REMEDIES FOR CHALLENGES IN DEEP LEARNING-BASED QUANTITATIVE ULTRASOUND IMAGING

BY

UFUK SOYLU

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2024

Urbana, Illinois

Doctoral Committee:

Professor Michael L. Oelze, Chair
Professor Minh N. Do
Assistant Professor Pengfei Song
Assistant Professor Yun-Sheng Chen
Assistant Professor Varun Chandrasekaran

ABSTRACT

The thesis of this dissertation suggests that there is a continuous demand for the development of data-efficient and robust deep learning (DL) algorithms in order to promote wider adoption of DL-based algorithms in clinical imaging. This dissertation proposes solutions specifically to address challenges in Deep Learning-based Quantitative Ultrasound (QUS).

QUS is a technique that uses ultrasound to estimate the properties of tissues, such as their scattering properties, elasticity or other relevant parameters, for the purposes of classifying tissue state. DL-based QUS aims to improve the accuracy and efficiency of this technique by using deep neural networks to analyze ultrasound image data and extract quantitative information. DL-based QUS approaches have the advantage of being model free and calibration-free if the data is acquired under a scenario of constant image settings from a single machine.

However, DL-based QUS faces several challenges, including the need for large amounts of data to train deep neural networks, the variability of ultrasound images due to differences in equipment and imaging settings, and the need for enhancing the understanding of security aspect of DL algorithms.

To address these challenges, this dissertation proposes the development of a data-efficient deep learning algorithm called *Zone Training*, which aims to learn diffraction patterns separately. Additionally, the study introduces a *Transfer Function* to mitigate variability between ultrasound images resulting from differences in equipment and imaging settings. Building on this, a strategy is developed to "steal" functionality from a victim machine and implement it on a perpetrator machine.

Overall, this study highlights the importance of developing data-efficient and robust DL algorithms for the wider adoption of DL-based quantitative ultrasound (QUS). It proposes several remedies to address the challenges faced by this emerging field. Additionally, it demonstrates the ease of stealing

the functionality of deep learning models, underscoring the need for security development of these models in clinical settings.

You are not a drop in the ocean
You are the entire ocean in a drop

ACKNOWLEDGMENTS

I am grateful to express my appreciation to Professor Michael Oelze, my Ph.D. advisor, for his valuable guidance and support throughout my academic journey and research endeavors. Professor Oelze has played a crucial role in directing my research and sparking new ideas, providing both mentorship and leadership to create a collaborative environment where novel ideas are nurtured. I am extremely grateful to my PhD committee members, whose contributions have been critical. I would like to acknowledge and thank all the members of Professor Oelze's lab for their unwavering support and assistance in my studies. Finally, I want to express my gratitude to my parents, Yasemin and İrfan, and my brother Utku.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

DL powered biomedical ultrasound imaging is becoming more advanced and coming closer to routine clinical applications in recent years [1]. DL is the process of learning a hierarchy of parameterized nonlinear transformations to perform a desired function. Therefore, DL algorithms extract a hierarchy of features from raw input images and image data automatically, rather than extracting features manually. Due to rapid increase in computational power and large data-sets, DL and machine learning algorithms have emerged as leading tools and have achieved impressive results in various research fields.

Some of the fields that DL has been researched and applied to include healthcare, self-driving cars, fraud detection, natural language processing, speech processing, and visual recognition [2–7]. Medical imaging is one of the fields in which DL has demonstrated success in various tasks such as medical image reconstruction, medical image enhancement, segmentation, image registration, detection [8–17]. Common DL applications that have provided notable results in the context of biomedical ultrasound imaging are classification [18–23], detection [18, 24], segmentation [18, 25–28], image reconstruction [29–37] and ultrasound elastography [38–42]. Furthermore, DL algorithms have been employed in advanced ultrasound imaging applications such as super-resolution imaging of microvasculature structure via Ultrasound Localization Microscopy [43–46]. Among DL algorithms, convolutional neural networks (CNN) use convolutional layers to embed structural priors of translational invariance, which make them parameter and data efficient learners for image analysis tasks. Respectively, CNNs are the most popular and successful DL structure for ultrasound biomedical imaging [18].

One application of DL in diagnostic ultrasound is the classification of tissue state from raw radiofrequency (RF) ultrasound backscatter. Classifying tissues has recently evolved from model-based approaches, such as QUS techniques, to model-free, DL-based techniques. Nguyen et al. [47] demonstrated

that QUS techniques are able to detect the presence of steatosis in a rabbit model of fatty liver with a classification accuracy of 84.11%. In a later study, Nguyen et al. [21] compared a DL-based classifier to a QUS-based classifier for the problem of fatty liver classifier and found that the DL-based classifier outperformed the QUS-based approach with the accuracy of 74% versus 59%. This study served as the baseline for this thesis by demonstrating the potential of DL-based approaches over QUS-based approaches. This led to a focus on DL-based approaches and the challenges that must be overcome for the clinical adoption of such techniques. While the traditional spectral-based QUS approach does not utilize the phase information in the RF signal, DL-based approaches can extract additional classification power from the lost phase information from the RF data. Furthermore, the DL-based approach does not require a model like the QUS approach, which means that features of the backscattered signal that are missed by the QUS approach can be picked up by the DL approach. Subsequently, the DL approach performs feature extraction and classification simultaneously. Similar approaches were later used to quantify liver disease in human patients and to characterize breast masses for cancer detection [22, 23, 48].

Even though DL is promising for biomedical imaging tasks such as classification of tissues based on their RF backscattered data, there are certain roadblocks to wider clinical adoption [49]. A major roadblock to deploying DL-powered algorithms to real clinical settings is data scarcity. Specifically, there is a scarcity of labeled data, largely due to the high costs of conducting lab experiments or acquiring expert annotations. Another roadblock is that there are large variations in ultrasound images due to operator, patient, or machine-dependent factors, which causes data mismatches between training data and testing data. Data mismatch happens due to mismatches between development and deployment environments and tends to limit the generalizability of DL-based algorithms. Overall, in the generic case, when there is not enough labeled data or when no assumptions can be made about the mismatches between development and deployment settings, any learning-based algorithm would be ineffective. Therefore, to turn DL-powered QUS into a reality, there is a constant need for developing DL algorithms that are data-efficient and more robust against data mismatches in ultrasound images.

Data efficiency is well studied in the DL literature [50]. Transfer learning (TL) is a key approach to overcoming the challenges related to limited

training data [51–57]. In TL, generalized knowledge can be shared between a pre-trained model and a new DL model if they perform similar tasks. This has the potential to reduce the amount of labeled data needed to train a new model. Similarly, self-supervised learning (SSL) is a successful technique in low-data regimes [58–60]. In SSL, a pretext task is developed to utilize unlabeled data, and then pre-task training is employed for downstream tasks with limited training data. Another approach is to use generative models to yield data with similar features as the training data. Generative adversarial networks [61, 62], variational autoencoders [63] are some examples from the literature. Additionally, model complexity is a relevant concept when dealing with low training data. Reducing model complexity, such as by reducing the number of layers or nodes and using simpler activation functions, can aid in generalization. This trade-off involves exchanging the capacity to represent complex relationships for potentially lower performance [64–68]. Similarly, the loss function is a critical component of DL training. Evaluating loss functions for data scarcity problems may lead to the selection of more optimal loss function designs in terms of handling data scarcity [69–72, 72, 73]. Physics-Informed Neural Networks (PINNs) are another approach to dealing with low training data. PINNs leverage underlying physics laws to formulate equations that can be incorporated into the design of the loss function [74–77]. Following general trends, the ultrasound imaging community has also quickly adopted these concepts in deep learning-based approaches. Transfer learning has been investigated for various ultrasound tasks, such as kidney segmentation, breast cancer image classification, lung segmentation, and liver steatosis [78–82]. Similarly, SSL has been investigated for various ultrasound tasks such as liver classification, 3D imaging, representation learning, cancer detection [83–87]. Additionally, generative models have been utilized to generate realistic images in the context of ultrasound imaging [88–90]. PINNs were utilized for investigating ultrasound wave propagation [91–93].

There is also a wide and rich literature anthology related to the data mismatch problem in DL [94, 95]. For example, data augmentation is a crucial tool for minimizing data mismatch. Some approaches build heuristic data augmentations to approximate the distribution shift between testing and training data, aiming to improve robustness [96–99]. The performance of these approaches depends on how well the approximation mitigates the distribution shift. Other approaches attempt to learn data augmentation by train-

ing a generative model between testing and training domains [100–103]. On the other hand, domain generalization approaches aim to recover feature representations that are independent of domains [104–106]. Their performance relies on the invariance of the learned features. Additionally, BN-Adapt [107] modifies batch normalization layers adaptively using test domain data. Moreover, pretraining is another significant concept [108–110]. Pretraining on a larger dataset could provide robust representations for downstream tasks. The issue of data mismatch has gained increased attention in recent literature focusing on DL-based QUS [103,111–116]. In the context of QUS, these concepts were investigated. Adaptive batch normalization was utilized [111]. Data augmentation with a meta learning algorithm was utilized to generate consistent attenuation coefficient images [116]. In a slightly different line of work, the issue of low sample size for estimating QUS parameters, specifically HK parametric images, was addressed using DL-based solutions [117]. Additionally, cycle-consistent generative adversarial networks were applied to address the issue of data mismatches in ultrasound imaging [103]. Furthermore, the Fourier Domain Adaptation technique was employed, proposing the replacement of lower frequency components within the frequency spectrum [115].

Another relevant literature pertains to test-time adaptation algorithms. One aspect of this thesis focuses on model security. It develops a model extraction attack using the tools developed to address the data mismatch issue and the most relevant framework is black-box unsupervised domain adaptation (UDA). A detailed survey on test-time adaptation algorithms can be found in [118]. Black-box UDA falls under the umbrella of test-time adaptation. There exist multiple strategies to address the black-box UDA problem: pseudolabeling aims to assign labels for unlabeled data via the black-box model. Consistency regularization aims to enforce consistent network predictions by adding a loss term in the training. Clustering assumes that the decision boundary in the unlabeled data should lie in low-density regions. Self-supervised learning aims to learn feature representation from unlabeled data based on auxiliary prediction task to be used in the downstreaming task. Iterative Learning with Noisy Labels (IterLNL) [119] is one of the state-of-the-art black-box UDA methods which conducts noisy labeling and learning with noisy labels iteratively. IterLNL is used to develop a model extraction strategy in this thesis. From a security perspective, developing

defense mechanisms is critical to ensure the wider clinical adoption of DL-based approaches. However, there is currently a lack of effective defense mechanisms against such model extraction attacks [120]. One promising approach as a defense mechanism is watermarking [121–123]. The concept of watermarks involves intentionally causing the watermarked model to overfit to outlier input-output pairs that are known only to the defender. This overfitting can then be used as a method to claim ownership of the model.

In this study, as a data-efficient DL algorithm, we propose a training strategy, which we call *Zone Training* to improve classification [124]. We consider the diffraction patterns associated with ultrasonic transducers and how they result in different regions or 'zones' that must also be learned to separate the system signal from the sample signal. In *Zone Training*, we propose to divide the complete field of view of an ultrasound image into multiple zones such as pre-focal, on focus and post focal zones. Then, we train separate neural networks for each zone by using the data belonging to the corresponding zone. In a sense, we train expert neural networks for each zone as opposed to *Regular Training*, which uses all data coming from the complete field of view to train a single neural network. The main intuition is that at each zone, there are different diffraction patterns and learning all the patterns by a single network is harder than learning a single diffraction pattern by a single expert network. The main advantage of *Zone Training* is that it requires less data to achieve similar classification performance in comparison to *Regular Training* in low data regime. Furthermore, the study of *Zone Training* revealed a previously unidentified type of data mismatch which we called zonal mismatches. The study shows that training DL-based algorithms with data acquired from a certain zone only works well for that specific zone. This could be problematic in scenarios where the training data were acquired from a certain zone, such as the on-focus or post-focal zone, and then the testing data occurs at a different zone, such as the pre-focal zone. This situation could arise in a scenario like liver disease assessment. Scanning can be done at the on-focus or post-focal zone for training, but if a patient is then scanned at the pre-focal zone, it can result in a data mismatch.

To increase the robustness of DL algorithms against data mismatches in ultrasound images, we propose the use of *Transfer Functions* [114, 125]. As mentioned earlier, there will inevitably be data mismatches between training and testing data distributions as the clinical environment is too complicated

to be fully realized in a development setting. We look at the problem from a signals and systems perspective and propose using calibration phantoms to obtain *Transfer Functions* capable of mitigating data mismatches in an economical way. We foresee two main applications in terms of model development: i) In System Model Transferability: In clinical practice, an operator of an ultrasound scanner will adjust scanner settings to acquire the best perceived image. In order to allow this flexibility with DL-based QUS, a large amount of data at each setting is needed to create a good model. With our approach a large amount of data is required at only one setting and a small amount of calibration data acquired at other scanner settings allowing the model to be transferred from the training setting to all other scanner settings. ii) Out of System Model Transferability: Our approach will also improve the machine learning model transferability between different systems, such as different transducers and different imaging machines, which leads to reduced cost of model development.

Expanding on the concept of *Transfer Functions*, we have developed a strategy to 'steal' the functionality of a deep learning (DL) model from one ultrasound machine and implement it on another. This demonstrates the ease with which the functionality of a DL model can be transferred between machines, highlighting the security risks associated with deploying such models in commercial scanners for clinical use. The proposed strategy is a black-box unsupervised domain adaptation technique that integrates the transfer function approach with an iterative schema. It does not require any information related to the internal workings of the victim machine's model, relying solely on the availability of the input-output interface and unlabelled data from the testing machine, i.e., the perpetrator machine. This scenario could become commonplace as companies deploy their DL functionalities for clinical use. Competing companies might acquire a victim machine and, through the input-output interface, replicate the functionality onto their own machines. Therefore, establishing security measures prior to deploying DL models in clinical settings is essential.

CHAPTER 2

A DATA-EFFICIENT DEEP LEARNING STRATEGY FOR QUANTITATIVE ULTRASOUND: ZONE TRAINING

2.1 Motivation

Throughout out the study, we examine DL techniques for classifying samples based on ultrasonic backscattered RF data. To improve classification, we consider the diffraction patterns associated with ultrasonic transducers and propose *Zone Training*, depicted in Fig. 2.1. In *Zone Training*, we divide the field of view of an ultrasound image into pre-focal, on focus and post focal zones. Then, we train expert networks for each zone. The main intuition is that learning different diffraction patterns using a single network is harder than learning a single diffraction pattern. The main advantage of *Zone Training* is that it requires less data to achieve similar classification performance in comparison to *Regular Training* in low data regime. *Zone Training* is similar to applying an attention mask to the input manually and training separate networks for each mask to learn dedicated convolutional filters per zone. In this sense, *Zone Training* applies attention in a simple and direct way to incorporate the physics of diffraction into DL training. There are methods in the literature that enable learning of varying convolution kernels over the complete field of view, e.g., pixel-adaptive convolution [126]. In general, attention mechanisms, e.g. self-attention mechanisms, [127] were invented initially for computer vision tasks, where the data is abundant and they apply attention by altering network architecture and hence model complexity to improve classification accuracy. However, in the context of biomedical imaging, we are in a different regime where the data is often scarce. Therefore, we favor utilizing a smaller training set to achieve a desired classification accuracy. Overall, *Zone Training* provides us a method to reduce training set size by modifying data distribution without altering model complexity. Furthermore, *Zone Training* can be perceived as utilizing a symbolistic approach, in

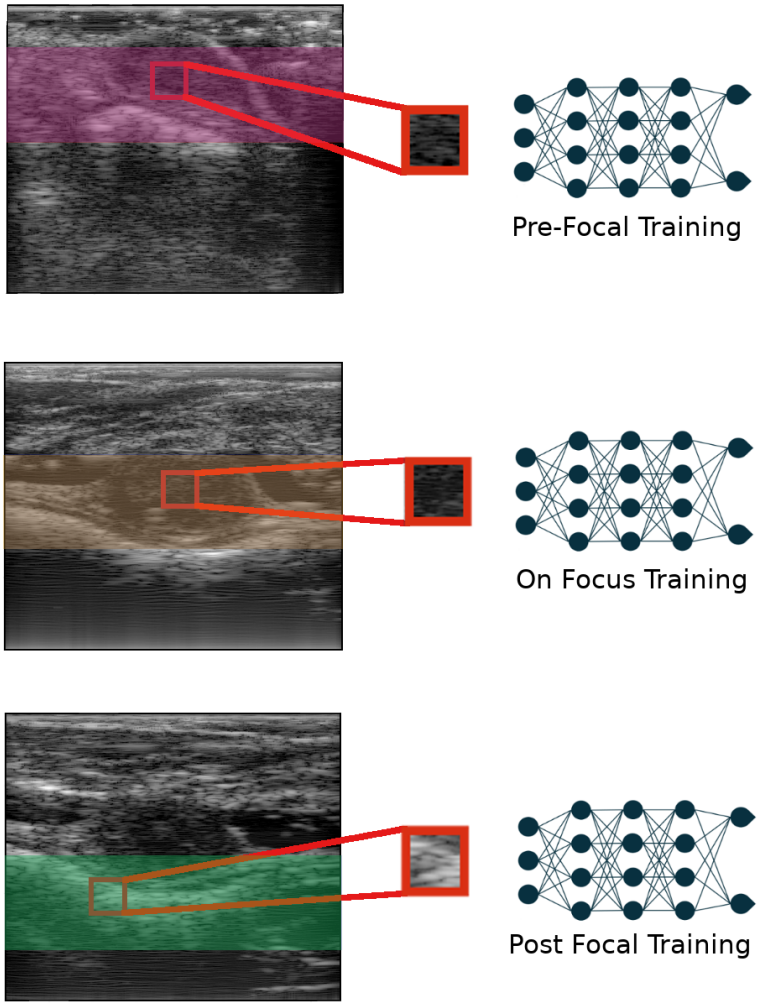


Figure 2.1: Zone Training

the form of a simple if-else structure (if data is from a certain zone, train a specific network), to transfer physics knowledge into DL training. Combining DL and symbolic reasoning is known in the literature as Neural-Symbolic Computing (NSC), which can lead to data-efficient AI [128].

