.73 m, 1.16 m, and 1.69 m, respectively. Fig. 3.17 shows the fine-tuning loss. The fine-tuning loss on the pre-trained model converges more quickly and is lower than that on the model without pre-training. The above results show that the model pre-training improves the efficiency of CL.

The model pre-training's requirement on data is also investigated. Two types of data are used for the pre-training, i.e., the synthetic data generated by the room acoustics simulator and real data collected from the different spaces. For the latter, real echoes collected in the lab space, office, and shopping mall are used for model pre-training. Then, the fine-tuning and evaluation are performed in the living room. Fig. 3.18 shows the median mapping error versus the amount of pre-training data. The mapping errors decrease with pre-training data volume for both types of data. If real data is used for pre-training, the median mapping error converges to about 0.13 m when data volume is more than 120 K. If synthetic data is used, the error converges to about 0.15 m when data volume is more than 600 K. This shows that both the real data and synthetic data are useful for model pre-training. Although real data shows better efficiency in supporting the pre-training, synthetic data can be generated at massive scales and thus suffice for the pre-training.

3.6.4 Impact of Finetuning Data Volume

The impact of the fintuning data volume on the performance of trajectory map construction is investigated. The used data volume for model finetuning is gradually decreased and the trajectory map construction results are shown in Fig. 3.19. In the living room, the median mapping error remains around 0.15 m when the used

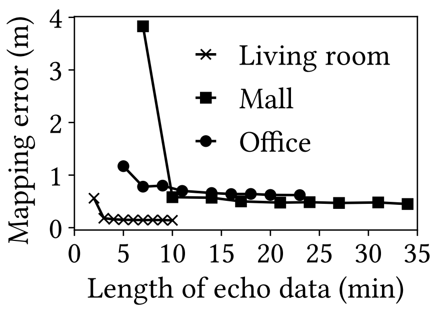


FIGURE 3.19: Impact of finetuning data volume.

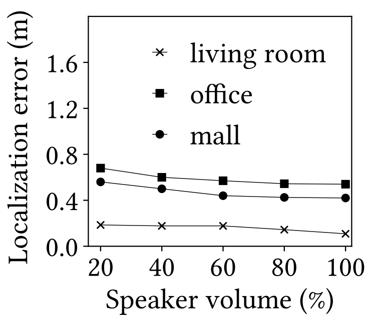


FIGURE 3.20: Impact of speaker volume.

TABLE 3.3: Localization error statistics (error in meters).

Modality	Living room			Office			Mall		
	$\tilde{x}$ [1]	$\bar{x}$ [2]	Q3 [3]	$\tilde{x}$	$\bar{x}$	Q3	$\tilde{x}$	$\bar{x}$	Q3
One-shot localization									
ELF	**0.10**	**0.29**	**0.14**	**0.54**	**0.60**	**0.80**	**0.42**	**0.79**	**0.67**
Wi-Fi	1.67	2.17	3.30	3.44	4.27	6.16	3.04	3.86	5.22
Geomag	1.06	2.31	3.93	2.19	3.95	5.38	12.48	13.36	19.28
Trajectory localization									
ELF	**0.10**	**0.22**	**0.64**	**0.41**	**0.54**	**0.97**	**0.47**	**0.53**	**0.86**
Wi-Fi	0.54	0.730	1.13	1.74	1.86	3.09	1.46	1.90	3.73
Mag	0.56	0.56	0.78	1.75	1.81	2.39	8.70	8.29	14.59

[1] Median error [2] Mean error [3] Third quartile of the error

data volume decreases from 10 min to 3 min. The error increases to 0.56 m when the used data volume is 2 min. The increased error is due to no loop closure formed and the estimated trajectory cannot be corrected. Similarly, the mapping errors remain low when the used data volume is larger than 7 min and 10 min where loop closures can be detected via the ELFs in the office and the mall, respectively. The results show that the ELF-SLAM only requires a few minutes of data for model finetuning as long as there is loop closure formed during the walking.

3.6.5 Localization Performance

Both the one-shot localization and trajectory localization of the three sensing modalities are evaluated. Table 3.3 lists the localization error statistics. For one-shot localization, ELF-SLAM achieves sub-meter median error in the three environments and outperforms both Wi-Fi SLAM and geomagnetic SLAM. For trajectory localization, each short trajectory consists of 8 consecutive footsteps. For Wi-Fi and geomagnetic SLAMs, the trajectory localization errors are less than the one-shot localization errors. For ELF-SLAM, trajectory localization does not

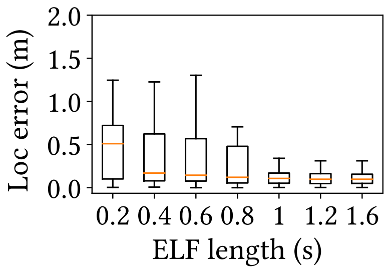


FIGURE 3.21: ELF sequence length.

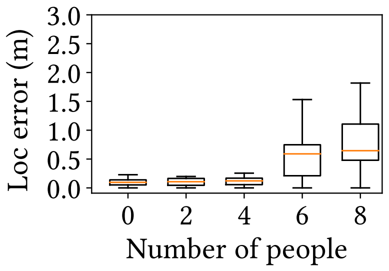


FIGURE 3.22: Moving people.

bring much accuracy improvement over the one-shot localization, because the latter has already achieved a high localization accuracy close to the modality's spatial resolution. However, it is shortly shown in §3.6.6 that the trajectory localization brings performance improvement when ELF-SLAM is affected by various affecting factors.

3.6.6 Sensitivity Analysis for Localization

Experiments are mainly conducted in the living room to evaluate the sensitivity of ELF-SLAM to various factors. By default, one-shot localization is considered.

3.6.6.1 Speaker volume.

The localization is evaluated with various settings for the smartphone's loudspeaker volume in emitting the chirps. Fig. 3.20 shows the results. When the volume decreases from 100% (i.e., the highest volume) to 20%, the median localization errors in the living room, office, and shopping mall increase from 0.1 m, 0.54 m, and 0.42 m to 0.18 m, 0.68 m, and 0.56 m, respectively. Note that with 20% loudspeaker volume, on the audible frequency band, the smartphone's sound is soft and becomes nearly imperceptible in environments with normal noise levels. Thus, ELF-SLAM maintains sub-meter accuracy when the chirp emission is soft.

3.6.6.2 ELF sequence length.

Fig. 3.21 shows the localization errors when the length of the ELF sequence used for computing ESS varies from 0.2 s to 1.6 s. A boxplot shows the localization error

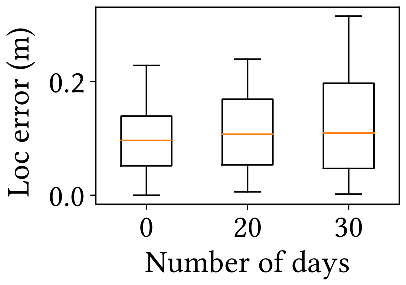


FIGURE 3.23: Aging.

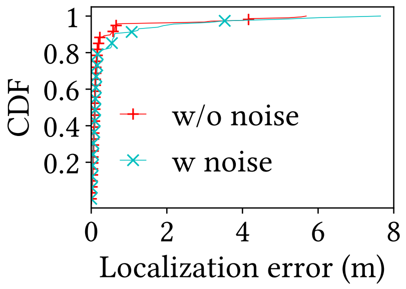


FIGURE 3.24: Audible noises.

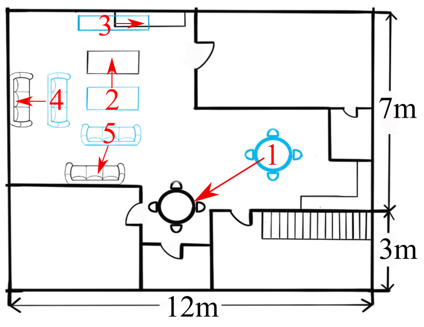
distribution. The horizontal line in each boxplot shows the median. It is observed that the localization error decreases with the ELF sequence length and becomes flat when the sequence length is more than 1 s. Note that at a human's average walking speed, the duration between two consecutive footsteps is about 0.6 s, which results in an ELF sequence length of 0.6 s as well. From Fig. 3.21, at this length setting, the one-shot localization median error is around 0.1 m. Thus, ELF-SLAM performs well when the user walks at a normal speed.

3.6.6.3 Nearby moving people.

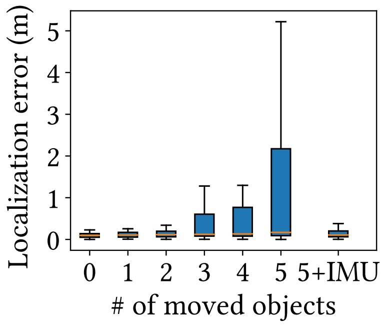
Human bodies can reflect the excitation chirp and generate echoes irrelevant to ELF. Thus, the impact of the nearby moving people on one-shot localization is evaluated. multiple volunteers are asked to walk freely in the living room and talk to each other during the localization phase. Fig. 3.22 shows the localization error versus the number of nearby moving people. The localization error remains low when the number of people is up to 4. Note that the tested area is only about 60 m^2. When there are 6 and 8 moving people, whose crowd density is similar to that in the shopping mall during peak hours, the median localization errors increase to 0.57 m and 0.6 m. Nevertheless, the errors remain at the sub-meter level. Thus, ELF-based localization can tolerate nearby moving people to a certain extent.

3.6.6.4 Map aging.

Whether the map constructed by ELF-SLAM ages is evaluated. Specifically, at day 0, ELF-SLAM is used to construct a trajectory map. Then, the ELF-based localization performance is evaluated multiple times during one month period. Fig. 3.23 shows the results. The median localization errors are 0.10 m, 0.11 m, and 0.12 m at day 0, 20, and 30, respectively. This suggests that the constructed map does not



(A)



(B)

FIGURE 3.25: (A) Movements of furniture objects in a living room; (B) the corresponding localization performance.

have a salient aging issue. In practice, a map can be continuously updated using the latest data contributed by users, to mitigate any potential aging issue.

3.6.6.5 Audible noises.

The robustness of ELF-based localization against audible noises is evaluated. A laptop computer is used to play video clips of different contents (music, speech, etc) from Youtube to generate the noises. From Fig. 3.24, the noises have little impact on the localization performance. This is because the system operates within the near-inaudible frequency band. Thus, audible noises have a negligible impact on ELF-based localization.

3.6.6.6 Space layout changes.

The layout changes of the target space may have an impact on the chirp reverberation processes. Thus, the furniture locations are deliberately changed in the living room to evaluate such impact. Fig. 3.25a illustrates how the furniture objects are moved. Specifically, five objects including a dining table, a tea table, a TV cabinet, and two sofas are moved. One object is moved at a time. Fig. 3.25b shows the localization error versus the number of moved objects. The error remains low when the number of moved objects is less than 5. When all 5 objects are moved, the mean localization error increases to 1.3 m. If the trajectory localization is applied, the mean localization error decreases to 0.3 m as labeled by “5+IMU” in Fig. 3.25b.

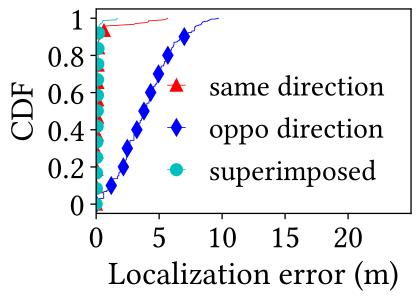


FIGURE 3.26: Trajectory map superimposition on small-scale real echoes.

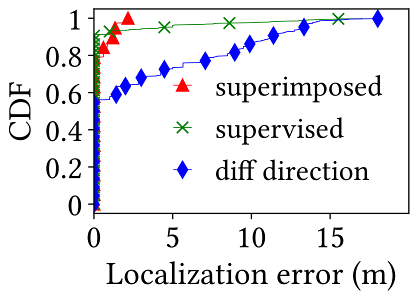


FIGURE 3.27: Trajectory map superimposition on large-scale synthetic echoes.

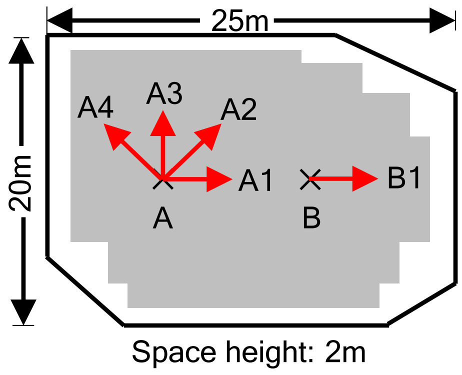
Therefore, the trajectory localization improves the robustness of ELF-based localization against layout changes. In practice, a map can be continuously updated using the latest data from the users to mitigate the impact of layout changes.

3.6.7 Trajectory Map Superimposition

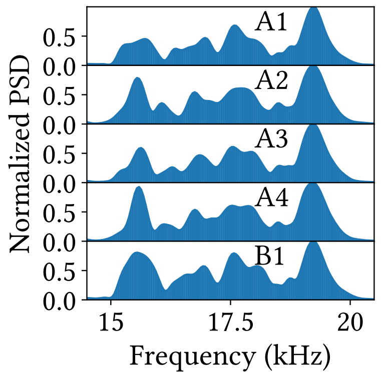
The performance of the one-shot localization is evaluated on the map superimposed from trajectory maps as described in §3.4.5.