Tissue classification was the primary application throughout the study. In this chapter, we tested the proposed method to classify three distinct tissue-mimicking phantoms. To further motivate *Zone Training*, we describe a clinical scenario when it is the most relevant and advantageous. For instance, ultrasound imaging can be used to examine and characterize tumors, whether benign or malignant, which can exist at different depths within a body. When

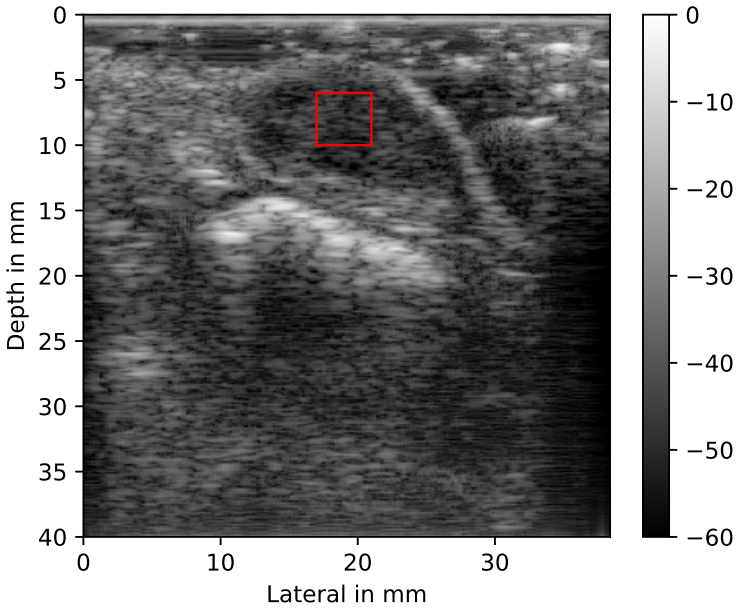
using QUS approaches for tumor characterization, a region of interest (ROI) inside the tumor is selected to examine the signals from the tumor. We show two tumor image examples where the tumors are at different depths in Fig. 2.2, but the same probe is used. Different depths correspond to different zones and red rectangles are sampled from the tumors in those ultrasound images. In *Zone Training*, we have separately trained DL algorithms for each zone. Experimental studies for *Zone Training* in this chapter were conducted under two assumptions following the clinical scenario. Firstly, we assume that we are not trying to detect the ROI. In other words, we are given rectangular patches of data to classify a tissue state. The ROI can be detected by another algorithm or by the operator. The operator in the clinic can adjust imaging settings to obtain the best imaging quality, and then, select the ROI, which should be considered as Human-centered AI, whose aim is to amplify and augment rather than displace human abilities [129]. The second assumption is that the ROI is larger than the rectangular patches of data so that the classification networks, take uniform rectangular patches as their input. This scenario and its two assumptions are valid for the rest of the chapters as well.

2.2 Methods

2.2.1 Phantoms

Three different tissue-mimicking phantoms were used in this chapter, which we designated as Phantom1, Phantom2 and Phantom3. They are cylindrically shaped as shown in Fig. 2.3 and their properties are summarized in Table 2.1.

Phantom1, which mimics human liver, has been described by Wear et al. [130]. Phantom1 had a measured attenuation coefficient slope of 0.4 $\mathrm{dB}\times\mathrm{cm}^{-1}\times\mathrm{MHz}^{-1}$. Its materials were produced based on the method of Madsen et al. [131] and they are macroscopically uniform. The only nonuniformity in Phantom1 results from the random positioning of microscopic glass bead scatterers. The component materials and their relative amounts by weight for Phantom1 are agarose (3.5%), n-propanol (3.4%), 75 to 90 μm-diameter glass beads (0.38%), bovine milk concentrated 3 times by reverse



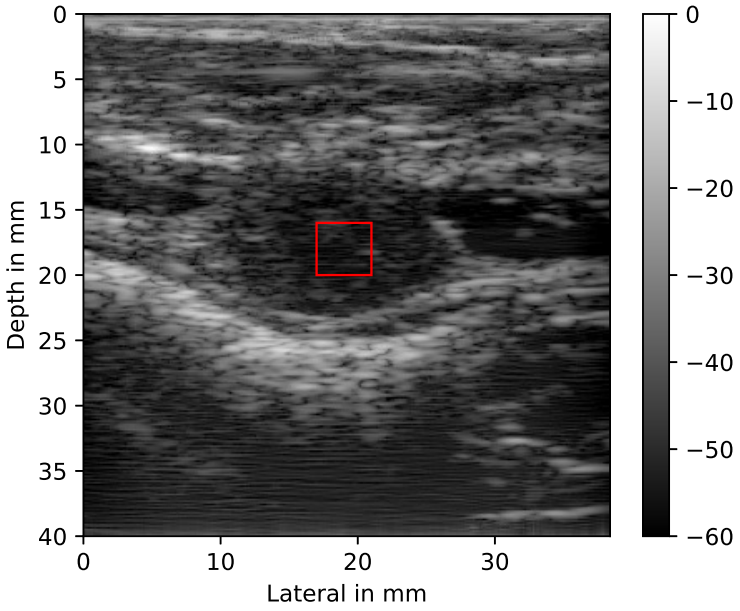


Figure 2.2: Tumor examples, in dB scale, whose scan depths vary in the field of a transducer probe. The tumor images were acquired from rabbits having mammary VX2 tumors. All animal experiments were approved by the Institutional Animal Care and Use Committee at the University of Illinois at Urbana–Champaign.

osmosis (24.5%), liquid Germall Plus preservative (International Specialty Products, Wayne, NJ) (1.88%), and 18-$M\Omega$-cm deionized water (66.3%).

Phantom2 and Phantom3 are both low attenuation phantoms, whose properties have been described by Anderson et al. [132] and constructions have been described Madsen et al. [133]. Both phantoms were made with the same weakly-scattering agar background material but contained different sizes of scatterers. They have an attenuation coefficient slope of equal to 0.1 dB$\times$cm$^{-1}\times$MHz $^{-1}$. Glass-sphere scatterers (Potters Industries, Inc., Valley Forge, PA; Thermo Fisher Scientific (formerly Duke Scientific), Inc., Waltham, MA) were used in both phantoms with weakly scattering 2% agar background. The only difference in the phantoms was the size distribution of the glass bead scatterers, i.e., Phantom2 had glass beads with a mean diameter of 41 μm and Phantom3 had glass beads with a mean diameter of 50 μm.

Phantom1 Phantom2 Phantom3

Figure 2.3: Tissue-mimicking Phantoms

Table 2.1: Phantom Properties

	Phantom1	Phantom2	Phantom3
Bead diameter (μm)	82.5 $\pm$ 7.5	41 $\pm$ 2	50 $\pm$ 2.4
Background	3.5% agar	2% agar	2% agar
Sound speed (m/s)	1540	1539	1539
Attenuation (dB/cm/MHz)	0.4	0.1	0.1

2.2.2 Ultrasound Scanning Procedures

Ultrasound gel was placed on the surfaces of the phantoms, and then the phantoms were scanned with an L9-4/38 transducer using a SonixOne sys-

tem (Analogical Corporation, Boston, MA, USA) providing an analysis bandwidth of 2-7.5 MHz. 1,007 frames of post-beamformed RF data sampled at 40 MHz were acquired from each phantom and saved for offline processing.

The imaging array had a center frequency of approximately 5.5 MHz and was operated with a single axial focus at 2 cm depth and a fixed elevational focus of 1.9 cm. The center frequency of the pulse was chosen as 9 MHz to provide higher bandwidth (resolution) for the transducer. The total imaging depth was chosen as 4 cm, which is equal to the height of the phantoms. Output power was chosen as -5 dB, which corresponds to -5 dB lower power level with respect to maximum output power of the system.

2.2.3 Data-set

We acquired 1,007 ultrasound images per phantom by free-hand motion. In total, we acquired 3,021 ultrasound frames. The size of an ultrasound image frame was 2,080 pixels×256 pixels. There were 2,080 samples along the axial direction that corresponded to 4 cm depth. Even though the L9-4/38 transducer has 128 channels, the SonixOne system interpolates to 256 channels that correspond to 256 lateral pixels. The data-set of ultrasound images is also publicly available at https://osf.io/7ztg3/ (DOI 10.17605/OSF.IO/7ZTG 3). After acquiring the ultrasound images per phantom, we extracted rectangular image patches to be used in training and testing.

In patch extraction, which is depicted in Fig. 2.4, we extracted rectangular image patches whose sizes were 200 pixels×26 pixels that correspond to square image patches whose sizes were 4 mm×4 mm in physical dimensions. From one ultrasound image, we could extract 81 (9 lateral × 9 axial) image patches when we used the complete field of view as in *Regular Training*. While extracting image patches, we did not use the first 540 pixels in the ultrasound image. Axially, we obtained the next line of individual patches by translating the start of the next patch by 100 pixels along the axial depth. Laterally, we obtained the next line of individual patches by translating the start of the next patch by 26 pixels along the axial depth. Overall, in patch extraction for *Regular Training*, there were 9 axial lines and 9 lateral lines to extract individual patches that led to extracting 81 image patches per ultrasound image.

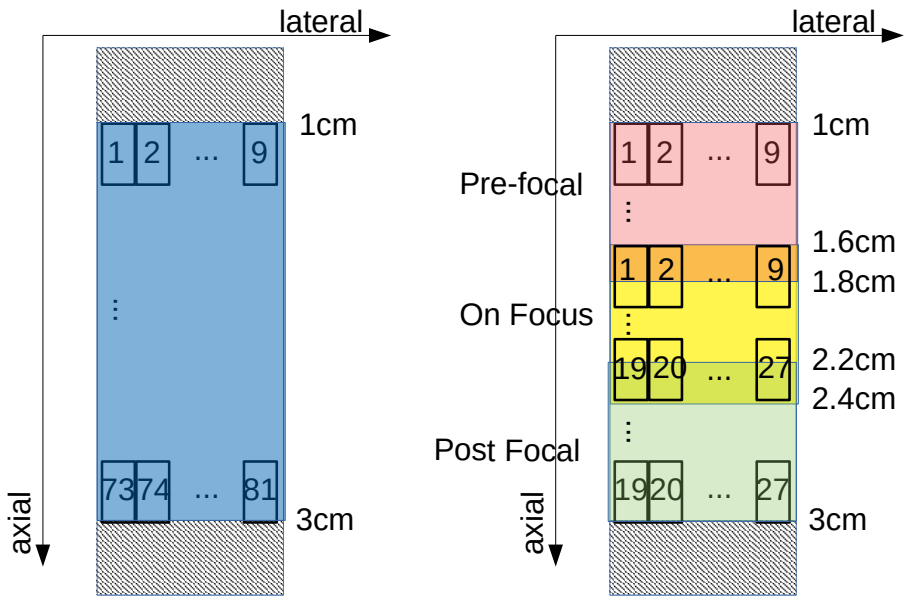


Figure 2.4: Patch Extraction

For *Zone Training*, we first developed definitions for the zones based on the diffraction pattern for a single focused transducer. In this chapter, we broke the complete field of view into three zones axially: a pre-focal zone which is centered at 1.4 cm, an on focus zone which is centered at 2 cm, a post focal zone which is centered at 2.6 cm. Then, each zone coincides with 3 axial lines of the complete field of view in patch extraction. Therefore, three zones together use the same data as in *Regular Training*. For each zone, we extracted 27 (9 lateral $\times$ 3 axial) image patches whose sizes were 200 pixels$\times$26 pixels corresponding to 4 mm$\times$4 mm in physical dimensions. Example B-mode images of the patches corresponding to each phantom are provided in Fig.2.5.

2.2.4 Training

DL training was done by using two machines each with a single GPU. One machine had TITAN RTX and the other machine had RTX A5000. All implementations were done with the PyTorch library [134]. As a data pre-processing step, we applied z-score normalization at the patch level, i.e., the mean intensity value of patches was subtracted from each patch, and then,

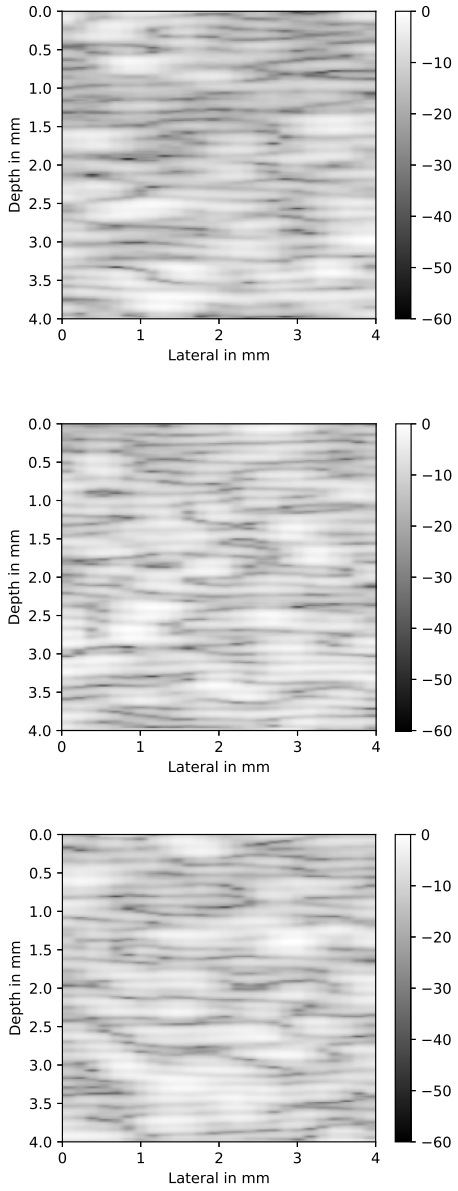


Figure 2.5: Example B-mode images of extracted patches in dB scale from the pre-focal zone centered at 1.4 cm. Classifying these patches are difficult by visual inspection. Top Row: Phantom1; Middle Row: Phantom2 and Bottom Row: Phantom3.

each pixel in a patch was divided by the standard deviation of the intensity of the patches. The batch number was chosen as 128 through out all experiments. Horizontal flip with 0.5 probability was implemented as a data augmentation step in the training process. We used the Adam algorithm [135] as the optimizer in all experiments. Additionally, the models were trained by

using cross entropy loss with uniform class weights, which includes built-in softmax function in PyTorch implementation [134].

In this chapter, we used CNN architectures consisting of two parts: feature extractors that consist of convolution layers, max-pooling layers and non-linear activation functions, and a classifier that consists of fully connected layers and non-linear activation functions. They also have significantly fewer parameters and so they can be trained more efficiently than fully connected networks [136]. We used a slightly modified CNN architecture, which is derived from AlexNet [137] and is shown in Table 2.2. In the training, dropout layers with 0.5 probabilities were added to improve the regularization and deal with over-fitting, before fully connected1 and fully connected2 layers. Initial weights for the network were chosen based on the original paper [137].

Table 2.2: Network Architecture

Layer Name	Output Size	Regular & Zone Training
conv1&relu	$48 \times 4 \times 96$	$11 \times 11, \text{stride4}$
conv2&relu	$48 \times 4 \times 256$	$5 \times 5, \text{pad2}$
conv3&relu	$48 \times 4 \times 384$	$3 \times 3, \text{pad1}$
conv4&relu	$48 \times 4 \times 384$	$3 \times 3, \text{pad1}$
conv5&relu	$48 \times 4 \times 256$	$3 \times 3, \text{pad1}$
maxpool1	$23 \times 1 \times 256$	$3 \times 3, \text{stride2}$
fc1&relu	4096	5888×4096 connections
fc2&relu	4096	4096×4096 connections
fc3	3	4096×3 connections

In the experiments, we searched the learning rate and the epoch number by using a validation set. More specifically, the learning rate and the epoch number were determined to achieve "asymptotic test accuracy", which ideally is defined as the number of epochs of training required such that any further training provides no improvement in test accuracy. The process of forming training, testing and validation sets started with randomly selecting the desired number of ultrasound images per phantom. The same number of ultrasound frames were set apart for validation, training and testing sets. Then, we extracted patches, as described in Section 2.2.3, to form the training, testing and validation sets. After adjusting the learning rate and epoch number by using the validation set, we trained neural networks in the training sets and obtained classification accuracies in the test sets. We repeated each experiment 10 times, starting from random ultrasound frame selection

for training and testing sets. In Section 2.3, we report the learning rate, epoch number, mean classification accuracy, and standard deviation for each experiment.

2.2.5 Depth-Aware Training

In patch extraction, global coordinates are lost. Therefore, in addition to *Regular Training*, *Zone Training* is also compared against *Depth Aware Training*, which utilizes global coordinates in the training. In *Depth Aware Training*, we input the depth as an additional feature. Specifically, the CNN now takes a two-layered input, one layer is the image patch of 200 pixels $\times$26 pixels and the other one is a constant array of 200 pixels$\times$26 pixels whose values correspond to the relative depth, as shown simplistically in Fig. 2.6. The depth information is normalized between 0-1 where 0 is the depth of the nearest patch and 1 is the depth of the farthest patch. Overall, *Depth Aware Training* is designed to consider the global location of the input patch during both training and testing so that the DL network adapts itself based on the relative depth being near 0 or 1.

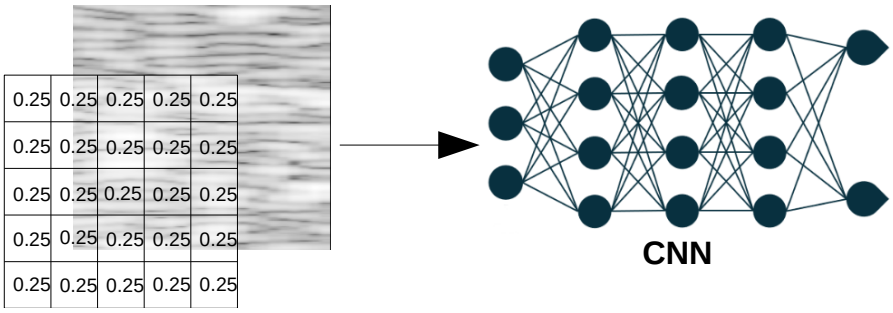


Figure 2.6: *Depth Aware Training*: Two-Layered Input, one with depth information and the other one with ultrasound RF data

2.3 Results

Results are organized into two parts. In the first part, we present results that allowed us to determine if our zone definitions were favorable by experimenting with axial zone widths, axial zone locations and by sweeping testing zone centers around training zone centers. Our purpose in this part was to

determine a reasonable way to divide the field of view into multiple zones, which is required for *Zone Training*. In the second part, we investigated the relationship between training set size and classification accuracy for *Zone Training*, *Regular Training*, and *Depth Aware Training*.

2.3.1 Examination of Zone Definitions

We now present four results that are helpful in determining zone definitions. In the first result, we investigated how much classification accuracy drops as we shifted the testing zone away from the training zone. Specifically, we trained a neural network by using patches from the on focus zone, and then, we tested the neural network with patches from nearby zones. This result revealed how much the diffraction patterns changed around the focal zone. In the second result, we repeated the same experiment for the pre-focal zone and the post focal zone to investigate how much the diffraction patterns changed around these zones. In the third result, we experiment with axial zone width in terms of number of overlapping patches per zone. We plotted classification accuracy for the on focus zone when we increase the number of overlapping patches used in patch extraction. In the fourth result, we experimented with axial zone locations, and we plotted classification accuracy at different zone centers.

In Fig. 2.7, classification accuracy was plotted as the testing zone center was swept by 0.8 cm towards and away from the transducer around the training zone center. We trained a neural network by using patches from the on focus zone centered at 2 cm depth, and then we tested the neural network with patches from zones centered at 1.2 cm, 1.4 cm, 1.6 cm, 1.8 cm, 2 cm, 2.2 cm, 2.4 cm, 2.6 cm and 2.8 cm, respectively. Overall, the y axis represents classification accuracy and the x axis represents the relative distance between the testing zone and the training zone. For instance, a value of -0.8 means that the testing zone was 0.8 cm closer to transducer than the training zone and +0.8 means that the testing zone was 0.8 cm farther away from transducer than the training zone. We repeated the experiments for different sizes of training sets. We used 675 image patches, 2,700 image patches and 13,500 image patches in the training which correspond to 25 ultrasound images, 100 ultrasound images and 500 ultrasound images, respectively. In the figure,

colors indicate the size of the training set. Epoch numbers and learning rates in the training were chosen as 2,000 and 5e-6 for 25 ultrasound images, 1,500 and 1e-5 for 100 ultrasound images, 400 and 1e-5 for 500 ultrasound images.

Several observations can be made form Fig. 2.7. For the small and medium sets, when the testing zone moved closer to the transducer by 0.4 cm, classification accuracy dropped to 70 percent. However, when the testing zone moved away from the transducer by 0.4 cm, classification accuracy remained above 80 percent. Similarly, for the large set, when the testing zone moved closer to the transducer by 0.4 cm, classification accuracy dropped to below 80 percent. However, when the testing zone moved away from the transducer by 0.4 cm, classification accuracy remained well above 85 percent. Similar observations can be made at other spatial locations as well.

In Fig. 2.8, similar to Fig. 2.7, we plot classification accuracy as the y axis and relative distance between testing zone and training zone as the x axis. In this figure, we experiment with the pre-focal zone and the post focal zone in addition to the on focus zone. When we trained AlexNet by using patches from the pre-focal zone, we tested the network with patches centered at 0.6 cm, 0.8 cm, 1 cm, 1.2 cm, 1.4 cm, 1.6 cm, 1.8 cm, 2 cm and 2.2 cm. When we trained a CNN by using patches from the post focal zone, we tested the network with patches centered at 1.8 cm, 2 cm, 2.2 cm, 2.4 cm, 2.6 cm, 2.8 cm, 3 cm, 3.2 cm and 3.4 cm. In this result, we used a fixed training set size, which resulted in 13,500 image patches or 500 ultrasound images. In the figure, colors represent the training zone. Epoch numbers and learning rates in the training were chosen as 400 and 1e-5 for all zones.

Several observations can be made from Fig. 2.8 results. When the testing zone was closer to the transducer by 0.4 cm, classification accuracies were slightly lower than 90 percent, slightly lower than 80 percent and 75 percent for the post focal zone, the on focus zone and the pre-focal zone, respectively. However, when the testing zone moved away from the transducer by 0.4 cm, classification accuracies were around 90 percent for all zones. Second, we observed that the post focal zone was the most robust zone against the shift in the testing. Classification accuracy for the post focal zone remained approximately above 80 percent in all shifts.

In Fig. 2.9, we plot classification accuracy as the y axis and axial zone width as the x axis for the on focus zone. In *Zone Training*, we extracted three

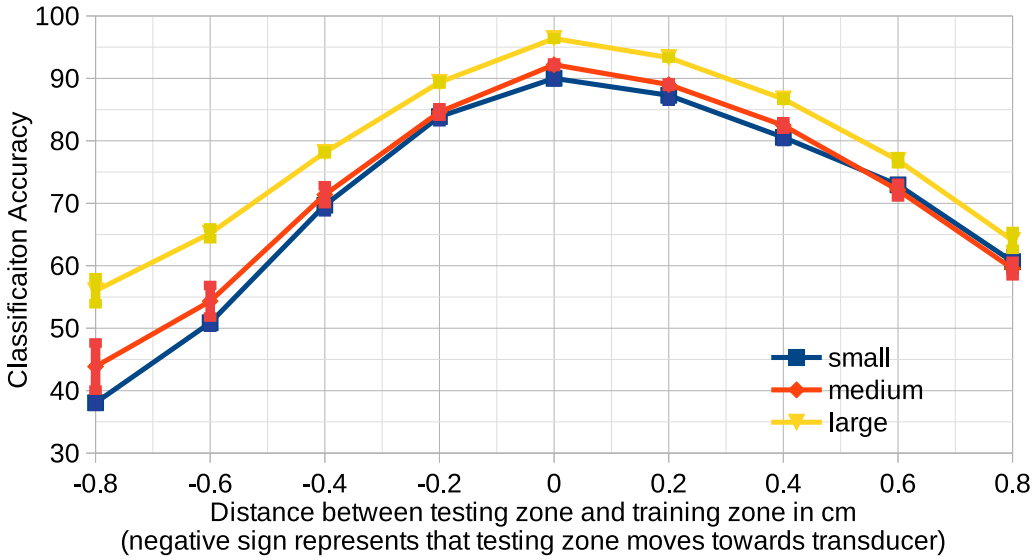


Figure 2.7: Sweeping testing zone center for a network trained by using the on focus zone: Classification Accuracy vs distance between the training and testing zones for different data-set sizes. The colors indicate the size of the training set. The blue color is for 675 patches which is labeled as small, the red color is for 2,700 patches which is labeled as medium and the yellow is for 13,500 patches which is labeled as large.

overlapping patches per ultrasound image as described in Section 2.2.3 and shown in Fig. 2.4. In this result, we make an exception to experiment with zone width, which was defined in terms of number of patches. We now extract 3, 6 and 9 overlapping patches from each ultrasound image for the on focus zone and these numbers form the x axis. Specifically, extracting 3 patches coincides with the original on focus zone definition, while extracting 6 patches coincides with merging pre-focal and on focus zones; and extracting 9 patches coincides with *Regular Training*. Additionally, we used three different sizes for the training set. We used 25 ultrasound images, which corresponds to 675, 1,350, 2,025 training image patches when we extract 3, 6 and 9 patches from each ultrasound image, respectively. Similarly, we used 100 ultrasound images, which corresponds to 2,700, 5,400, 8,100 training image patches and we used 500 ultrasound images, which corresponds to 13,500, 27,000, 40,500 training image patches. As a side note, for this graph, we used the same training and testing zones, unlike the previous two graphs, and colors in the graph represent training set sizes. Moreover, epoch numbers and learning rates were were chosen in accordance with the previous figures. From the Fig.

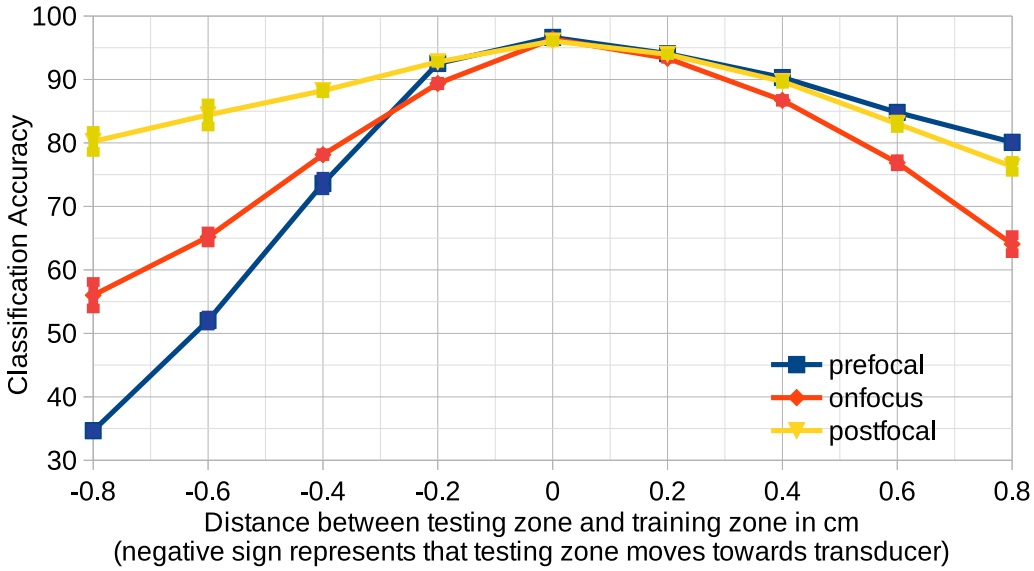


Figure 2.8: Sweeping testing zone center for networks trained on different zones: Classification Accuracy vs distance between training and testing zones. Colors represent the training zone. The blue color is for the pre-focal zone, which is labeled as prefocal. The orange color is for the on focus zone, which is labeled as onfocus. The yellow color is for the post focal zone, which is labeled as postfocal.

2.9, one can observe that for the small data set size increasing the number of patches, i.e., broadening the zone size, resulted in poorer classification.

In Fig. 2.10, we plot classification accuracy as the y axis and zone center as the x axis. For this graph, we tested and trained networks from the same zone while sweeping the zone center axially. We trained and tested our networks for zones centered at 1.2 cm, 1.4 cm, 1.6 cm, 1.8 cm, 2.0 cm, 2.2 cm, 2.4 cm, 2.6 cm and 2.8 cm. We repeat the experiments for different sizes of training sets. We used 675 image patches (25 ultrasound images), 2,700 image patches (100 ultrasound images) and 13,500 image patches (500 ultrasound images) in the training. Epoch numbers and learning rates were chosen in accordance with the previous figures.

2.3.2 Training Set Size vs Classification Accuracy

This section compares the performance of *Zone Training* against that of *Regular Training* and *Depth-Aware Training* under various data conditions from low data size regimes to larger data size regimes to investigate if *Zone*

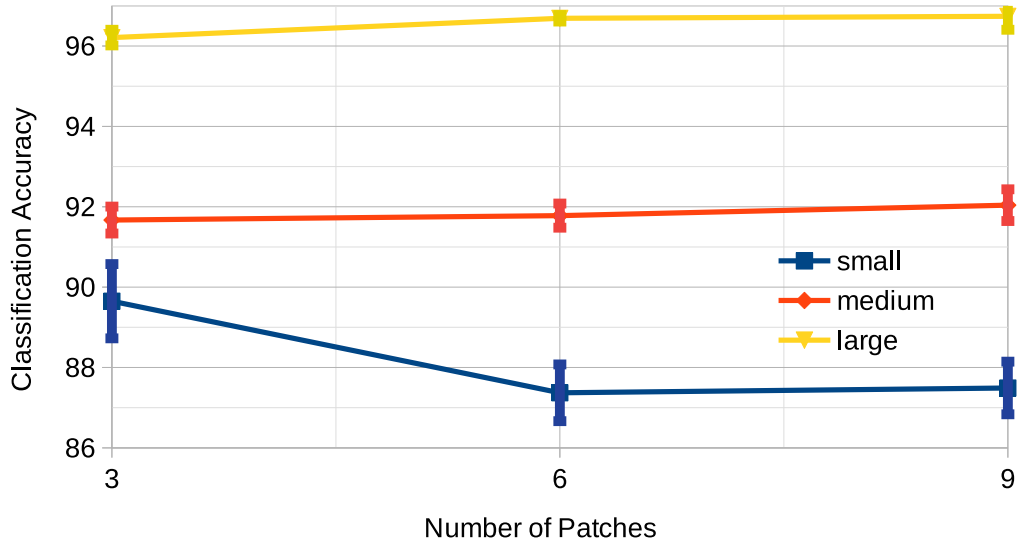


Figure 2.9: Classification accuracy vs axial zone width for the on focus zone. The colors indicate the size of the training set. The blue color is for 25 ultrasound images which is labeled as small, the red color is for 100 ultrasound images which is labeled as medium and the yellow is for 500 ultrasound images which is labeled as large.

Training is more successful when there is a small amount of data. Tables 2.3 - 2.8 are confusion matrices that list the classification accuracies for different training and testing strategies by using training set sizes of 10, 25, 50, 100, 200 and 500 ultrasound images. Rows represent training strategies: The first row, denoted as pre-focal, is for training with patches from the pre-focal zone. The second row, denoted as on focus, is for training with patches from the on focus zone. The third row, denoted as post focal, is for training with patches from the post focal zone. The fourth row, denoted as regular, is for training with *Regular Training* strategy. The last row, denoted as depth-aware, is for training with *Depth-Aware Training* strategy. Columns represent testing strategies: testing with patches from the pre-focal zone, testing with patches from the on focus zone, testing with patches from the post focal zone and testing with complete field of view, respectively, from first to last column. Epoch numbers and learning rates were were chosen in accordance with the previous figures, which were 2,500 and 5e-6 for 10 ultrasound images, 2,000 and 5e-6 for 25 ultrasound images, 2,000 and 1e-5 for 50 ultrasound images, 1,500 and 1e-5 for 100 ultrasound images, 1,000 and 1e-5 for 200 ultrasound

images, 400 and 1e-5 for 500 ultrasound images.

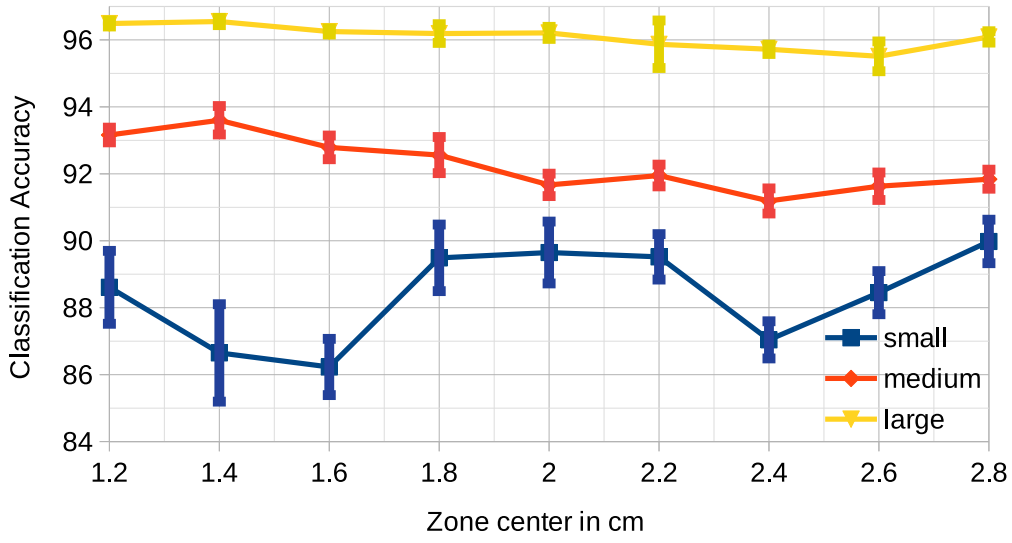


Figure 2.10: Classification accuracy vs zone center for different data-set sizes. Colors represent training set sizes where the blue color is for 675 patches which is labeled as small, the red color is for 2,700 patches which is labeled as medium and the yellow is for 13,500 patches which is labeled as large.

The tables verify that *Zone Training* had better classification accuracy than *Regular Training* and *Depth-Aware Training* in the low data regime. When we used 10, 25 or 50 ultrasound images in training, *Zone Training* performed 1-5 percent better than *Regular Training* and 1-4 percent better than *Depth-Aware Training*. Additionally, *Depth-Aware Training* performed approximately 1 percent better than *Regular Training* for all training set sizes. Lastly, the performance of the zones varied as the size of the training set was reduced. The classification accuracy dropped around 17 percent when we used 10 ultrasound images in training in comparison to 500 ultrasound images for the pre-focal pattern. For the on focus pattern, the same percentage drop was approximately 12 points and for the post focal pattern, the same percentage drop was around 10 points.

Table 2.3: Classification Accuracies with 10 ultrasound images

	Pre	On	Post
Pre-Focal	**81.01±3.20**	75.18±3.67	38.99±1.91
On Focus	53.00±4.25	**86.77±2.81**	74.56±3.33
Post Focal	42.78±3.90	74.00±4.82	86.96±1.88
Regular	78.62±2.53	83.52±2.00	85.14±2.13
Depth-Aware	80.74±2.43	83.96±3.17	**87.27±2.45**

Table 2.4: Classification Accuracies with 25 ultrasound images

	Pre	On	Post
Pre-Focal	**86.65±2.91**	77.70±4.14	42.73±9.93
On Focus	54.19±2.86	**89.65±1.85**	73.07±3.11
Post Focal	43.11±3.61	73.17±4.40	88.45±1.29
Regular	81.64±2.80	87.49±1.31	87.35±1.73
Depth-Aware	82.79±1.52	87.89±0.91	**88.81±1.23**

Table 2.5: Classification Accuracies with 50 ultrasound images

	Pre	On	Post
Pre-Focal	**91.66±0.93**	83.91±2.41	56.60±8.91
On Focus	53.42±3.12	**90.42±0.96**	73.98±3.51
Post Focal	44.75±3.10	77.87±2.98	89.63±1.12
Regular	89.66±1.61	89.23±1.85	89.01±1.06
Depth-Aware	90.15±1.90	90.32±1.49	**89.91±1.95**

Table 2.6: Classification Accuracies with 100 ultrasound images

	Pre	On	Post
Pre-Focal	**93.60±0.85**	84.09±1.51	62.19±8.20
On Focus	57.60±3.94	91.67±0.67	74.40±2.06
Post Focal	48.62±3.13	78.53±4.54	91.63±0.82
Regular	93.36±1.22	92.04±0.79	92.08±0.57
Depth-Aware	93.58±1.17	**92.61±0.99**	**92.77±1.06**

Table 2.7: Classification Accuracies with 200 ultrasound images

	Pre	On	Post
Pre-Focal	94.69±0.41	84.43±0.56	63.77±3.93
On Focus	61.45±4.47	94.19±0.35	76.06±2.28
Post Focal	57.95±5.66	83.79±1.98	93.41±0.53
Regular	94.79±0.35	94.50±0.41	94.00±0.35
Depth-Aware	**95.17±0.27**	**95.00±0.29**	**95.08±0.36**

2.4 Discussion

We proposed a DL training strategy, named *Zone Training*, where we split the complete field of view into zones such as the pre-focal, the on focus and

Table 2.8: Classification Accuracies with 500 ultrasound images

	Pre	On	Post
Pre-Focal	96.55±0.22	84.53±0.88	66.53±5.62
On Focus	62.25±4.92	96.21±0.36	75.87±3.05
Post Focal	58.97±8.04	82.34±2.74	95.51±0.89
Regular	97.09±0.77	96.74±0.67	96.75±0.32
Depth-Aware	**97.40±0.10**	**97.32±0.16**	**97.15±0.16**

the post focal zones. Then, we trained separate networks for each zone. We investigated *Zone Training* thoroughly by experimenting with zone definitions and their behavior under different training set sizes.