3.6.7.1 Evaluation on a small-scale dataset.

Experiments are conducted in the living room. The route in Fig. 3.16a is followed and two opposite directions are walked to generate different trajectory maps. Then, the proposed CL approach is applied for map superimposition. Fig. 3.26 shows the localization results. The plot labeled “same direction” is obtained when the smartphone during the localization phase is in the same orientation as the used map. The median localization error is 0.1 m. The curve labeled “oppo direction” is for the case when the smartphone during localization is in the opposite orientation as the trajectory map. The median localization error increases up to 3.8 m. The increased error is due to the sensitivity of ELF to large phone orientation deviations. The curve labeled “superimposed” shows the localization results using the map superimposed via the CL approach. The median localization error is 0.1 m, which is the same as the “same direction” result. This small-scale experiment shows that the map superimposed via CL can improve the ELF-based localization performance when trajectory maps with opposite directions are available.



(A)



(B)

Figure 3.28: (A) Simulated space for extended evaluation of floor-level CL. (B) Spot A's echo PSDs on directions 1 to 4 and Spot B's echo PSD at direction 1.

3.6.7.2 Evaluation on a large-scale synthetic dataset.

This thesis also evaluates whether the floor-level CL can handle massive echo data from many trajectory maps. This experiment omits trajectory map construction and only focuses on evaluating the superimposition performance, in terms of the one-shot localization error using the floor map. To allow the evaluation to easily scale up, the `pyroomacoustic` simulator is used to generate a large-scale synthetic dataset for an indoor space as shown in Fig. 3.28a. Data are collected at 4,000 spots in the grey area.At each spot, echo data are collected when the simulated smartphone points to the directions as marked by the red arrows at spot A in Fig. 3.28a. For each orientation, 100 echo traces with random perturbations are collected to the orientation such that the traces are slightly different. As a result, a total of 16 million echo traces are collected. The first four rows of Fig. 3.28b show the synthetic echoes' PSDs for four directions at spot A. They are slightly different from each other. The last row of Fig. 3.28b shows the echo's PSD at spot B. It is different from all PSDs synthesized at spot A. This shows that the used simulator can synthesize both the orientation- and location-dependent echoes. Note that, from the estimation, collecting the same amount of data in the real world requires about 400 hours of manual labor. This experiment using synthetic data focuses on evaluating the scalability of map superimposition algorithm. The pilot deployment of the system that crowdsources many users' data for map superimposition is left to future work.

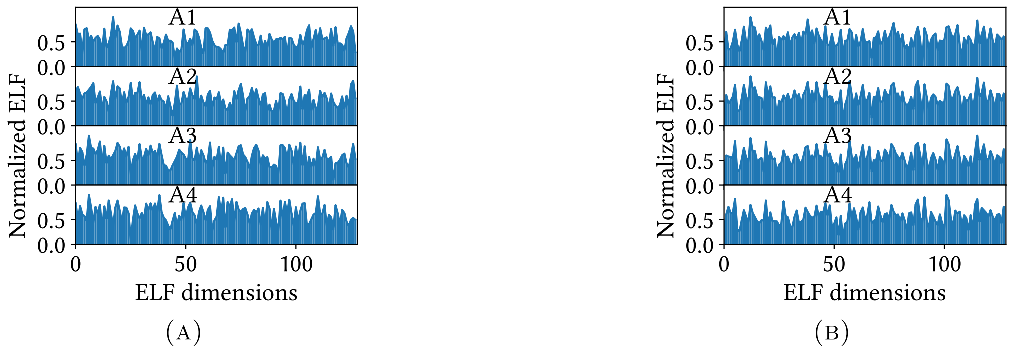
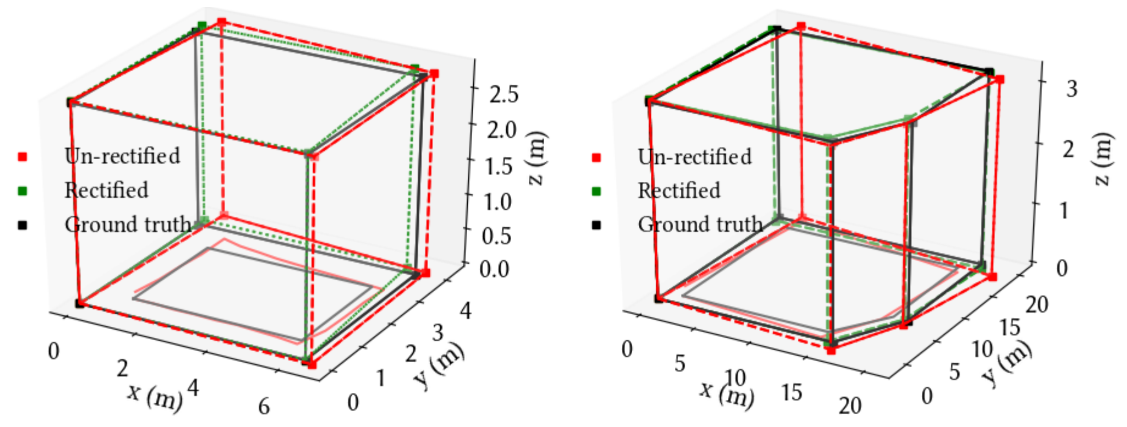


FIGURE 3.29: ELFs average similarity is (A) 0.34 before map superimposition and (B) 0.76 after map superimposition.

Fig. 3.29 shows the spot A's ELFs from directions 1 to 4 before/after map superimposition. The ELFs' average cosine similarity is 0.34 before the map superimposition. This value increases to 0.76 after applying the floor-level CL for map superimposition. This result shows that map superimposition is effective in reconciling the ELFs collected in different orientations. Fig. 3.27 shows the localization results on the synthetic data. The plot labeled "diff direction" shows the CDF when the CL-based map superimposition is not applied and the evaluated samples are in a different phone orientation from that in the used map. The mean localization error is 3.2 m. This poor result shows the necessity of the CL-based reconciliation. The curve labeled "superimposed" shows the results obtained using the floor map constructed by the floor-level CL. The mean localization error decreases to 0.24 m. The supervised fingerprint approach is also employed as a baseline, which forms the training dataset by labeling the echoes synthesized at the same spot with the same location label and trains a DNN to classify the 4,000 spots. The CDF curve labeled "supervised" shows the results. The mean localization error is 0.56 m. The supervised fingerprint approach is inferior to the proposed solution that performs localization using the floor map.