The figures provide several important observations. From Fig. 2.7, we observed that as the testing zone moved towards the transducer, classification accuracy dropped faster and it was valid for small, medium and large training set sizes. The observation indicates that the pre-focal diffraction pattern was more complicated and it changed faster than the post focal diffraction pattern.

In Fig. 2.8, we quantified how classification accuracy decreased when the testing zone moved away from the training zone for training with the on focus zone, the pre-focal zone and the post focal zone. First, we observed that when the testing zone was closer to the transducer, classification accuracies dropped faster for the pre-focal and on focus zones. For the post focal zone, classification accuracies were relatively symmetric around the zone center. This further verified our previous observation stating that the pre-focal pattern was more complicated and changed quickly in comparison to the post focal pattern. Another observation was that for the pre-focal training, classification accuracy deteriorated slowly when the testing zone moved away from the transducer in comparison to the testing zone moving towards the transducer, which further illustrates the complicated behaviour of the pre-focal pattern.

In Fig. 2.9, we investigated the relationship between classification accuracy and zone width in terms of overlapping patches for the on focus zone. First, we observed that as we increased the number of patches, classification accuracy remained relatively constant for the medium size training set, while classification accuracy slightly increased for the large size training set. However, classification accuracy dropped as we increased the number of patches

for the small size training set. Specifically, classification accuracy dropped to around 87 percent from 90 percent as we increased the number of patches from 3 to 6 and it stayed relatively constant when we increased the number of patches to 9. These observations indicate that *Zone Training* was more robust when the training set size was smaller. However, *Regular Training* can be preferable when the training set size was larger.

In Fig. 2.10, we determined the best zone location axially in terms of classification accuracy. We observed that the relationship between classification accuracy and the zone location depended on the training set size. For the large training set size, the classification accuracy stayed relatively constant around 96 percent in all axial locations. For the medium training set size, the classification accuracy degraded from approximately 94 percent to 92 percent when the zone location moved from 1.2 cm to 2.8 cm. That indicates the pre-focal zones are the most desirable zones for the medium size training set size. However, for the small training set size, the zones around the on focus zone are the most desirable. Figure 2.10 is useful to determine the most optimal zone center to characterize tissue samples for different training conditions. However, using a single zone is only meaningful when the phantoms are uniform and we don't loose any information by discarding other zones in our decision process. If there is some spatial information to be taken advantage of in our classification decision or we want to increase classification accuracies by using all information that we have, then we need to separate the complete field of view into multiple zones and train multiple expert networks to be used in a voting schema. In that case, Fig. 2.10 is still useful for determining which expert network should have higher effect in a voting schema.

In Tables 2.3 - 2.8, we presented confusion matrices to quantify classification accuracies with respect to different training set sizes, which are 10, 25, 20, 100, 200 and 500 ultrasound images. First, *Zone Training* had better classification accuracy than *Regular Training* when the training data was scarce. However, when the training data size was larger, *Regular Training* performed better than *Zone Training*. For example, when we used 200 or 500 ultrasound images in training, *Regular Training* performed around 1 percent better than *Zone Training* for all zones. However, when we used 10, 25 or 50 ultrasound images in training, *Zone Training* performed better than *Regular Training*. When we used 100 ultrasound images, *Regular Training* and *Zone*

Training performed similarly. Second, *Depth-Aware Training* was always better than *Regular Training*. Third, *Zone Training* was slightly better than *Depth-Aware Training* when the training data was scarce. When we used 10, 25 or 50 ultrasound images in training, *Zone Training* performed better than *Depth-Aware Training* for the pre-focal and on focus zones. However, they performed similarly for the post focal zone. Moreover, the post-focal pattern was more robust against decreasing training set size in comparison to the on focus pattern; and the on focus pattern was more robust in comparison to the pre-focal pattern for all training strategies (*Zone, Regular and Depth-Aware Training*). Finally, the results from the tables indicate that the training set should comprise data specifically from areas that are task relevant. For example, if a clinical imaging session investigates data in the pre-focal zone, but the training data came from the focal or post-focal zone, the classification performance may degrade significantly.

In light of the results, there are multiple emerging directions to which the proposed method could be extended. While the method was applied to tissue classification, it has the potential to be applicable to other deep learning applications such as detection, segmentation, and image formation. Therefore, *Zone Training* can be tested for different applications. Additionally, it would be interesting to test *Zone Training* with different types of neural network structures, even though *Zone Training* is not directly related to the neural network structure. Moreover, the optimal number of zones should be investigated in greater detail. The optimal number of zones can change from problem to problem depending on the imaging substrates, imaging system, problem complexity and imaging settings. It is also important to consider varying zone widths for different focal zones, as this can have a significant impact on the optimal number of zones, e.g., breaking the pre-focal zone into multiple, smaller zones might improve the accuracy in the pre-focal region. Furthermore, different patch sizes including pixel-wise classification, which is known as image segmentation, can be investigated within the context of *Zone Training*. Last, the implementation code of this chapter is available at https://github.com/usoylu2/zonetrain.

CHAPTER 3

CALIBRATING SETTING MISMATCHES IN DEEP LEARNING FOR QUANTITATIVE ULTRASOUND BY USING TRANSFER FUNCTIONS

3.1 Motivation

The clinical environment is too complicated to be fully realized in a development setting where operators, such as sonographers, are adjusting settings to obtain the best perceived image quality. Therefore, there will be inevitable data mismatches if the operator is given the freedom to select settings that provide perceived optimal image quality and these settings do not match scanner settings associated with the training data. Specifically, we consider acquisition-related data mismatches, i.e., data mismatches caused by variations in scanner settings such as the number of foci and their locations or pulse frequency, and develop a method by looking at the problem from a signals and systems perspective. Such data mismatches are pervasive in biomedical ultrasound imaging due to its operator-dependent and patient-dependent nature and mitigating their effects is essential for wider clinical adoption of DL-based methods for tasks such as tissue classification.

Any form of learning algorithm would be ineffective if there are data mismatches between training and testing data, and no assumptions can be made to calibrate the mismatches. Ideally, we need to collect a large and diverse training data set at each imaging setting to completely eliminate data mismatches caused by scanner parameters. However, acquiring such a training data set can be extremely expensive. Another approach could be training on a subset of imaging settings, which makes the data generation and gathering process less expensive. However, there will still be generalization issues for the settings that are not included in the training set. The question we address in this chapter is "How can we mitigate acquisition-related data mismatches in an inexpensive and generalizable way for QUS?"

To accomplish this, we consider a systems response approach. Biomed-

ical ultrasound imaging can be viewed as a system, which encodes all the information related to an imaging system working on a tissue signal, which encodes all the information related to an imaging substrate. Subsequently, an image obtained by an ultrasound imaging system can be decomposed into two parts: a system response and a tissue signal. Then, the problem of acquisition-related data mismatch can be posed as matching system responses. When training inputs and testing inputs are from different distributions due to different scanner settings, such as a different excitation pulse frequency, there are two different system responses in the imaging process, one corresponding to training and the other one corresponding to deployment or testing. Potentially, a function can be defined that allows a system to transfer from one environment to the other environment.

More precisely, under a single scattering approximation and when at least one aperture diameter away from the transducer surface, the backscattered frequency spectrum from a medium can be represented as [138]

$$W(f, \mathbf{x}) = T(f, \mathbf{x})A(f, \mathbf{x})D(f, \mathbf{x})H(f)R(f, \mathbf{x}) \tag{3.1}$$

where f represents frequency, x represents axial direction, $T(f, \mathbf{x})$ incorporates the transmission losses between tissues, $A(f, \mathbf{x})$ is the frequency-dependent attenuation, $D(f, \mathbf{x})$ represents the diffraction effects of the transducer, $H(f)$ is the impulse response of the transducer system and incorporates the electro-mechanical response, and $R(f, \mathbf{x})$ is the scattering function describing the underlying tissue micro-structure. Therefore, an ultrasound image can be naively decomposed as

$$W(f, \mathbf{x}) = S_{\phi}(f, \mathbf{x})P(f, \mathbf{x}) \tag{3.2}$$

where $S_{\phi}(f, \mathbf{x}) = D(f, \mathbf{x})H(f, \mathbf{x})$ is the system response which incorporates all the information related to ultrasound imaging system and $P(f, \mathbf{x}) = T(f, \mathbf{x})A(f, \mathbf{x})R(f, \mathbf{x})$, which is the tissue or sample signal that incorporates all the information related to imaging substrate (i.e., attenuation, transmission losses and scattering function). The subscript ϕ in $S_{\phi}(f, \mathbf{x})$ represents the scanner setting, exclusively ϕ_{train} stands for training environment and ϕ_{test} stands for testing environment, for later use. To calibrate the acquisition-related data mismatches between two system settings, we setup

the scenario where the tissue signal, $P(f, \mathbf{x})$, does not change between testing and training, such that,

$$\frac{W_{test}(f, \mathbf{x})}{W_{train}(f, \mathbf{x})} = \frac{S_{\phi_{test}}(f, \mathbf{x})}{S_{\phi_{train}}(f, \mathbf{x})} \tag{3.3}$$

$$= \Gamma(f, \mathbf{x}) \tag{3.4}$$

where $\Gamma(f, \mathbf{x})$ represents the *"setting transfer function"* between training and testing system parameters. This can be done in a practical way by selecting a tissue mimicking phantom with uniform scattering properties and fixing the transducer to scan and record the signal from a single location in the phantom while the settings are changed from training to testing. To calibrate in training time, $\Gamma(f, \mathbf{x})$ is sufficient:

$$W_{train \to test}(f, \mathbf{x}) = \Gamma_{train \to test}(f, \mathbf{x}) W_{train}(f, \mathbf{x}) \tag{3.5}$$

and hence, it is a convolution operation in time direction, t

$$w_{train \to test}(t, \mathbf{x}) = \gamma_{train \to test}(t, \mathbf{x}) *_t w_{train}(t, \mathbf{x}) \tag{3.6}$$

where $\gamma_{train \to test}$ represents the *"setting transfer filter"* for training time. For the train-time calibration, a DL model is trained at testing data settings, and therefore, testing data can be input into the DL model directly. In order to achieve this, the training data are converted to the testing data via the setting transfer function $\Gamma_{train \to test}(f, \mathbf{x})$ in training time and a new model is developed by the network, which allows the testing data settings to match with training settings used to create the model.

Similarly, $\Gamma_{test \to train}(f, \mathbf{x}) = \Gamma^{-1}_{train \to test}(f, \mathbf{x})$ is sufficient for calibrating in testing time:

$$W_{test \to train}(f, \mathbf{x}) = \Gamma_{test \to train}(f, \mathbf{x}) W_{test}(f, \mathbf{x}), \tag{3.7}$$

which results in the following filtering operation

$$w_{test \to train}(t, \mathbf{x}) = \gamma_{test \to train}(t, \mathbf{x}) *_t w_{test}(t, \mathbf{x}) \tag{3.8}$$

where $\gamma_{test \to train}$ represents the *"setting transfer filter"* for testing time. For the test-time calibration, a DL model is trained at the training data settings,

and therefore, testing data cannot be input into the DL model directly. The testing data needs to be converted to training data via the setting transfer function $\Gamma_{test \to train}(f, \mathbf{x})$. Then, the models developed with the original training data and its associated settings are used with the converted testing data.

Therefore, we propose to use a signals and systems perspective to *calibrate* training or testing data. The proposed method is depicted in Figure 3.1. Here is the proposed method step by step at a high-level:

1. Gather a *large* amount of training data at a single setting.
2. Gather a *small* amount of calibration data at each scanner setting.
3. Calculate W_{test} or W_{train} by using the calibration set to calculate the setting transfer function $\Gamma_{test \to train}$ and $\Gamma_{train \to test}$.
4. Construct linear phase filters $\gamma_{test \to train}$ or $\gamma_{train \to test}$ by using the magnitude responses of $\Gamma_{test \to train}$ and $\Gamma_{train \to test}$.
5. When a different setting than training setting is being used in the scanning process, calibrate the mismatch by using filters $\gamma_{test \to train}$ or $\gamma_{train \to test}$ either in the training time or in the testing time in the DL network, respectively.

The proposed method is an inexpensive way of mitigating generazibility issues caused by acquisition-related data mismatches in biomedical ultrasound imaging. In this chapter, we utilized a setup similar to the one developed in the previous section. DL approaches were trained to classify raw RF data to identify distinct tissue-mimicking phantoms. This chapter shows that a reference phantom can be used to calibrate the setting mismatch in DL-based techniques similar to spectral-based QUS approaches.

To further motivate the proposed method, some of the well-established methods from the literature are discussed. Transfer learning (TL) is a technique that aims to address the issue of data mismatch by first training a learning model and then fine-tuning it [108, 109]. TL is potentially more costly than the proposed calibration method because it requires diverse sample data in the testing domain to fine-tune a model. In the proposed method, a single frame from a single calibration source from the testing domain is sufficient. The problem of data mismatch has also become more prominent in

recent literature on DL-based QUS [103, 111–113, 115]. Therani et al. [111] utilized reference phantoms which have known scatter number density to mitigate system dependency in the problem of classifying scatterer number density through Adaptive Batch Normalization. In contrast, the proposed calibration method employs a single reference phantom that is not dependent on the type of classes. In another interesting work, Sharifzadeh et al. [115] proposed replacing the magnitude of the low-frequency spectrum inspired by Fourier Domain Adaption (FDA) in the field of computer vision. Unlike that work, the method proposed here is capable of utilizing the entire frequency spectrum by requiring only a single frame from the testing domain. Additionally, Tierney et al. [103] used cycle-consistent generative adversarial networks to eliminate data mismatches in ultrasound images. It should be noted, however, that generative models require a larger amount of diverse data in the testing domain, making them more resource-intensive in comparison to the method proposed in this paper.

3.2 Methods

3.2.1 Phantoms

Two different tissue-mimicking phantoms were classified in the experiments, which we designated as Phantom1 and Phantom2, and another phantom was used as the calibration phantom. They are cylindrically shaped as shown in Fig. 3.2.

Phantom1, which mimics soft tissue, has been described by Wear et al. [130]. Their materials were produced based on the method of Madsen et al. [131] and they are macroscopically uniform. The only nonuniformity results from the random positioning of microscopic glass bead scatterers. Phantom1 had a measured attenuation coefficient slope of approximately 0.7 $\mathrm{dB}\times\mathrm{cm}^{-1}\times\mathrm{MHz}^{-1}$, respectively. The component materials and their relative amounts by weight for Phantom1 are agarose (2.34%), n-propanol (2.92%), 75 to 90 μm-diameter glass beads (1.87%), bovine milk concentrated 3 times by reverse osmosis (47.9%), liquid Germall Plus preservative (International Specialty Products, Wayne, NJ) (1.87%), and 18-$M\Omega$-cm deionized water (43.1%).

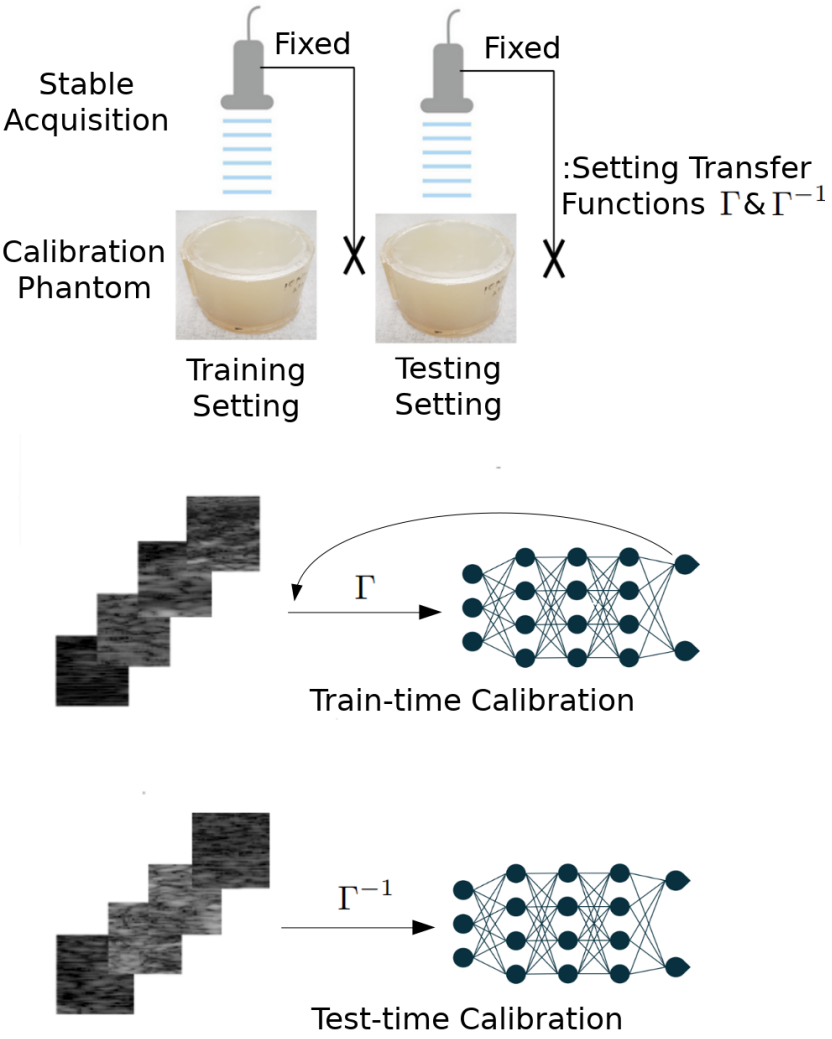
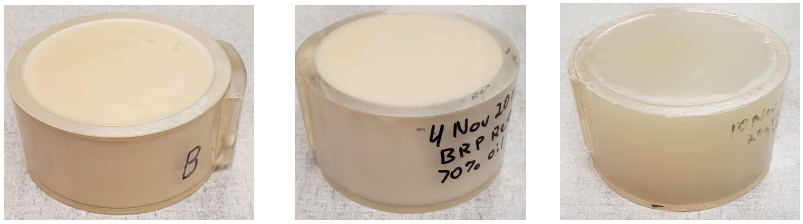


Figure 3.1: Setting Transfer Function

Phantom2 has been described by Nam et al. [139] as a reference phantom. Its measured attenuation coefficients at frequencies from 2 to 10 MHz were fit to a power law function of frequency, $\alpha(f) = 0.256f^{1.366}$, where f is the frequency in terms of MHz and $\alpha(f)$ is in terms of dB/cm. The phantom was made with 6.4 g of 5-43 μm-diameter glass beads uniformly distributed spatially at random in a gel background. The background material was a gelatin emulsion containing 70% safflower oil [140].

The calibration phantom was a low attenuation phantom, which was constructed as described by Anderson et al. [132]. It had a weakly scattering 2% agar background with 150-180 μm glass beads, which had a slightly broader

distribution of scatterer sizes (160 ± 60 μm). The glass bead concentration was 20 g/L and the beads were randomly distributed spatially within the phantom.