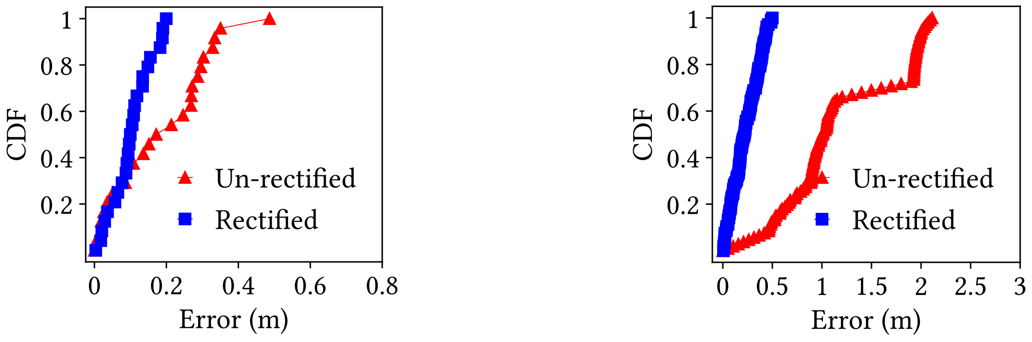
3.6.8 Room Geometry Reconstruction

Evaluation environments: Experiments are conducted in two polyhedron-shape rooms. The first one is a $4 \times 6.5 \times 2.8\,\mathrm{m}^3$ living room filled with furniture like TV and sofas. The second one is an $18 \times 20 \times 3.2\,\mathrm{m}^3$ relatively empty exhibition hall. In each room, a user holds the smartphone and walks along the sidewalls to collect the IMU and the echo data.



(A) Living room (B) Exhibition hall

FIGURE 3.30: Room construction results. The black curve on the xy plane represents the rectified trajectory, the red curve represents the unrectified trajectory.



(A) Living room (B) Exhibition hall

FIGURE 3.31: Room construction results.

Evaluation results: Fig. 3.30 shows the room reconstructions of both rooms. The polyhedron labeled ”Un-rectified” is the estimated room shape using the un-rectified IMU trajectory and the estimated wall distances. This plot represents the essence of [31, 32], where the performance of the mapping relies on the accuracy of the estimated IMU trajectory. The polyhedron labeled ”Rectified” is constructed using the rectified IMU trajectory via the ELF-SLAM and the estimated wall distances. The results show that the room geometry constructed upon the rectified user trajectories is closer to the ground truth compared to those constructed using the un-rectified trajectories. The distances between the constructed and the ground-truth walls are calculated to obtain the CDF of the room construction errors. Fig. 3.31 shows the results. The median construction errors for the ”Un-rectified” approach are about 0.32 m, and 1.2 m in the living room and the exhibition hall, respectively. The errors decrease to about 0.15 m and 0.35 m

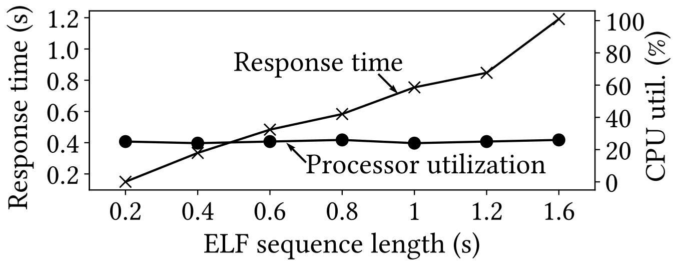


FIGURE 3.32: Model execution overhead.

for "Rectified" approach, representing a 2× and 4× error reduction. Thus, the room geometry reconstruction outperforms [31, 32] that rely on the un-rectified IMU results.

The system requires the user to walk a full loop along the walls of a room. The amount of data needed depends on the size of the room. Considering the average human walking speed of 1.2 m/s, the time needed for data collection is about 18 s and 64 s in the living room and the exhibition hall, respectively. Thus, the data collection for room geometry reconstruction incurs little overhead.

3.6.9 System Overhead

The computation overheads of the ELF extractor and real-time one-shot localization are evaluated using the floor map on the Google Pixel 4 smartphone. `Pytorch-Mobile` [113] is used to optimize and deploy the model. The compressed model is about 96 MB.

3.6.9.1 App's response time and processor utilization.

When the ELF sequence length varies from 0.2 s to 1.6 s, from Fig. 3.32, the smartphone processor utilization remains at around 20%. The memory usage of storing 4, 000 spots' ELFs is less than 4 MB. The app's response time is also measured, which includes the times for extracting ELFs and matching the ELF trace against the floor map. The response time increases from 0.18 s to 1.2 s, when the ELF sequence length varies from 0.2 s to 1.6 s. The increased response time is from the localization module, because the computation overhead of the feature matching increases with the ELF sequence length. From Fig. 3.21, by setting the ELF sequence length to be 0.6 s, the system achieves 0.1 m median localization error, while the corresponding measured response time is about 0.5 s. Thus, the user can get the localization result in about 1.1 s.

3.6.9.2 App's network bandwidth and battery usages.

To continuously transmit echo and IMU data to the cloud server for map construction, the app's bandwidth usage is around 90 kbps. This data rate is similar to that of Advanced Audio Coding (AAC), a widely adopted standard for lossy audio compression. Note that as the localization phase of ELF-SLAM is performed locally on the phone, it requires no data transmission. The `battery historian` [114] is used to estimate the app's energy usage. The app's energy usage per hour is around 270 mAh when the app performs localization continuously. This energy usage is similar to that of the Google Map app in continuous navigation, i.e., around 280 mAh and much lower than a visual SLAM [115], whose energy consumption is around 450 mAh. Thus, the ELF-based localization system introduces acceptable overhead.

3.7 Discussion

■ **Concurrent use and security.** It is common to have multiple users simultaneously use their smartphones for location sensing in the same indoor space. To avoid signal collision, carrier-sense multiple access (CSMA) protocols can be implemented in the ELF-SLAM system to manage the traffic. Specifically, when a smartphone is about to transmit the chirp, it uses its microphone to detect whether there is an ongoing echoing process. If so, the smartphone will defer the chirp play. The ELF-SLAM can be vulnerable to malicious attacks. For example, attackers can deploy speakers and play the sound in the used band to mislead the system. The development of the defense mechanism will be considered in future work.

■ **Domain adaptation across different devices.** The microphone and speaker hardware differences on smartphones could affect the performance of ELF-SLAM. Several machine learning techniques are available to address the domain shift problem, e.g., data augmentation [7], few-shot learning [37], and adversarial learning [116], etc. This thesis will explore an effective way to address device heterogeneity in future work.

■ **Impact of inaudible sound to human.** The experimental results in [117] show that participants experience mild side effect after 20 minutes' exposures in frequency from 12.5 Hz to 20 kHz with sound level from 44 to 71 dB. The played chirp level on Google Pixel 4 is investigated by increasing the sound volume from 20% to 100%, and the corresponding sound level is found to be between 40 - 68 dB. As shown in Fig. 3.19, ELF-SLAM requires less than 20 min data collection in the tested environment. Thus, ELF-SLAM will have little impact on humans within tens of minutes of exposure.

■ **Generalizing to other sensing modalities.** The proposed feature learning approach is general and can be applied to other sensing modalities. For example, the same approach can be applied to human activity recognition (HAR) tasks. By considering the characteristics of human movement, contrastive learning can be applied to learn useful features from massively unlabeled IMU sensor data. Then, a small amount of labeled data can be used to finetune the model to a specific application. The generalization to other sensing modalities will be considered in future work.

3.8 Summary

This chapter presents ELF-SLAM, an indoor smartphone SLAM system using acoustic echoes. ELF-SLAM uses a smartphone's audio hardware to emit near-inaudible chirps and record acoustic echoes in an indoor space, then uses the echoes to detect loop closures that regulate the IMU-based dead reckoning. To effectively capture loop closures, this thesis designs a trajectory-level contrastive learning procedure and applies it to the echoes to learn ELFs. Then, a clustering-based approach is designed to remove the false detection results and curate the loop closures. Third, the rectified trajectory map is applied to reconstruct the room's geometry. Lastly, the floor-level contrastive learning is designed to superimpose the trajectory maps. Extensive experiments show that ELF-SLAM achieves sub-meter accuracy in both mapping and localization, and outperforms both Wi-Fi RSSI and geomagnetic SLAMs. The room geometry reconstruction also outperforms the latest echo-based systems.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

This thesis identifies the *label scarcity* challenge in the development of machine learning algorithms for efficient AIoT sensing. The root causes of label scarcity arise from two aspects of AIoT sensing, i.e., difficulty in data annotation due to *the uninterpretable nature of IoT sensing data* and *run-time domain shift* caused by sensor characteristics or dynamically changing environment. This thesis exploits sensor and process characteristics to tackle label scarcity in AIoT sensing applications.

The first study, PhyAug proposes data augmentation guided by sensor and process characteristics to deal with domain shifts in AIoT sensing applications. The application scope of PhyAug includes both IoT sensing and multi-media sensing. The presented five case studies have distinct objectives and require deep models with different architectures and scales, which demonstrate the versatility of PhyAug. The extensive and comparative experiments show that PhyAug can recover significant portions of accuracy losses caused by sensors' characteristics and reduce target-domain training data sampling complexity.

The second study, ELF-SLAM exploits the kinetic relationship of acoustic data and designs customized contrastive learning procedures to learn a new echoic location feature. The proposed contrastive learning procedures use synthetic training data generated by the acoustic simulator for model pre-training. Then the pre-trained model is finetuned to the target indoor space using a few minutes of unlabeled acoustic data. The ELFs learned via contrastive learning are used in a newly developed indoor smartphone SLAM system for loop closure detection and user trajectory rectification. The designed SLAM system achieves sub-meter accuracy in both mapping and localization and outperforms Wi-Fi RSSI and geomagnetic SLAMs.

4.2 Future Work

This thesis targets AIoT applications where the generation of data is governed by certain first principles and exploits sensor and process characteristics to tackle the label scarcity challenge. The promising results of DNN performance and the reduction in data labeling encourage us to further exploit first principles to enhance machine learning algorithms in physics-rich AIoT sensing. The future research directions are summarized as follows.

Exploring new approaches for knowledge integration. Physics-informed machine learning leverages the known prior knowledge to accelerate model training or improve the performance of machine learning algorithms. PIML-based DNN models demonstrate superior performance in molecular properties prediction, fluid flow inference, edge plasma dynamics modeling, etc. A recent review [6] classifies the current PIML approaches into three categories based on their representation of physics. The method based on *observational biases* utilizes the data generated by the physical laws to guide the training of DNN models. The DNN models trained on such data can capture the mechanism governing data generation. The method based on the *inductive biases* focuses on designing specialized neural network architectures with physical laws embedded. The study [118] provides an example of this method, in which the cooling and heating units in a data center space are incorporated into the architecture of the neural network for temperature prediction. The method based on *learning biases* incorporates physical laws into the DNN model's loss function. This yields a more efficient model training due to additional constraints imposed by the physical laws. The study [12] is an example of this method, in which the free-fall law is incorporated into the loss function of an object detection neural network. The approaches proposed in this paper belong to the method based on *observational biases*. This method does not modify the existing DNN designs and learning algorithms, rendering it universally applicable. The methods based on *inductive biases* and *learning biases* require redesigning of neural network architecture and/or loss function. However, they bring the benefits of a more accurate model, reduced data requirement in training, improved model interpretability, and more robustness to noisy data, etc. Future research can explore the possibility of integrating known physics using methods based on the *inductive biases* and *learning biases* in AIoT sensing applications.

Handling intricate physical laws. This thesis uses the sensor and physical processes that can be parameterized to address the label scarcity in AIoT sensing. However, in many sensing applications, data generation can be governed by intricate physical laws. Developing mathematical models to accurately describe such physical laws can be difficult. Future research in handling intricate physical processes can opt to design neural works to model and simulate the underlying physical laws using the collected observational data. The neural networks can be treated as the surrogate model of the physical law and can be further used for data generation. For example, the recorded audio data in an indoor space is determined by the room impulse response (RIR) and can affect the performance of the deployed audio sensing model. The model adaptation can leverage the RIRs generated in the target room and forms the target-domain data for model training. However, the collection of RIRs requires specialized equipment and can involve significant manual effort. Multiple simulation platforms are established to generate RIRs by giving the parameters of a target room. As the actual RIR generation involves intricate physical processes, the existing simulators either make simplified assumptions about the room or require substantial predefined parameter tuning. The DNN models are promoted for complicated physical process modeling. Applied in this problem, a DNN model can be trained to generate RIRs for a target room. As such, the DNN model acts as a surrogate model of the physical laws governing RIRs generation. Using neural networks for physical laws modeling has the following advantages over conventional analytical approaches. First, neural networks are capable of handling complex data whereas analytical modeling suffers from the curse of dimensionality. Second, neural networks offer the flexibility of finetuning using the newly collected data samples. Mathematical models usually require handcrafted parameters and are less tunable.

Physics-informed parameter tuning for online model transfer. The domain shifts in many AIoT sensing applications are caused by gradually changed factors, such as the deterioration of the sensing devices and the changes in the ambient environment. Let Δ denote the measurement of the feature distance between the newly sensed data and the original source-domain data. Δ can be quantified based on the observed physical process governing the domain shifts. The measured Δ is featured in increasing over time. A larger Δ causes a more severe domain shift. The existing solutions in addressing domain shifts focus on training-based approaches. In future work, this thesis will explore the training-free approach based on the measured Δ for online model transfer. Specifically, a special neural network architecture, Hypernet is used to generate the parameters of a target neural network. The Δ is used as the model input of a Hypernet during the training. At the inference stage, by giving the Δ based on a small amount of sensed data, the trained Hypernet can generate the adjustment of parameters in order to adapt a base model to a target domain. Different from the training-based approach that can incur computational overhead for each target domain, using Hypernet for parameter generation is training-free and can adapt the model to a target domain on the fly.

List of Author's Awards and Publications[1]

Awards

- **Best Artifacts Award Runner-up**, *The 20th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'21).*

Conference Proceedings

- Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan and Guosheng Lin, "Indoor Smartphone SLAM with Learned Echoic Location Features", in *The 20th ACM Conference on Embedded Networked Sensor Systems (SenSys), Nov 6-9, 2022, Boston, USA*. DOI: 10.1145/3560905.3568510.