Phantom1 Phantom2 Calibration

Figure 3.2: Photographs of The Classification and Calibration Phantoms

3.2.2 Ultrasound Imaging Device and Imaging Settings

Ultrasound gel was placed on the surfaces of the phantoms and then the phantoms were scanned with an L9-4/38 transducer using a SonixOne system (Analogical Corporation, Boston, MA, USA) providing an analysis bandwidth of 2-7.5 MHz and focusing with an F-number of 3 for transmit and receive. Ultrasound frames of post-beamformed RF data sampled at 40 MHz were acquired from each of the phantoms and saved for offline processing. The ultrasound post-beamformed RF data were directly used in the training, test and calibration processes. The imaging array had a center frequency measured at 5.5 MHz and was operated with a fixed elevational focus of 1.9 cm. Scanner parameters that were adjusted for each experiment can be found in Table 3.1. Changes in the focus occurred on transmit. We acquired data from the phantoms via two scanning procedures. In the first, we recorded a video of 1,007 ultrasound frames by free-hand motion. Free-hand acquisition provided us a large data set of independent frames for each phantom to be used in the training and testing. In the second procedure, we stabilized the transducer using a bar clamp holder and then recorded 10 identical frames at both the training and testing settings from the exact same location in the phantom which provided us with the calibration data to be used in calculating the setting transfer functions.

Table 3.1: Scanner Parameters for SonixOne System

	Pulse Freq	Focus	Output Power
Training in Sec.3.3.1	9 MHz	@2cm	0 dB
Test in Sec.3.3.1	5 MHz	@2cm	0 dB
Training in Sec.3.3.2	9 MHz	@2cm	0 dB
Test in Sec.3.3.2	9 MHz	@1cm & 3cm	0 dB
Training in Sec.3.3.3	9 MHz	@2cm	0 dB
Test in Sec.3.3.3	9 MHz	@2cm	-6 dB

3.2.3 Data-Set

The total size of an ultrasound image frame from the phantoms was 2,080 pixels $\times$ 256 pixels. There were 2,080 samples along the axial direction that corresponded to a 4 cm imaging depth. Even though the L9-4/38 transducer has 128 channels, the SonixOne system interpolates to 256 channels that correspond to 256 lateral samples. The data used in the DL network were the raw backscattered RF data. The data-set of ultrasound frames is also publicly available at https://figshare.com/s/7ae94a537a56e5db3525.

After acquiring ultrasound frames by either stable acquisition or free-hand acquisition, we extracted square data patches from the image frames whose sizes were 200 samples $\times$ 26 samples that correspond to square image patches whose size were 4 mm $\times$ 4 mm in physical dimensions, to be used in training, validation and testing sets. The motivation behind patch extraction was described in our previous work through a clinical scenario [141]. For instance, ultrasound imaging can be used to examine and characterize tumors, whether benign or malignant. When using QUS approaches for tumor characterization, a region of interest (ROI) is selected inside the tumor to examine the signals from the tumor. Therefore, we performed the patch extraction in this work to examine the proposed method in the context of DL-based QUS.

From one ultrasound image, we could extract 81 (9 lateral $\times$ 9 axial) image patches as depicted in Fig. 3.3. While extracting image patches, we did not use the first 540 pixels in the ultrasound image. Axially, we obtained the next line of individual patches by translating the start of the next patch by 100 pixels along the axial depth. Laterally, we obtained the next line of individual patches by translating the start of the next patch by 26 pixels along the axial depth. Overall, in patch extraction, there were 9 axial lines and

9 lateral lines to extract individual patches that led to extracting 81 image patches per ultrasound image. As a consequence, the training set consisted of image patches extracted from ultrasound frames acquired at scanner settings for the training. On the other hand, ultrasound frames acquired for the test data were split into two sets: one was for the validation set and the other was for the test set.

In training, we extracted 81,000 patches, which is equal to 1,000 frames $\times$ 81 patches per frame. The 1,000 frames were randomly selected out of 2,014 total ultrasound frames at the scanner setting for the training. Similarly, after acquiring 2,014 frames at scanner settings for the testing, as the validation and test sets, we randomly selected 750 ultrasound frames out of 2,014 ultrasound frames, which resulted in 60,750 patches for validation and testing. We repeated the random selection of training and testing patches ten times for each experiment.

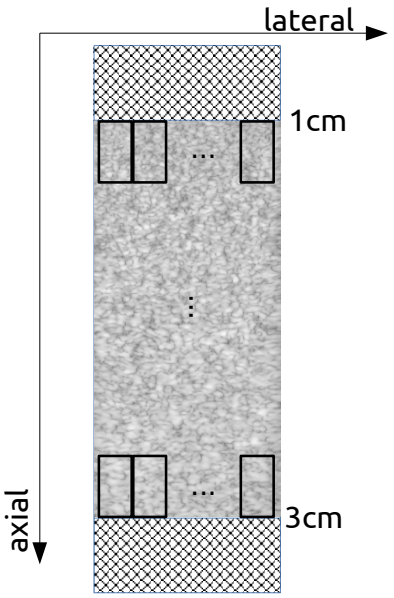


Figure 3.3: Patch Extraction: Local patches, whose sizes were 200 $\times$ 26 samples, were extracted to be input into a CNN. The first 540 samples were not used. Each frame resulted in 81 extracted patches due to the 9 axial and 9 lateral lines used for patch extraction.

3.2.4 Network Structure

In this work, we used two well-known CNN architectures, i.e., ResNet-50 [96] and DenseNet-201 [142]. CNNs have several advantages among other DL structures for the tasks related to 2D images. CNNs are similar to the human visual system, which makes them effective at learning and extracting abstractions of 2D images [136]. The CNN architectures were slightly modified and can be found in Tables 3.2 and 3.3. The main motivation for using architectures like ResNet-50 and DenseNet-201, which are relatively larger, is that these models have significant potential in solving real clinical tasks. They are representative of the types of models that would likely be used in clinical settings. Furthermore, in the context of data mismatch, it is known that deeper architectures are harder to calibrate due to their multiple layers, especially batch normalization layers. Therefore, being able to calibrate these models effectively would indicate that the proposed method can also calibrate simpler architectures.

Table 3.2: ResNet-50

stage	output	kernels
conv1	100×13	$7 \times 7, 64$, stride 2
conv2	50×7	3×3 max pool, stride 2
		$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3	25×4	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
conv4	13×2	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
conv5	7×1	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	global average pool 2-d fc, softmax

Table 3.3: DenseNet-201

stage	output	kernels
conv1	102×15	$7 \times 7, 64$, stride 2
dense1	51×8	3×3 max pool, stride 2
		$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix} \times 6$
tr1	25×4	1×1, 256 and 2×2 avgpool, stride 2
dense2	25×4	$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix} \times 12$
tr2	12×2	1×1, 512 and 2×2 avgpool, stride 2
dense3	12×2	$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix} \times 48$
tr3	6×1	1×1, 1024 and 2×2 avgpool, stride 2
dense4	6×1	$\begin{bmatrix} 1 \times 1 \\ 3 \times 3 \end{bmatrix} \times 32$
	1×1	global average pool 2-d fc, softmax

3.2.5 Training

The DL training was done on a machine having a TITAN RTX and on two machines each having a RTX A5000. Each experiment was conducted separately on one of the three available GPUs. All implementations were done with the PyTorch library [134].

As a data preprocessing step, we applied z-score normalization at the patch level, i.e., the mean intensity value of patches was subtracted from each patch, and then, each pixel in a patch was divided by the standard deviation of the intensity of the patches. Then, the models were trained by using cross entropy loss with uniform class weights. Horizontal flip with 0.5 probability was implemented as a default data augmentation step in the training process. The batch number was chosen as 128 for all experiments. We used the Adam algorithm [135] as the optimizer in all experiments. The learning rates and the epoch numbers were determined to achieve "asymptotic test accuracy" by using the validation set.

After adjusting all the training parameters, we repeated training and testing for each experiment 10 times starting from random selection of ultrasound frames and dividing them into patches. Then, we calculated the mean classification accuracies and standard deviations in the test set. Also, we calculated mean area under the receiver operator characteristic curve (AUC) and

standard deviations. The metrics were calculated patch-wise and reported in Section 3.3 for each experiment.

3.2.6 Calibration

We first obtained $W_{train}(f, \mathbf{x})$ and $W_{test}(f, \mathbf{x})$, defined in (3.3) by using the calibration set obtained at testing and training settings. We gathered the calibration set by stabilizing the transducer array on top of a phantom and acquiring scans from the exact same view for each setting. For the training and testing settings, we acquired 10 identical frames from the calibration phantom to be used in averaging and reducing any systematic noise. Because of the stable acquisition setup, i.e. using the same tissue signal $P(f, \mathbf{x})$ in the calibration set, by taking ratios of $W_{train}(f, \mathbf{x})$ and $W_{test}(f, \mathbf{x})$, we obtained setting transfer functions $\Gamma_{train \to test}(f, \mathbf{x})$ and $\Gamma_{test \to train}(f, \mathbf{x})$. The ratios could be used either in training time by applying $\Gamma_{train \to test}(f, \mathbf{x})$ as shown in (3.5), which we call *"train-time calibration"* or in test time by applying $\Gamma_{test \to train}(f, \mathbf{x})$ as shown in (3.7), which we call *"test-time calibration"*.

In practice, we implemented the setting transfer functions in a manner inspired by the Wiener filter [143],

$$\Gamma^{Wiener} = \frac{|\Gamma|^{-1}}{|\Gamma|^{-2} + SNR^{-1}}. \tag{3.9}$$

In the formula above, Γ represents either $\Gamma_{train \to test}$ or $\Gamma_{test \to train}$. SNR was estimated through the power spectra of W_{train} and W_{test}, resulting in SNR_{train} and SNR_{test}. First, we determined a noise floor level by looking at the lowest values of the power spectra outside of the transducer bandwidth at 20 MHz. Then, at each frequency bin, we calculated the tissue signal level by subtracting the noise floor. Subsequently, we obtained SNR values by taking the ratios of the tissue signal and the noise floor at each frequency bin for both the power spectra of W_{train} and W_{test}.

$$SNR_{train} = \frac{|W_{train}|^2 - \min_f |W_{train}|^2}{\min_f |W_{train}|^2}, \tag{3.10}$$

$$SNR_{test} = \frac{|W_{test}|^2 - \min_f |W_{test}|^2}{\min_f |W_{test}|^2}. \tag{3.11}$$

Lastly, we took the minimum SNR value between SNR_{train} and SNR_{test} used that value in the filter.

$$SNR = min(SNR_{train}, SNR_{test}). \tag{3.12}$$

In this filter design, at frequency bins when the signal level of the setting transfer function was small compared to the noise, the filter acted like a denoising filter. When the signal level of the setting transfer function was high compared to noise, the filter was equivalent to the original Γ. This provides a robust way to use the complete bandwidth of setting transfer functions. For the rest of the study, $\Gamma_{train \to test}$ and $\Gamma_{test \to train}$ refers to the Wiener implementation, for simplicity.

In the patch extraction, patches originated from 9 different depth or axial lines as described in Sec. 3.2.3. Therefore, the power spectra W_{train} and W_{test}, and hence the setting transfer functions $\Gamma_{train \to test}$ and $\Gamma_{test \to train}$, were obtained in a depth aware manner, i.e., transfer functions were calculated for each axial line (i.e., each depth), resulting in 9 transfer functions per setting mismatch. After calculating $\Gamma_{train \to test}$ and $\Gamma_{train \to test}$, $\gamma_{train \to test}$ and $\gamma_{test \to train}$ from were constructed as FIR filters with linear phase from the given frequencies and corresponding gains. The number of taps in the FIR filter was searched in the hyperparameter optimization and selected as 51.

FIR filters were constructed by using scipy.signal.firwin2 function from Python. The implementation of convolution operations in (3.6) and (3.8) were done via torch.nn.functional.conv1d whose parameter padding was selected as 'same', which pads the input so the output has the same shape as the input. The implementation code can be found at https://github.com/usoylu2/calibration.

Train-time calibration and *test-time calibration* are explained in Algorithm 1 and Algorithm 2, respectively. X represents "patch-wise" post-beamformed RF data, I represents the Fourier spectrum of X and y represents the phantom identity. During the implementation of experiments, filtering operations due to $\Gamma_{train \to test}$ or $\Gamma_{test \to train}$ were conducted in the time domain. Along with patch data, the axial location was also tracked during patch extraction, resulting in transfer functions having axial location identity as well. Therefore, in the filtering operation, X and its corresponding M2M transfer

function based on axial location identity were utilized.

From Figs. 3.4-3.6, we plotted power spectra for different training-testing setting pairs corresponding to Table 3.2 along with the transfer function $\Gamma_{train \to test}(f, \mathbf{x})$. Both Wiener and plain versions of the transfer function are plotted. These graphs are obtained from data acquired at a fixed axial location, which is around 2 cm. The left sub-figures show power spectra W_{train} and W_{test}. Furthermore, the right sub-figures depict $\Gamma_{train \to test}(f, \mathbf{x})$ with Γ being the plain version and Γ_{Wiener} being the Wiener version. Additionally, Fourier Transform of the linear phase filter $\gamma_{train \to test}$ depicted as $\mathcal{F}\{\gamma_{train}\}$.

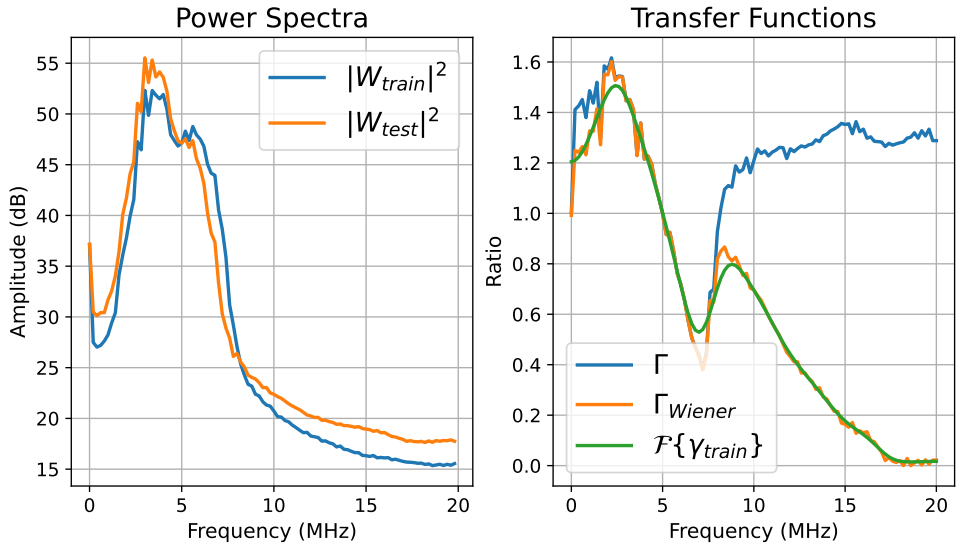


Figure 3.4: Example calibration plots for the pulse frequency mismatches

3.2.7 Transfer Learning (Fine-Tuning)

In this chapter, we used Fine-Tuning as the baseline method. We fine-tuned CNNs, which had been trained using a large amount of data acquired from the samples at the training setting, by using a smaller amount of data acquired from the samples using the testing setting. Specifically, we compared the proposed method with Fine-Tuning for three different data set sizes after training CNNs with the complete training set which consisted of 1,000 frames. For the first set, we fine-tuned the network using 2 diverse frames (162 patches) in training, 2 diverse frames in validation and 2 diverse frames

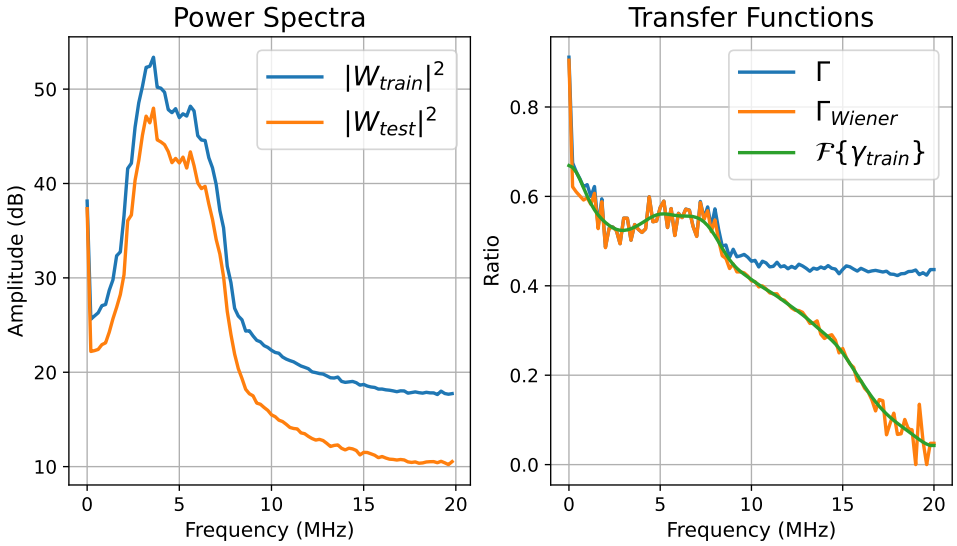


Figure 3.5: Example calibration plots for focal location mismatches

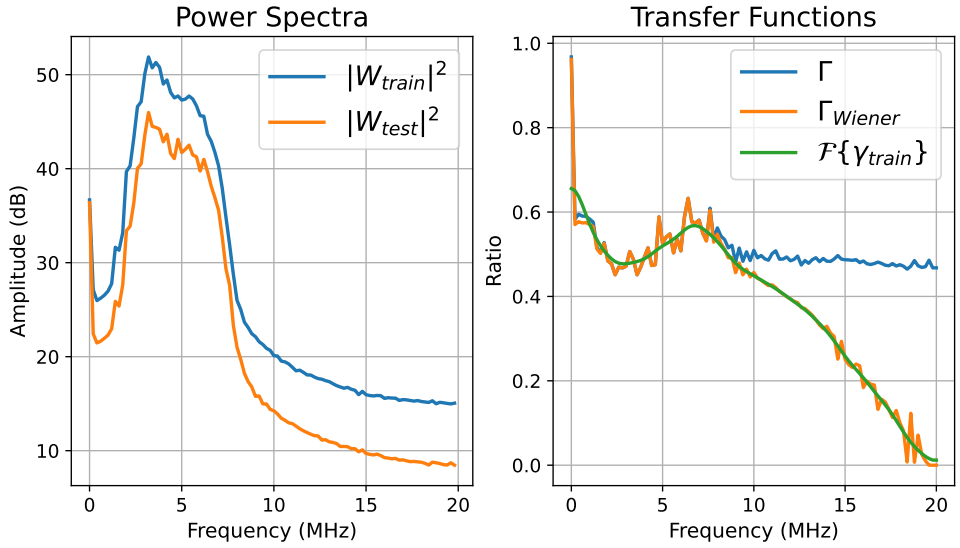


Figure 3.6: Example calibration plots for the output power mismatches

in testing. For the second, we used 10 diverse frames (810 patches) in training, 10 diverse frames in validation and 10 diverse frames in testing. For the third, we used 20 diverse frames (1,620 patches) in training, 20 diverse frames in validation and 20 diverse frames in testing. The major limitation for adopting the Fine-Tuning method, in comparison to the proposed calibration method, is that it requires acquisition of a set of diverse frames from