- Qun Song, Zhenyu Yan, Wenjie Luo and Rui Tan, "Sardino: Ultra-fast dynamic ensemble for secure visual sensing at mobile edge", in *The 19th International Conference on Embedded Wireless Systems and Networks (EWSN), October 3 - 5, Linz, Austria.*

- Dongfang Guo, Chaojie Gu, Linshan Jiang, Wenjie Luo, Rui Tan, "ILLOC: In-Hall Localization with Standard LoRaWAN Uplink Frames," *ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp), September 11 - 15, Atlanta, USA and Cambridge, UK. 6.1 (2022): 1-26.*

[1]The superscript * indicates joint first authors

- Wenjie Luo, Zhenyu Yan, Qun Song and Rui Tan, "Phyaug: Physics-directed data augmentation for deep sensing model transfer in cyber-physical systems", in *The 20th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), pages 31-46, May 18 - 21, 2021, Nashville, USA*. DOI: 10.1145/3412382.3458255.

Workshop Paper

- Wenjie Luo and Rui Tan, "Physics-informed Machine Learning Model Generalization in AIoT: Opportunites and Challenges," *The 3rd Workshop on Data-Driven and Intelligent Cyber-Physical Systems for Smart Cities (DI-CPS), 6 pages, San Antonio, May 9, 2023, Texas USA*. DOI: 10.1145/3576914.3588751

Demo

- Wenjie Luo, Dongfang Guo*, Chaojie Gu, Yuting Wu, Qun Song, Zhenyu Yan, Rui Tan, "Demo Abstract: Infrastructure-Free Smartphone Indoor Localization Using Room Acoustic Responses," *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys), November 15-17, 2021, Coimbra, Portugal.*

Journal Articles

- Wenjie Luo, Zhenyu Yan, Qun Song and Rui Tan, "Physics-directed data augmentation for deep model transfer to specific sensor," *ACM Transactions on Sensor Networks (TOSN), 19(1):1-30, 2022*. DOI: 10.1145/3549076.

Under Review

- Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan and Guosheng Lin, "Indoor Smartphone SLAM with Learned Echoic Location Features," *Under review by a journal.*

Bibliography

[1] 2020. `https://www.visualcapitalist.com/aiot-when-ai-meets-iot-technology/`. xiii, 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009. 2, 47

[3] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*. IEEE, 2015. 2

[4] `https://www.mturk.com/`. 2

[5] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 2019. 4

[6] George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, pages 422–440, 2021. 4, 100

[7] Akhil Mathur, Tianlin Zhang, Sourav Bhattacharya, Petar Velickovic, Leonid Joffe, Nicholas D Lane, Fahim Kawsar, and Pietro Lió. Using deep data augmentation training to address software and hardware heterogeneities in wearable and smartphone sensing devices. In *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 200–211. IEEE, 2018. 5, 6, 16, 17, 18, 19, 25, 34, 35, 40, 97

[8] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 2009. 6

[9] Dipanjan Sarkar, Raghav Bali, and Tamoghna Ghosh. *Hands-On Transfer Learning with Python*. Packt Publishing, 2018. ISBN 1788831306. 6

[10] Akhil Mathur, Anton Isopoussu, Fahim Kawsar, Nadia Berthouze, and Nicholas D Lane. Mic2mic: using cycle-consistent generative adversarial networks to overcome microphone variability in speech systems. In *Proceedings of the 18th international conference on information processing in sensor networks*, pages 169–180, 2019. 6, 7, 15, 16, 17, 18, 25, 26, 29, 34, 35, 36, 40

[11] Chengcheng Yu, Xiaobai Liu, and Song-Chun Zhu. Single-image 3d scene parsing using geometric commonsense. In *International Joint Conference on Artificial Intelligence*, 2017. 6

[12] Russell Stewart and Stefano Ermon. Label-free supervision of neural networks with physics and domain knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 6, 100

[13] Luning Sun, Han Gao, Shaowu Pan, and Jian-Xun Wang. Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. *Computer Methods in Applied Mechanics and Engineering*, 361:112732, 2020. 6

[14] Anthony Thyssen. Imagemagick v6 examples – distorting images, 2009. URL `https://legacy.imagemagick.org/Usage/distorts/`. 7, 46

[15] Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. *NIPS*, 30, 2017. 7, 16, 17, 18, 26, 35

[16] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017. 16, 17, 18

[17] Ali Akbari and Roozbeh Jafari. Transferring activity recognition models for new wearable sensors with deep generative domain adaptation. In *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, pages 85–96, 2019. 7, 16, 17, 18

[18] Sean Naren. deepspeech2.pytorch. `https://github.com/SeanNaren/deepspeech.pytorch`, 2020. 7, 15, 34

[19] Google play top charts, 2022. `https://play.google.com/store/apps/top`. 8

[20] Paramvir Bahl and Venkata N Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *Proceedings IEEE INFOCOM 2000. Conference on computer communications. Nineteenth annual joint conference of the IEEE computer and communications societies (Cat. No. 00CH37064)*, volume 2, pages 775–784. Ieee, 2000. 8

[21] Moustafa Youssef and Ashok Agrawala. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, pages 205–218, 2005. 8

[22] Jeffrey Hightower, Sunny Consolvo, Anthony LaMarca, Ian Smith, and Jeff Hughes. Learning and recognizing the places we go. In *International Conference on Ubiquitous Computing*, pages 159–176. Springer, 2005. 8

[23] Yin Chen, Dimitrios Lymberopoulos, Jie Liu, and Bodhi Priyantha. Fm-based indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pages 169–182, 2012. 8

[24] Chi Zhang and Xinyu Zhang. Litell: Robust indoor localization using unmodified light fixtures. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 230–242, 2016. 8

[25] Ruipeng Gao, Yang Tian, Fan Ye, Guojie Luo, Kaigui Bian, Yizhou Wang, Tao Wang, and Xiaoming Li. Sextant: Towards ubiquitous indoor localization service by photo-taking of the environment. *IEEE Transactions on Mobile Computing*, 15(2):460–474, 2015. 8

[26] Stephen P Tarzia, Peter A Dinda, Robert P Dick, and Gokhan Memik. Indoor localization without infrastructure using the acoustic background spectrum. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, pages 155–168, 2011. 8, 39, 41, 62, 63

[27] Suining He and Kang G Shin. Geomagnetism for smartphone-based indoor localization: Challenges, advances, and comparisons. *ACM Computing Surveys (CSUR)*, 50(6):1–37, 2017. 8

[28] Mirco Rossi, Julia Seiter, Oliver Amft, Seraina Buchmeier, and Gerhard Tröster. Roomsense: an indoor positioning system for smartphones using active sound probing. In *Proceedings of the 4th Augmented Human International Conference*, pages 89–95, 2013. 8, 9, 62, 63

[29] Yu-Chih Tung and Kang G Shin. Echotag: Accurate infrastructure-free indoor location tagging with smartphones. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015. 9, 62, 63, 79