Algorithm 1: Train-Time Calibration

Data: Training Data: $\{X_{train}, y_{train}\}$, Testing Data: $\{X_{test}, y_{test}\}$, Calibration Data: $\{X_{calibration}\}$
Preparation: Calculate $\Gamma_{train \to test}$ & Normalize X_{train}
Step1: $W_{train}\&W_{test} \leftarrow \mathcal{F}\{X_{calibration}\}$;
Step2: $\Gamma_{train \to test} \leftarrow W_{test} \div W_{train}$;
Step3: $X_{train}, X_{validation} = \text{split}(X_{train})$;
Step4: $X_{train \to test} \leftarrow \Gamma_{train \to test}\{X_{train}\}$;
Step5: $X_{validation \to test} \leftarrow \Gamma_{train \to test}\{X_{validation}\}$;
Step6: $X \leftarrow (X - \mathbb{E}X_{train \to test})/\sigma X_{train \to test}$;
Step7: $X \leftarrow (X - \mathbb{E}X_{valid \to test})/\sigma X_{valid \to test}$;
Training: Launch a DL model f_θ
while *epoch* **do**
  Update parameters in f_θ ;
end
Inference: Measure Performance
Step1: Z-score Normalization of X_{test} via $(X - \mathbb{E}X_{train \to test})/\sigma X_{train \to test}$;
Step2: $y = f_\theta(X_{test})$;
Step3: Calculate the metrics using y and y_{test} ;

Algorithm 2: Test-Time Calibration

Data: Training Data: $\{X_{train}, y_{train}\}$, Testing Data: $\{X_{test}, y_{test}\}$, Calibration Data: $\{X_{calibration}\}$
Preparation: Calculate $\Gamma_{test \to train}$ & Normalize X_{train}
Step1: $W_{train}\&W_{test} \leftarrow \mathcal{F}\{X_{calibration}\}$;
Step2: $\Gamma_{test \to train} \leftarrow W_{train} \div W_{test}$;
Step3: $X_{train}, X_{valid} = \text{split}(X_{train})$;
Step6: $X_{train} \leftarrow (X - \mathbb{E}X_{train})/\sigma X_{train}$;
Step7: $X_{valid} \leftarrow (X - \mathbb{E}X_{valid})/\sigma X_{valid}$;
Training: Launch a DL model f_θ
while *epoch* **do**
  Update parameters in f_θ;
end
Inference: Measure Performance
Step1: $X_{test \to train} \leftarrow \Gamma_{test \to train}\{X_{test}\}$;
Step2: Z-score Normalization of $X_{test \to train}$ via $(X - \mathbb{E}X_{train})/\sigma X_{train}$;
Step3: $y = f_\theta(X_{test \to train})$;
Step4: Calculate the metrics using y and y_{test} ;

the actual samples at the the testing settings to be transferred to the model that was developed with data acquired using the training settings.

3.2.8 Model Interpretation

In this chapter, we investigated the interpretation of deep learning model behavior through pixel attribution. Especially in clinical tasks, the black-box nature of DL-based approaches is undesirable. Therefore, interpreting DL-based approaches would be beneficial in the context of QUS and data mismatch as well. We utilized pixel attribution maps that highlight the pixels in the input that were relevant for a certain image classification decision by a DL network. Specifically, Vanilla Gradient [144] and SmoothGrad [145] were utilized to obtain pixel attributions. In Vanilla Gradient, we visualize the gradient of the loss function with respect to the input pixels. SmoothGrad aims to make these gradient-based explanations less noisy by adding noise and averaging over these artificially noisy gradients.

3.3 Results

3.3.1 Pulse Frequency Mismatch

First, we investigated whether transfer functions could mitigate the effects of a frequency mismatch. We acquired the training data at a 9-MHz pulse frequency setting and the testing data at a 5-MHz pulse frequency setting.

In Tables 3.4 and 3.5, *"train-time calibration"* and *"test-time calibration"* for the pulse frequency mismatch are compared to a benchmark experiment named "Benchmark" in which there was no mismatch, i.e. the same pulse frequency was used for the training and testing data, to an experiment named "No Calibration" in which we did not use any calibration and to the baseline method Fine-Tuning. For *train-time calibration*, there were two experiment types: "Train-Time Calibration (50%)" in which 50% of the training data is calibrated and the remaining 50% of the training data is uncalibrated, and "Train-Time Calibration (100%)" in which 100% of the training data is calibrated. For the baseline method TL, there were three experiment types: "Fine-Tuning with 2 Frames" in which the training set consists of 2 diverse

Table 3.4: Pulse Frequency Mismatch Sec.3.3.1

Experiment Type	Accuracy (ResNet)	Accuracy (DenseNet)
Train-Time Calibration (50%)	93.31±0.65	90.83±2.68
Train-Time Calibration (100%)	**96.77±1.04**	**95.45±1.34**
Test-Time Calibration	93.25±2.91	93.53±1.52
Fine-Tuning with 2 Frames	94.69±2.63	94.38±2.38
Fine-Tuning with 10 Frames	97.11±1.11	95.64±1.42
Fine-Tuning with 20 Frames	97.41±0.33	95.83±0.49
No Calibration	52.35±0.88	52.33±1.10
Benchmark	99.65±0.51	99.46±0.58

Table 3.5: Pulse Frequency Mismatch Sec.3.3.1

Experiment Type	AUC (ResNet)	AUC(DenseNet)
Train-Time Calibration (50%)	0.987±0.004	0.982±0.005
Train-Time Calibration (100%)	**0.996±0.001**	**0.994±0.001**
Test-Time Calibration	0.989±0.003	0.989±0.002
Fine-Tuning with 2 Frames	0.986±0.014	0.984±0.013
Fine-Tuning with 10 Frames	0.995±0.003	0.989±0.006
Fine-Tuning with 20 Frames	0.996±0.001	0.992±0.003
No Calibration	0.927±0.001	0.938±0.010
Benchmark	0.999±3.4e-5	0.999±4.3e-5

frames, "Fine-Tuning with 10 Frames" in which the training set consists of 10 diverse frames, and "Fine-Tuning with 20 Frames" in which the training set consists of 20 diverse frames. For *train-time calibration* and *test-time calibration*, learning rates were 5e-5, 1e-4 and epoch numbers were 20, 30. For Fine-Tuning experiments, learning rates were 2e-6 and epoch numbers were 20. For "No Calibration", learning rates were 1e-6 and epoch numbers were 20. For "Benchmark", learning rate were 5e-5, 1e-5 and epoch numbers were 25.

3.3.2 Focus Mismatch

In this subsection, we investigated whether transfer functions could mitigate effects of a focus mismatch. We acquired the training data focused at 2 cm

Table 3.6: Focal Mismatch Sec.3.3.2

Experiment Type	Accuracy (ResNet)	Accuracy (DenseNet)
Train-Time Calibration (50%)	94.95±0.96	95.63±0.90
Train-Time Calibration (100%)	94.37±1.53	94.68±1.73
Test-Time Calibration	**96.67±0.46**	**96.34±0.77**
Fine-Tuning with 2 Frames	93.51±3.03	94.07±3.65
Fine-Tuning with 10 Frames	96.66±0.86	95.41±1.15
Fine-Tuning with 20 Frames	98.22±0.35	96.62±0.53
No Calibration	83.44±1.52	85.52±0.73
Benchmark	99.65±0.51	99.46±0.58

Table 3.7: Focal Mismatch Sec.3.3.2

Experiment Type	AUC (ResNet)	AUC(DenseNet)
Train-Time Calibration (50%)	0.991±0.002	0.994±0.001
Train-Time Calibration (100%)	**0.997±0.001**	**0.996±0.001**
Test-Time Calibration	0.996±0.001	0.995±0.001
Fine-Tuning with 2 Frames	0.987±0.011	0.980±0.021
Fine-Tuning with 10 Frames	0.993±0.003	0.988±0.007
Fine-Tuning with 20 Frames	0.997±0.001	0.993±0.003
No Calibration	0.929±0.012	0.939±0.009
Benchmark	0.999±3.4e-5	0.999±4.3e-5

and the test data with dual foci at 1 cm and 3 cm.

Similar to the case with the pulse frequency mismatch, in Tables 3.6 and 3.7, *"train-time calibration"* and *"test-time calibration"* for the focus mismatch are compared to "Benchmark", "No Calibration" and Fine-Tuning experiments. For *train-time calibration* and *test-time calibration*, learning rates were 5e-5, 5e-6, 1e-5 and epoch numbers were 20, 25. For Fine-Tuning experiments, learning rates were 2e-6 and epoch numbers were 20. For "No Calibration", learning rates were 5e-5 and epoch numbers were 20. The "Benchmark" was the same as the previous subsection.

Table 3.8: Output Power Mismatch Sec.3.3.3

Experiment Type	Accuracy (ResNet)	Accuracy (DenseNet)
Train-Time Calibration (50%)	98.24±0.59	**98.39±0.61**
Train-Time Calibration (100%)	97.06±0.54	98.15±0.74
Test-Time Calibration	**98.99±0.35**	98.26±0.21
Fine-Tuning with 2 Frames	96.85±1.05	97.09±2.05
Fine-Tuning with 10 Frames	98.65±0.45	98.73±0.42
Fine-Tuning with 20 Frames	99.26±0.34	98.73±0.38
No Calibration	86.98±1.45	84.41±1.15
Benchmark	99.65±0.51	99.46±0.58

3.3.3 Output Power Mismatch

Finally, we investigated if we were able to mitigate effects of a data mismatch of output power by using transfer functions. We acquired the training data by using 0 dB output power, which represents the maximum output power of the imaging system, and the test data by using -6 dB output power, which represents the output power level that is 6 dB below the maximum.

Similar to the case with the pulse frequency mismatch, in Tables 3.8 and 3.9, *"train-time calibration"* and *"test-time calibration"* for the output power mismatch are compared to "Benchmark", "No Calibration" and Fine-Tuning experiments. For *train-time calibration* and *test-time calibration*, learning rates were 5e-5, 1e-5 and epoch numbers were 20, 25. For Fine-Tuning experiments, learning rates were 2e-6 and epoch numbers were 20. For "No Calibration", learning rates were 5e-5 and epoch numbers were 20. The "Benchmark" was the same as the previous subsections.

3.3.4 Pixel Attribution

In Fig. 3.7, we plotted the pixel attribution for the training domain without any mismatch. In Fig. 3.8, we plotted the pixel attribution in the case of frequency mismatch, and in Fig. 3.9, we plotted the pixel attribution after the calibration approach. Similarly, Figures 3.10 and 3.11 depict the pixel attribution changes for the focus mismatch. Moreover, Figures 3.12 and 3.13 depict the pixel attribution changes for the power mismatch.

Table 3.9: Output Power Mismatch Sec.3.3.3

Experiment Type	AUC (ResNet)	AUC(DenseNet)
Train-Time Calibration (50%)	0.999±4.1e-4	0.999±4.2e-4
Train-Time Calibration (100%)	0.998±0.002	0.999±0.001
Test-Time Calibration	**0.999±3.2e-4**	**0.999±3.3e-4**
Fine-Tuning with 2 Frames	0.995±0.007	0.997±0.002
Fine-Tuning with 10 Frames	0.999±7.3e-4	0.999±4.0e-4
Fine-Tuning with 20 Frames	0.999±5.1e-4	0.999±8.0e-4
No Calibration	0.957±0.010	0.923±0.002
Benchmark	0.999±3.4e-5	0.999±4.3e-5

3.4 Discussion

From Figs. 3.4-3.6, the effect of the data mismatches on the averaged power spectrum are visualized. In Fig. 3.4, the averaged power spectrum of the data from the training setting was shifted to higher frequencies in comparison to the averaged power spectrum of the data from the testing setting. That is expected because we used a 9-MHz pulse frequency in the training setting and a 5-MHz pulse frequency in the test setting. However, the actual shift in the spectrum was relatively narrower than 4 MHz. Overall, while the averaged power spectrum of the data from the testing setting was shifted to lower frequencies, the averaged power spectrum of the data from the training setting was shifted to a higher frequency. The shift led to the setting transfer functions Γ and Γ_{wiener} to be greater than unity with frequency below 5 MHz and less than unity with frequencies above 5 MHz. Γ_{wiener} matched Γ well around the analysis bandwidth due to high SNR and rapidly approached zero above 10 MHz.

In Fig. 3.5, the averaged power spectrum of the data from training setting had higher amplitude than the averaged power spectrum of the data from testing setting because those plots were obtained around 2 cm axially, which corresponds to the focal region of the training setting. As a result, the setting transfer functions Γ and Γ_{wiener} were approximately constant at 0.6 around the analysis bandwidth. Similarly, in Fig. 3.6, the averaged power spectrum of the data from the training setting had higher amplitude than the averaged power spectrum of the data from the testing setting. That is due to using 6 dB higher output power in the data acquisition, which led to the setting

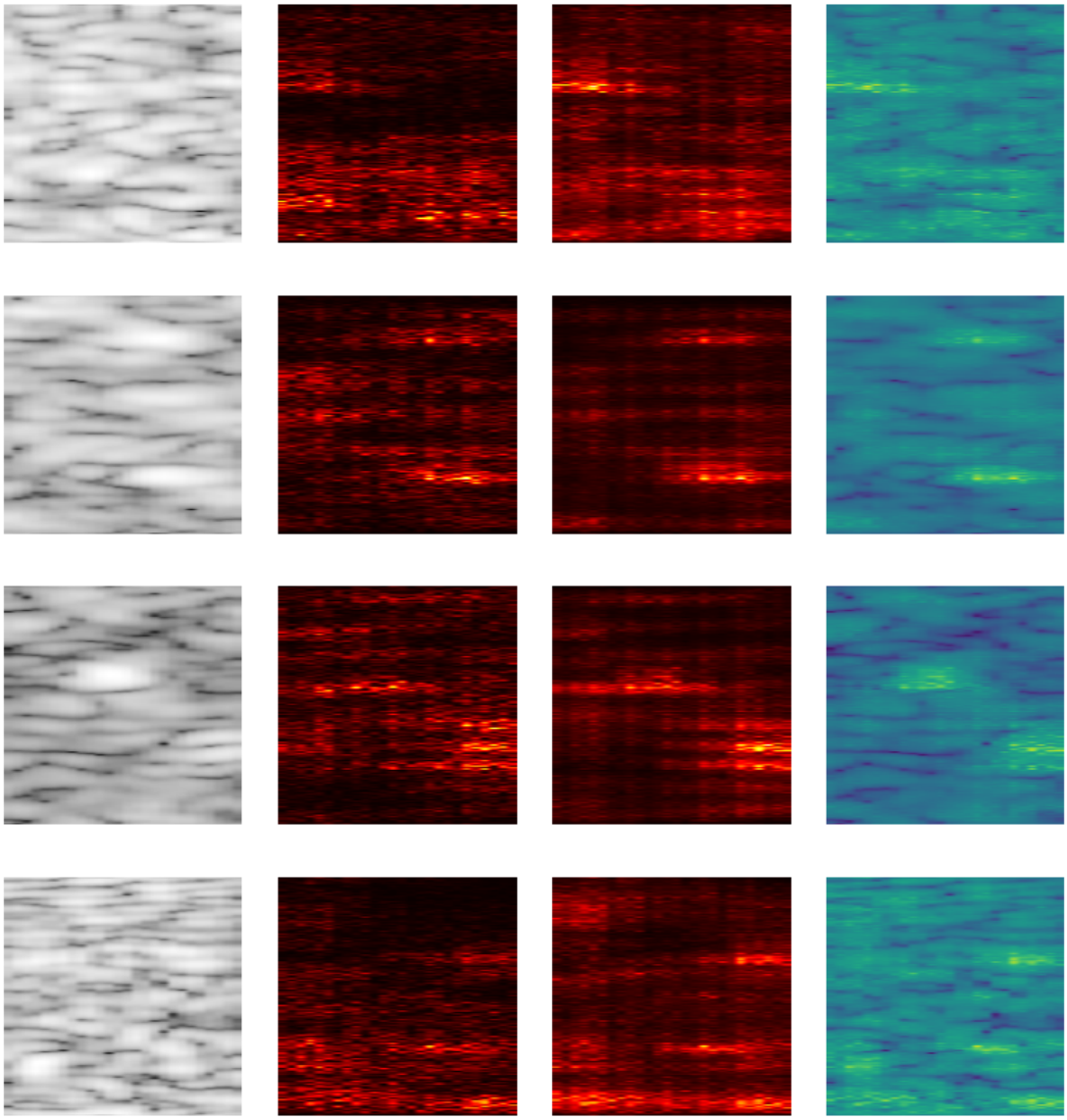


Figure 3.7: Model Interpretation: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from training domain images using the DL network from the same domain. Hence, these results represent the case when there is no mismatch between images and the DL network.

transfer functions Γ and Γ_{wiener} to be relatively constant at 0.5 around the analysis bandwidth. Similar to Fig. 3.4, in Fig. 3.5 and Fig. 3.6, Γ_{wiener} approached zero above 10 MHz due to low SNR.