[30] Qun Song, Chaojie Gu, and Rui Tan. Deep room recognition using inaudible echos. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018. 9, 39, 40, 62, 63, 65

[31] Swadhin Pradhan, Ghufran Baig, Wenguang Mao, Lili Qiu, Guohai Chen, and Bo Yang. Smartphone-based acoustic indoor space mapping. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018. 9, 62, 63, 95, 96

[32] Bing Zhou, Mohammed Elbadry, Ruipeng Gao, and Fan Ye. Batmapper: Acoustic sensing based indoor floor plan construction using smartphones. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, pages 42–55, 2017. 62, 63, 95, 96

[33] Jie Lian, Jiadong Lou, Li Chen, and Xu Yuan. Echospot: Spotting your locations via acoustic sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2021. 8, 9, 61, 62, 63

[34] Wenjie Luo, Zhenyu Yan, Qun Song, and Rui Tan. Phyaug: Physics-directed data augmentation for deep sensing model transfer in cyber-physical systems. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021)*, pages 31–46, 2021. 13

[35] Wenjie Luo, Zhenyu Yan, Qun Song, and Rui Tan. Physics-directed data augmentation for deep model transfer to specific sensor. *ACM Transactions on Sensor Networks*, 19(1):1–30, 2022. 13

[36] Matthias Pohl, Michael Schaeferling, Gundolf Kiefer, Plamen Petrow, Egmont Woitzel, and Frank Papenfuß. Leveraging polynomial approximation for non-linear image transformations in real time. *Computers & Electrical Engineering*, 40(4):1146–1157, 2014. 14

[37] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. Metasense: few-shot adaptation to untrained conditions in deep mobile sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 110–123, 2019. 16, 17, 18, 34, 97

[38] Rui Xiao, Jianwei Liu, Jinsong Han, and Kui Ren. Onefi: One-shot recognition for unseen gesture via cots wifi. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 206–219, 2021. 17, 18

[39] Rui Wang, Zuxuan Wu, Zejia Weng, Jingjing Chen, Guo-Jun Qi, and Yu-Gang Jiang. Cross-domain contrastive learning for unsupervised domain adaptation. *IEEE Transactions on Multimedia*, 2022. 16, 17, 26

[40] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015. 18

[41] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[42] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

[43] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 18

[44] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018. 21

[45] Yundong Zhang, Naveen Suda, Liangzhen Lai, and Vikas Chandra. Hello edge: Keyword spotting on microcontrollers. 2017. 22

[46] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008. 30, 67

[47] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015. 34

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019. 41

[49] Hyungtae Kim, Jaehoon Jung, and Joonki Paik. Fisheye lens camera based surveillance system for wide field of view monitoring. *Optik*, 127(14):5636–5646, 2016. 44

[50] Dieter Schmalstieg and Tobias Hollerer. *Augmented reality: principles and practice*. Addison-Wesley Professional, 2016. 44

[51] Jonathan Horgan, Ciarán Hughes, John McDonald, and Senthil Yogamani. Vision-based driver assistance systems: Survey, taxonomy and advances. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 2032–2039. IEEE, 2015. 44

[52] Zhen Chen and Anthimos Georgiadis. Parameterized synthetic image data set for fisheye lens. In *2018 5th International Conference on Information Science and Control Engineering (ICISCE)*, pages 370–374. IEEE, 2018. 45

[53] Senthil Yogamani, Ciarán Hughes, Jonathan Horgan, Ganesh Sistu, Padraig Varley, Derek O'Dea, Michal Uricár, Stefan Milz, Martin Simon, Karl Amende, et al. Woodscape: A multi-task, multi-camera fisheye dataset for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9308–9318, 2019. 45, 48

[54] Ciaran Hughes, Martin Glavin, Edward Jones, and Patrick Denny. Review of geometric distortion compensation in fish-eye cameras. 2008. 46

[55] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004. 47

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 47

[57] Hong Minhee. Magic wand, 2021. URL `https://pypi.org/project/Wand/`. 47

[58] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A toolbox for easily calibrating omnidirectional cameras. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006. 48

[59] opencv. Fisheye camera model, 2021. URL `https://docs.opencv.org/4.5.2/db/d58/group__calib3d__fisheye.html`. 48

[60] Matthew Faulkner, Michael Olson, Rishi Chandy, Jonathan Krause, K Mani Chandy, and Andreas Krause. The next big one: Detecting earthquakes and other rare events from community-based sensors. In *Proceedings of the 10th ACM/IEEE IPSN*, pages 13–24. IEEE, 2011. 52, 58

[61] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. Fidelity and yield in a volcano monitoring sensor network. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation (OSDI)*, pages 381–396, 2006. 52, 54

[62] Mostafa Mirshekari, Shijia Pan, Jonathon Fagert, Eve M Schooler, Pei Zhang, and Hae Young Noh. Occupant localization using footstep-induced structural vibration. *Mechanical Systems and Signal Processing*, pages 77–97, 2018. 52, 54

[63] Jose Clemente, WenZhan Song, Maria Valero, Fangyu Li, and Xiangyang Liy. Indoor person identification and fall detection through non-intrusive floor seismic sensing. In *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 417–424. IEEE, 2019. 52

[64] Jonathan M Lees and Robert S Crosson. Bayesian art versus conjugate gradientf methods in tomographic seismic imaging: An application at mount st. helens, washington. *Lecture Notes-Monograph Series*, pages 186–208, 1991. 52, 54

[65] Kevin Hupp, Brandon Schmandt, Eric Kiser, Steven M Hansen, and Alan Levander. A controlled source seismic attenuation study of the crust beneath mount st. helens with a dense array. In *American Geophysical Union (AGU) Fall Meeting Abstracts*, volume 2016, pages V33E–3170, 2016. 52

[66] Gregory Hackmann, Fei Sun, Nestor Castaneda, Chenyang Lu, and Shirley Dyke. A holistic approach to decentralized structural damage localization using wireless sensor networks. *Computer Communications*, 36(1):29–41, 2012. 52

[67] Michael J Bianco and Peter Gerstoft. Travel time tomography with adaptive dictionaries. *IEEE Transactions on Computational Imaging*, 4(4):499–511, 2018. 53

[68] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006. 55

[69] Clive D Rodgers. *Inverse methods for atmospheric sounding: theory and practice*, volume 2. World scientific, 2000. 55

[70] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011. 56

[71] Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan, and Guosheng Lin. Indoor smartphone slam with learned echoic location features. *The 20th ACM Conference on Embedded Networked Sensor Systems (SenSys), Nov 6-9, 2022, Boston, USA*, 2022. 61

[72] Wenjie Luo, Qun Song, Zhenyu Yan, Rui Tan, and Guosheng Lin. Indoor smartphone slam with learned echoic location features. *Under review*, 2023. 61

[73] Sheng Shen, Daguan Chen, Yu-Lin Wei, Zhijian Yang, and Romit Roy Choudhury. Voice localization using nearby wall reflections. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020. 61, 62, 63