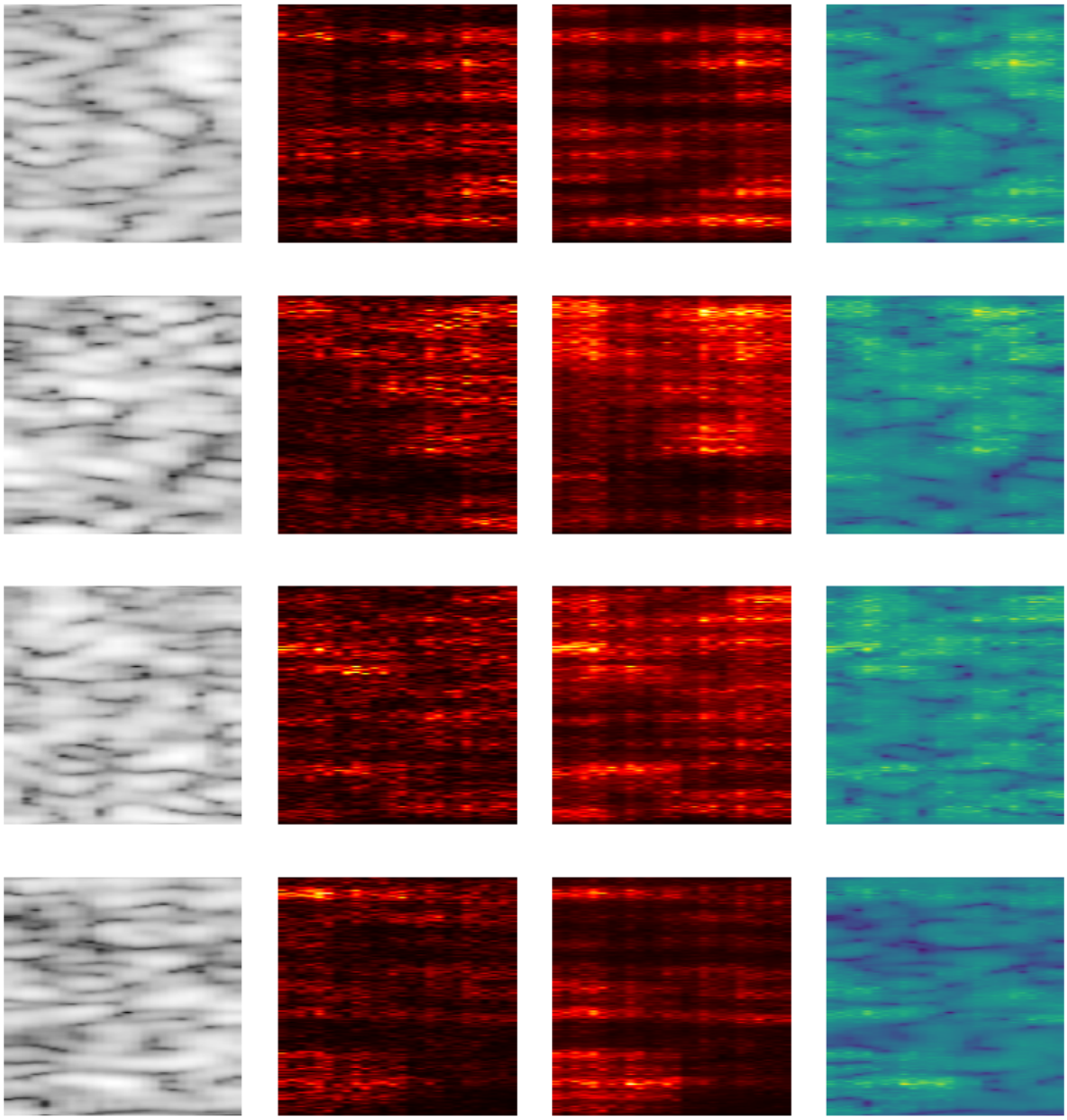


Figure 3.8: Model Interpretation for the Frequency Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images using the DL network from the training domain to investigate the effect domain mismatch in the interpretation result.

In Tables 3.4 and 3.5, we observed that the proposed method mitigated the effects of the given frequency mismatch. ResNet and DenseNet trained without any calibration resulted in mean classification accuracies of 52.35% and

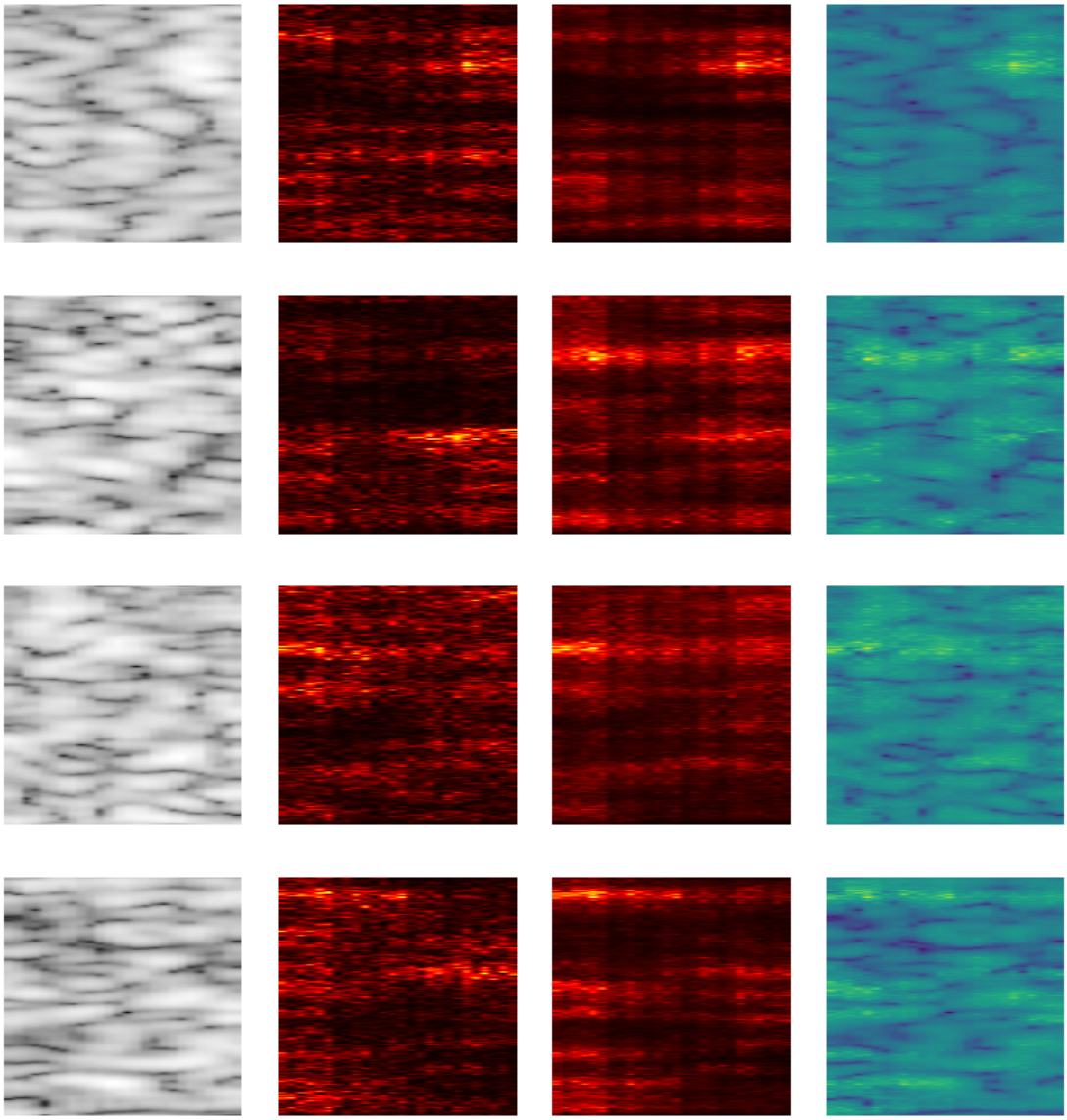


Figure 3.9: Model Interpretation for the Frequency Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images after calibration using the DL network from the training domain. Hence, these results investigate the effect of the proposed calibration on the interpretation results. Identical images from Fig. 3.8 are used in this figure.

52.33%, respectively, which are equivalent to random guess classifiers. When the proposed method was applied, we obtained mean classification accura-

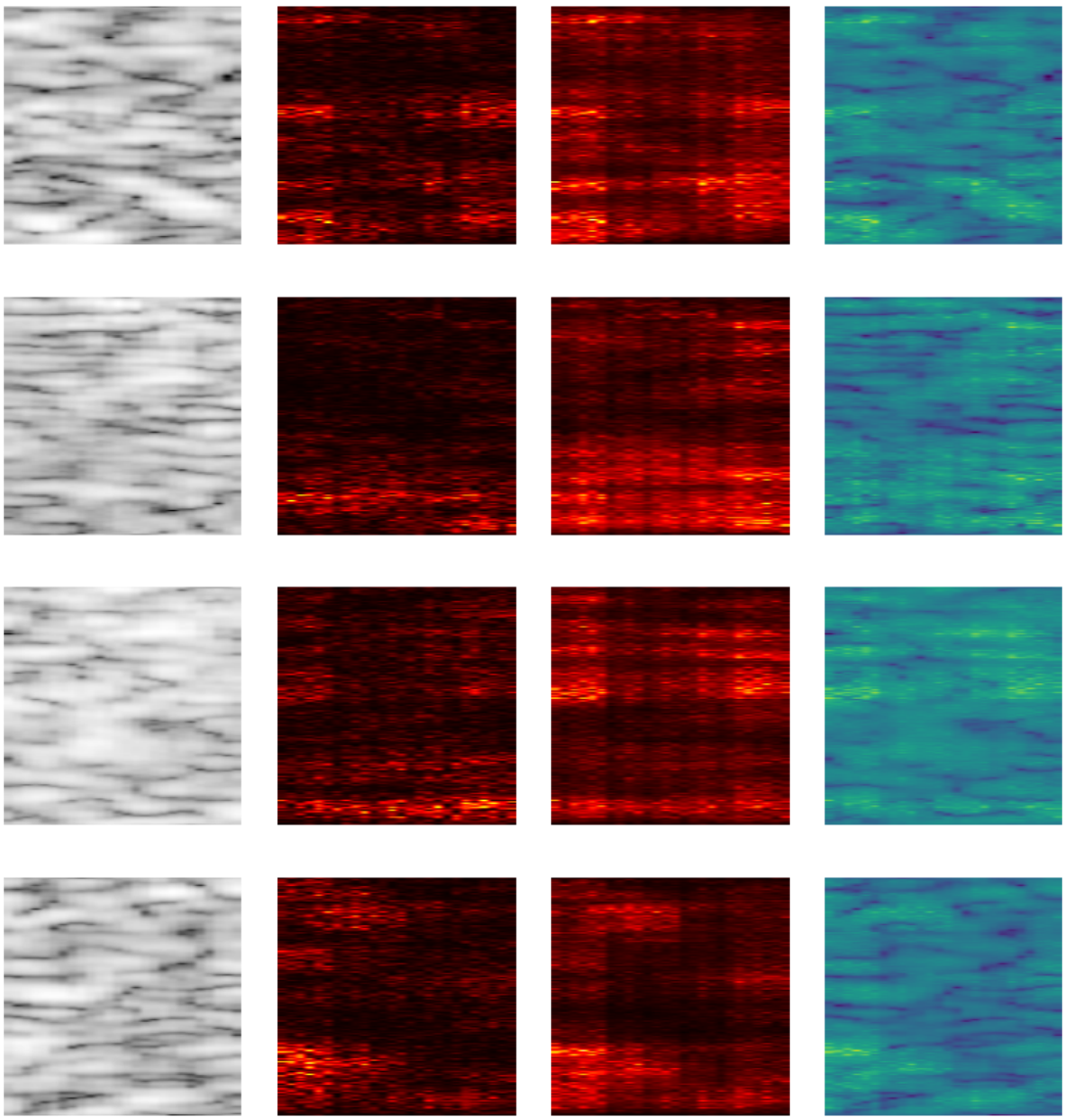


Figure 3.10: Model Interpretation for the Focus Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images using the DL network from the training domain to investigate the effect domain mismatch in the interpretation result.

cies of 96.77% and 95.45%, respectively, which are substantially closer to the benchmark performance. In terms of AUC, ResNet, and DenseNet without any calibration resulted in mean AUC of 0.927 and 0.938, respectively. When

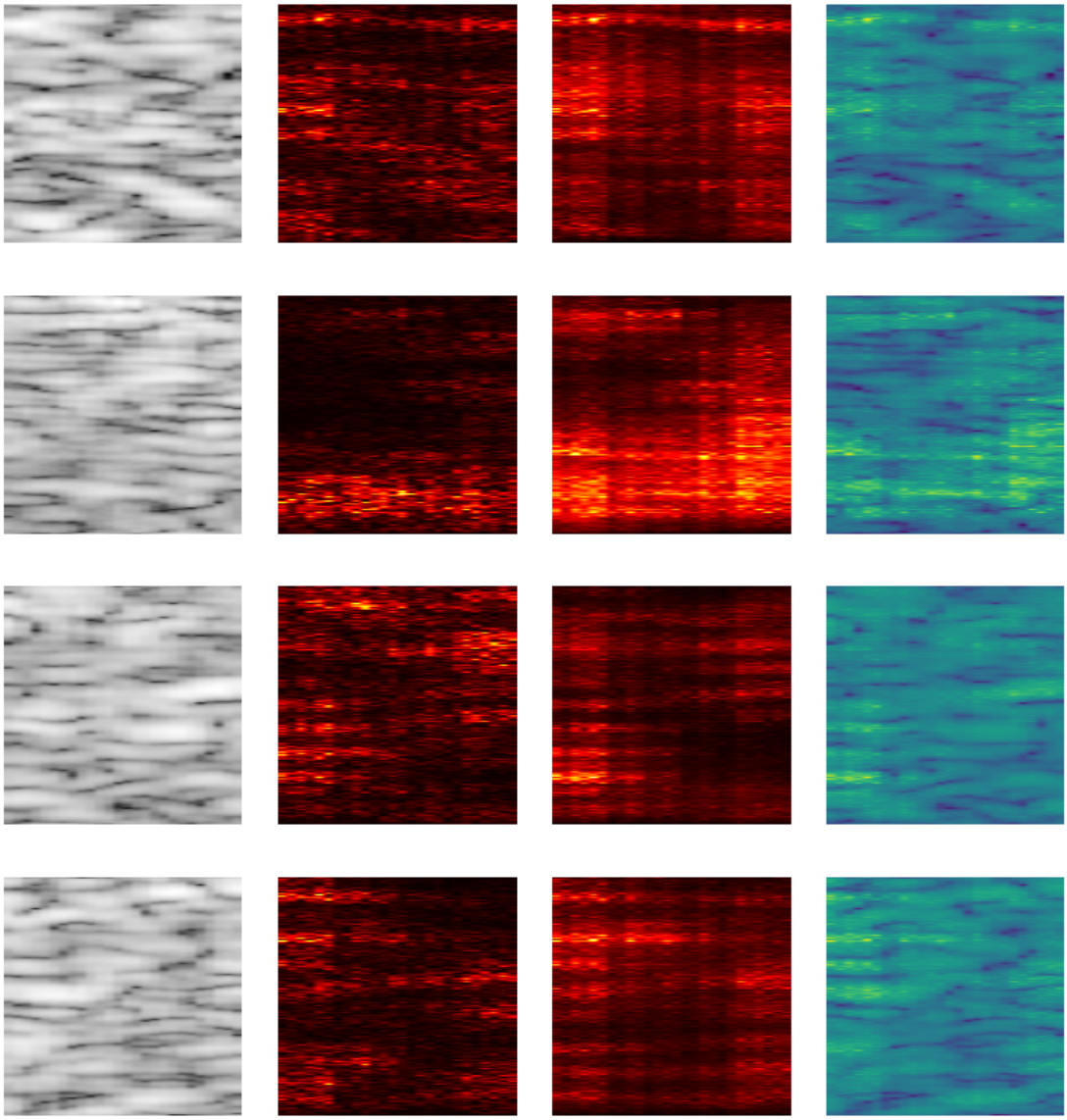


Figure 3.11: Model Interpretation for the Focus Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images after calibration using the DL network from the training domain. Hence, these results investigate the effect of the proposed calibration on the interpretation results. Identical images from Fig. 3.10 are used in this figure.

the proposed method was applied, mean AUCs were 0.996 and 0.994, respectively. In comparison to the baseline method, the proposed method was more

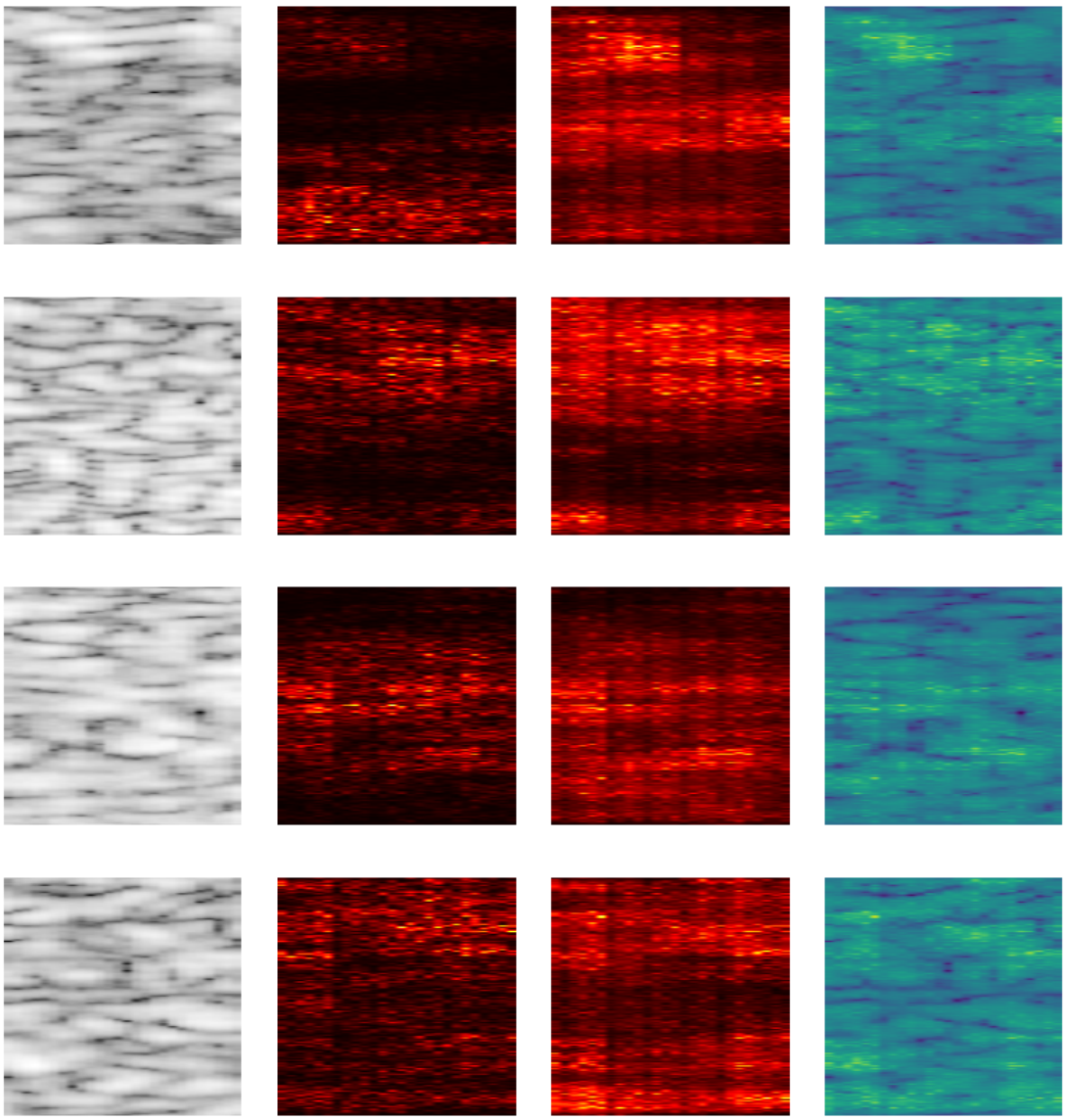


Figure 3.12: Model Interpretation for the Power Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images using the DL network from the training domain to investigate the effect domain mismatch in the interpretation result.