[74] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 261–272, 2009. 62, 63

[75] Miranda Kreković, Ivan Dokmanić, and Martin Vetterli. Echoslam: floor with acoustic echoes. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11–15. Ieee, 2016. 63

[76] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th international conference on Embedded networked sensor systems*, 2007. 63

[77] Zengbin Zhang, David Chu, Xiaomeng Chen, and Thomas Moscibroda. Swordfight: Enabling a new class of phone-to-phone action games on commodity phones. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, 2012. 63

[78] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. Fingerio: Using active sonar for fine-grained finger tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016. 63

[79] Sangki Yun, Yi-Chao Chen, Huihuang Zheng, Lili Qiu, and Wenguang Mao. Strata: Fine-grained acoustic-based device-free tracking. In *Proceedings of the 15th annual international conference on mobile systems, applications, and services*, 2017. 63, 64

[80] Rajalakshmi Nandakumar, Alex Takakuwa, Tadayoshi Kohno, and Shyamnath Gollakota. Covertband: Activity information leakage using music. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2017. 63

[81] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. Soundwave: using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012. 63, 64

[82] Jagmohan Chauhan, Yining Hu, Suranga Seneviratne, Archan Misra, Aruna Seneviratne, and Youngki Lee. Breathprint: Breathing acoustics-based user authentication. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, 2017. 63

[83] Xiangyu Xu, Jiadi Yu, Yingying Chen, Yanmin Zhu, Linghe Kong, and Minglu Li. Breathlistener: Fine-grained breathing monitoring in driving environments utilizing acoustic signals. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, 2019. 63

[84] Xingzhe Song, Boyuan Yang, Ge Yang, Ruirong Chen, Erick Forno, Wei Chen, and Wei Gao. Spirosonic: monitoring human lung function via acoustic sensing on commodity smartphones. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020. 63

[85] Jan Willem Marck, Ali Mohamoud, Eric vd Houwen, and Rob van Heijster. Indoor radar slam a radar application for vision and gps denied environments. In *2013 European Radar Conference*, pages 471–474. IEEE, 2013. 64

[86] David Droeschel and Sven Behnke. Efficient continuous-time slam for 3d lidar-based online mapping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5000–5007. IEEE, 2018. 64

[87] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. Visual slam algorithms: A survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1):1–11, 2017. 64

[88] Brian Ferris, Dieter Fox, and Neil D Lawrence. Wifi-slam using gaussian process latent variable models. In *IJCAI*, pages 2480–2485, 2007. 64

[89] Aditya Arun, Roshan Ayyalasomayajula, William Hunter, and Dinesh Bharadia. P2slam: Bearing based wifi slam for indoor robots. *IEEE Robotics and Automation Letters*, 2022. 64

[90] Sen Wang, Hongkai Wen, Ronald Clark, and Niki Trigoni. Keyframe based large-scale indoor localisation using geomagnetic field and motion pattern. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 1910–1917. IEEE, 2016. 64, 86

[91] Chris Xiaoxuan Lu, Yang Li, Peijun Zhao, Changhao Chen, Linhai Xie, Hongkai Wen, Rui Tan, and Niki Trigoni. Simultaneous localization and mapping with power network electromagnetic field. In *Proceedings of the 24th annual international conference on mobile computing and networking*, pages 607–622, 2018. 64, 77, 85, 86

[92] Christine Evers and Patrick A Naylor. Acoustic slam. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1484–1498, 2018. 64

[93] Xiwu Zhang, Lei Wang, and Yan Su. Visual place recognition: A survey from deep learning perspective. *Pattern Recognition*, 113:107760, 2021. 64

[94] Daniele Cattaneo, Matteo Vaghi, and Abhinav Valada. Lcdnet: Deep loop closure detection and point cloud registration for lidar slam. *IEEE Transactions on Robotics*, 2022. 64

[95] Chao Cai, Rong Zheng, and Jun Luo. Ubiquitous acoustic sensing on commodity iot devices: A survey. *IEEE Communications Surveys & Tutorials*, 2022. 64

[96] The audible spectrum, 2022. `https://www.ncbi.nlm.nih.gov/books/NBK10924/`. 65

[97] Patrick Lazik and Anthony Rowe. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, pages 99–112, 2012. 65

[98] Human height, 2022. `https://ourworldindata.org/human-height`. 69

[99] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam: A factored solution to the floor problem. *AAAI*, 2002. 71

[100] Giorgio Grisetti, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2010. 72

[101] Cliff Randell, Chris Djiallis, and Henk Muller. Personal position measurement using dead reckoning. In *Seventh IEEE International Symposium on Wearable Computers, 2003. Proceedings*. IEEE Computer Society, 2003. 72

[102] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g 2 o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011. 72

[103] Sanjeev Arora, Wei Hu, and Pravesh K Kothari. An analysis of the t-sne algorithm for data visualization. In *Conference On Learning Theory*. PMLR, 2018. 74

[104] Robin Scheibler, Eric Bezzam, and Ivan Dokmanić. Pyroomacoustics: A python package for audio room simulation and array processing algorithms. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 351–355. IEEE, 2018. 75

[105] Jont B Allen and David A Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65 (4):943–950, 1979. 75

[106] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029*, 2020. 75

[107] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607. PMLR, 2020. 75

[108] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining*, 1996. 78

[109] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 78

[110] Ming Cui, John Femiani, Jiuxiang Hu, Peter Wonka, and Anshuman Razdan. Curve matching for open 2d curves. *Pattern Recognition Letters*, 30(1):1–10, 2009. 80

[111] Steven M Hernandez and Eyuphan Bulut. Performing wifi sensing with off-the-shelf smartphones. In *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 1–3. IEEE, 2020. 86

[112] Zheng Yang, Chenshu Wu, and Yunhao Liu. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 269–280, 2012. 86

[113] Pytorch mobile, 2022. `https://pytorch.org/mobile/home/`. 96

[114] Profile battery usage with batterystats and batter historian, 2022. `https://developer.android.com/topic/performance/power/setup-battery-historian`. 97

[115] Monocular visual odometry, 2022. `https://github.com/sunzuolei/mvo_android`. 97

[116] Akhil Mathur, Anton Isopoussu, Fahim Kawsar, Nadia Berthouze, and Nicholas D Lane. Mic2mic: using cycle-consistent generative adversarial networks to overcome microphone variability in speech systems. In *IPSN*, pages 169–180, 2019. 97

[117] A Van Wieringen. Undesirable effects as a result of short-term exposure to an ultrasonic repellent device. part ii—exposure of volunteers. *Assignment from the Federal Public Service in Belgium for Health, Food Chain Safety and Environment*, 2014. 98

[118] Ruihang Wang, Xin Zhou, Linsen Dong, Yonggang Wen, Rui Tan, Li Chen, Guan Wang, and Feng Zeng. Kalibre: Knowledge-based neural surrogate model calibration for data center digital twins. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 200–209, 2020. 100