data-efficient as it used a single frame to calibrate the mismatch while Fine-Tuning needed 10 diverse training frames and 10 diverse validation frames to catch up with the proposed method in terms of accuracy. Moreover, Fine-

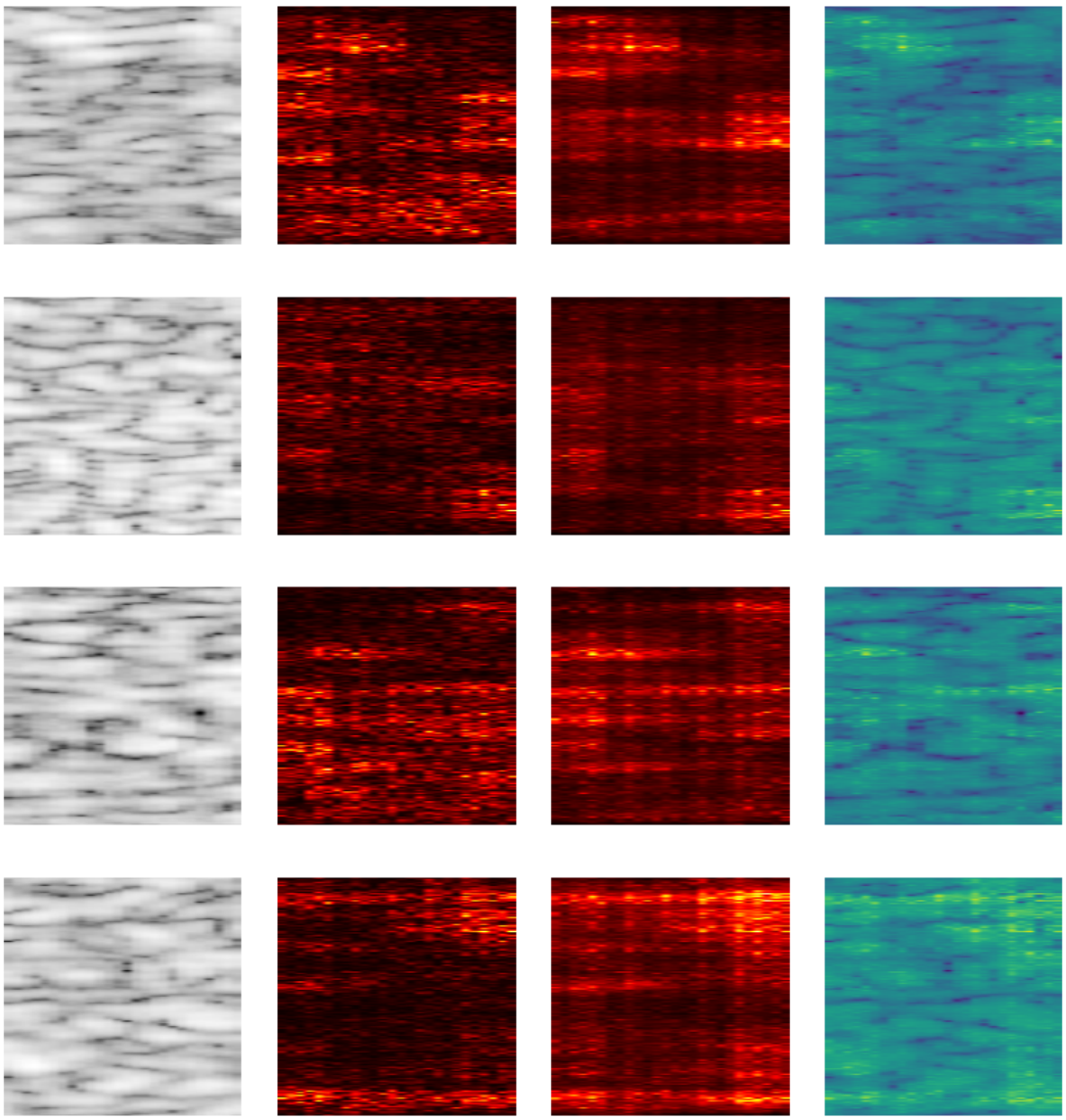


Figure 3.13: Model Interpretation for the Power Mismatch: The leftmost images are B-mode images of the input patch. The second leftmost images show the results of Vanilla Saliency. The second from the rightmost images display the results of SmoothGrad. The rightmost images show overlays between the B-mode and SmoothGrad results. These results are obtained from test domain images after calibration using the DL network from the training domain. Hence, these results investigate the effect of the proposed calibration on the interpretation results. Identical images from Fig. 3.12 are used in this figure.

Tuning needed 20 diverse training frames and 20 diverse validation frames to achieve similar performance as the proposed method, in terms of AUC. Ad-

ditionally, *train-time calibration* performed better than *test-time calibration* for the given mismatch in terms of both accuracy and AUC. Furthermore, when we applied *train-time calibration* for all the training data (calibration 100%) performed better than applying *train-time calibration* for half of the training data (calibration 50%). This indicates that calibrated training data provided all the potential performance increase. Using mismatched (uncalibrated) training data did not provide any additional performance increases for the given experiment.

In Tables 3.6 and 3.7, we observed that the proposed method mitigated the effects of the given focus mismatch. ResNet and DenseNet trained without any calibration resulted in mean classification accuracies of 83.44% and 85.52%, respectively. When the proposed method was applied, we obtained mean classification accuracies of 96.67% and 96.34%, respectively. In terms of AUC, ResNet and DenseNet without any calibration resulted in mean AUC of 0.929 and 0.939, respectively. When the proposed method was applied, mean AUCs were 0.997 and 0.996, respectively. Similar to the frequency mismatch, in comparison to the baseline method, the proposed method is more data-efficient as it used a single frame to calibrate the mismatch. Unlike the frequency mismatch, *test-time calibration* performed better than *train-time calibration* and using mismatched training data provided additional performance increases for the given experiment.

In Table 3.8 and 3.9, we observed that the proposed method mitigated the effects of the given output power mismatch. ResNet and DenseNet trained without any calibration resulted in mean classification accuracies of 86.98% and 84.41%, respectively. When the proposed method was applied, we obtained mean classification accuracies of 98.99% and 98.39%, respectively. In terms of AUC, ResNet and DenseNet without any calibration resulted in mean AUC of 0.957 and 0.923, respectively. When the proposed method was applied, mean AUCs were 0.999 and 0.999, respectively. Similar to the results from the previous mismatches, in comparison to the baseline method, the proposed method was more data-efficient. Similar to the results from the focal mismatch, *test-time calibration* performed slightly better than or close to *train-time calibration*, and using mismatched training data provided additional performance increases for the given experiment.

In Figures 3.7-3.12, we investigated the pixel attributions with and without the calibration approach. These figures indicate that after the calibration

approach, the pixel attributions become similar to Fig. 3.7, which is the case where there is no mismatch. The key observation is that when there is a mismatch, the pixel attribution becomes more distributed over more pixels. When the calibration approach is implemented, the DL network becomes more selective in terms of pixel attribution, and certain pixels only contribute to the decision. Another interesting observation regarding the pixel attributions is the presence of vertical lines where the attribution is lower compared to the adjacent lines. This indicates an interpolation effect in the lateral dimension. Indeed, the RF data inherently has fewer channels, and the SonixOne machine outputs after interpolating it to 256 channels. Apparently, those interpolated lines do not contribute to the DL network decision as much.

Regarding *test-time calibration* vs *train-time calibration*, we observed that they were relatively comparable in terms of AUC. However, in terms of accuracy, *train-time calibration* performed better for the frequency mismatch and *test-time calibration* performed better for the focal and the output power mismatches. One advantage of *test-time calibration* over *train-time calibration* is its simplicity. The *train-time calibration* requires the training data to be converted, added to the data, and the model retrained. With the *test-time calibration*, there is no need for any retraining or fine tuning of the network when a new test setting is being used. Therefore, the *test-time calibration* approach is more suitable for real-time clinical applications. On the other hand, *train-time calibration* has some algorithmic advantages, such as choosing hyper-parameters being more convenient because the training error is directly related to validation error.

In the three types of mismatches, we consistently observed a small decrease in AUC while the classification accuracy dropped significantly. For example, in the frequency mismatch case, the classification accuracy dropped from 99% to 52%, while the AUC dropped from 0.99 to 0.93. These observations indicate that the separability between the two classes remains high, even when the optimal threshold changes significantly. Therefore, this suggests that the DL network could be calibrated through AUC analysis to identify the new optimal threshold. However, this process requires data to be acquired from actual samples under testing conditions for training similar to Fine-Tuning. Overall, the proposed method increased accuracy significantly and improved AUC, which verifies its calibration capabilities without the need

for data acquired from actual samples under testing conditions.

Among the investigated acquisition-related data mismatches, the frequency mismatch led to the largest drop in accuracy for the no calibration case compared to the focal and output power mismatches. Specifically, the mean classification accuracy dropped from 99% to 52% for the frequency mismatch, while it dropped from 99% to 84% for the focal and output power mismatches. These observations suggest that the frequency mismatch caused the most disturbance to the optimal threshold.

Using uncalibrated data in the *train-time calibration* improved accuracy for the focal and output power mismatch but not for the frequency mismatch. Therefore, depending on the type of applications and data mismatches, using uncalibrated data in training could potentially provide richer training data and better generalizability. The proposed method resulted in similar performance improvements for both ResNet and DenseNet, verifying its validity for various network structures. Additionally, the proposed method was more data-efficient than the baseline method, as it only required a single calibration view, while the baseline method required 10 diverse frames in the training and 10 diverse frames in the validation to perform similarly in terms of accuracy, and required 20 diverse frames in the training and 20 diverse frames in the validation to perform similarly, in terms of AUC.

Moreover, the proposed method was more practical than the baseline method in the clinical workflow. As it does not require any diverse calibration views, the calibration set can be acquired through automation by stable acquisition. Another point related to the clinical workflow is the need for actual samples. The proposed method does not require actual samples to calibrate the data mismatches because the calibration phantom is sample-irrelevant. However, transfer learning requires real sample data to fine-tune the models, and it demands more data as the number of classes increases.

One open question would be related to how to select the calibration phantom. When non-linearities in the system and imaging substrate were negligible, the system response of an ultrasound system and hence the setting transfer function, Γ, should be the same irrespective of the imaging substrate. Therefore, we could use any phantom with uniform scattering properties as the calibration phantom. However, due to random spatial variation noise from the subresolution scatterers, there could be some fluctuation in the setting transfer function Γ as it can be observed from Figs. 3.4-3.6. If one could

average multiple views, variation in the power spectra would decline. However, it would also increase the complexity of the approach. In the proposed approach, we acquired multiple frames of the same view after stabilizing the transducer for the calibration data. This way, the acquisition of the calibration data could be automated easily for all imaging settings without any human intervention. The need of multiple views in the calibration data would make it more complicated and expensive. As a next step, we extended setting transfer function to investigate the use of a more diverse probe model and imaging machines to introduce machine-to-machine transfer function. As long as the frequency spectrum of training and testing overlap, the proposed method is expected to work well.

CHAPTER 4

MACHINE-TO-MACHINE TRANSFER FUNCTION IN DEEP LEARNING-BASED QUANTITATIVE ULTRASOUND

4.1 Motivation

A Transfer Function approach was demonstrated to mitigate data mismatches at the acquisition level for a single ultrasound scanner in deep learning (DL) based quantitative ultrasound (QUS) in the previous chapter. As a natural progression, we further investigate the transfer function approach and introduce a Machine-to-Machine (M2M) Transfer Function, which possesses the ability to mitigate data mismatches at a machine level, depicted in Fig. 4.1. This ability opens the door to unprecedented opportunities for reducing DL model development costs, enabling the combination of data from multiple sources or scanners, or facilitating the transfer of DL models between machines.

As described earlier, the adoption of DL-powered biomedical imaging has been slow due to data scarcity and data mismatch issues. We developed *Zone Training* to address data scarcity aspect, and proposed *Setting Transfer Function* to address acquisition-related data mismatches. The transfer function approach significantly improved mean classification accuracies for pulse frequency, output power, and focal region mismatches within the same imaging machine, increasing them from 52%, 84%, and 85% to 96%, 96%, and 98%, respectively. Therefore, the transfer function approach has emerged as an economical way to generalize a DL model for tissue characterization in cases where scanner settings cannot be fixed, thus improving the robustness of DL-based algorithms. Building on the transfer function idea, we now propose the *M2M Transfer Function*.

To further motivate the *M2M Transfer Function*, we investigated related approaches. For example, data augmentation is a crucial tool for minimizing data mismatch by approximating the distribution shift between testing and

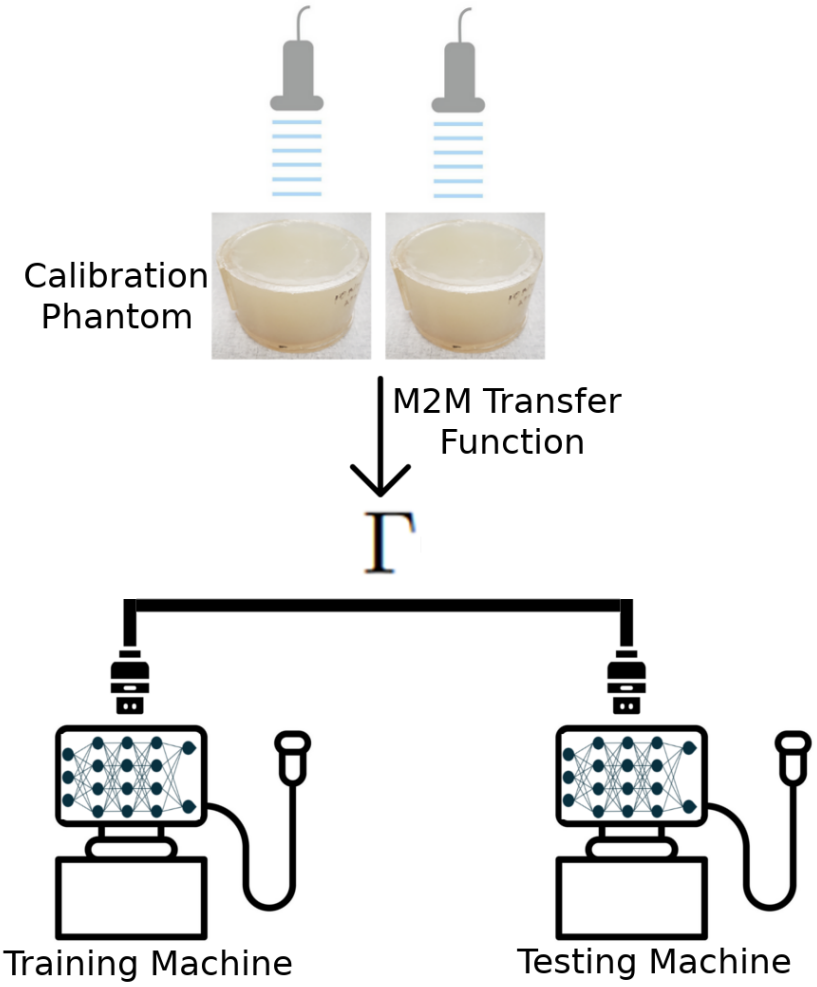


Figure 4.1: M2M Transfer Function

training data [96–99]. The performance of these approaches depends on how well the approximation mitigates the distribution shift. Other approaches attempt to learn a generative model between testing and training domains [100–103]. On the other hand, domain generalization approaches aim to recover feature representations that are independent of domains [104–106]. Their performance relies on the invariance of the learned features. Additionally, BN-Adapt [107] modifies batch normalization (BN) layers adaptively using test domain data. Moreover, pretraining is another significant concept [108–110]. Pretraining on a larger dataset could provide robust representations for downstream tasks. The issue of data mismatch has gained increased attention, as discussed earlier, in recent literature focusing on DL-based QUS [103, 111–115]. In contrast to these methodologies, the transfer

function approach developed in the previous chapter does not require real sample data from the testing domain to be used for training. Instead, it relies on a calibration phantom that can be tailored to the specific characteristics of the real sample at hand. The transfer function separates the calibration process from data acquisition with patients, making the calibration process patient-free from a clinical perspective. Therefore, the transfer function approach provides a practical method to shift the domain of the training dataset to the testing domain, or vice versa, in contrast to these methodologies.

As the transfer function holds the potential for practical implementation within clinical settings, given that it does not necessarily require real samples from the testing domain, it is essential to further validate and identify its strengths and weaknesses under more substantial mismatches. In this chapter, the application of the transfer function approach was extended to address data mismatches between different imaging machines. By doing so, the transfer function approach would increase its utility in multiple ways. First, being able to transfer between machine domains can lower the cost of DL-based QUS approaches. Specifically, data from different machines can be combined to develop more robust and accurate DL-based models. This has the potential to provide a simple and efficient means of utilizing existing data from different machines and sources, which helps address the high cost associated with labelled data collection.

4.2 Methods

4.2.1 Calibration

The derivation of the "M2M Transfer Function" remains identical to that of the previous chapter. However, in this context, the transfer function is calculated between ultrasound machines rather than between different imaging settings within the same machine.

$$\frac{W_{test}(x,f)}{W_{train}(x,f)} = \frac{S_{\phi_{test}}(x,f)}{S_{\phi_{train}}(x,f)}, \tag{4.1}$$

$$= \Gamma_{train\rightarrow test}(x,f). \tag{4.2}$$

where ϕ_{train} and ϕ_{test} represents the training machine and testing machine, respectively. Then, the M2M transfer function, denoted by $\Gamma_{train\rightarrow test}$, is capable of transferring between training and testing machines when the bandwidth is overlapping via,

$$I_{train\rightarrow test}(x,f) = \Gamma_{train\rightarrow test}(x,f) I_{train}(x,f), \tag{4.3}$$

in train-time, depicted in Algorithm 1. Or,

$$I_{test\rightarrow train}(x,f) = \Gamma_{test\rightarrow train}(x,f) I_{test}(x,f). \tag{4.4}$$

in test-time, depicted in Algorithm 2.

In this chapter, we investigated two methods for calculating the M2M transfer function. In the first method, stable acquisition was implemented. This involved fixing the transducer using holders and clamps. Following the acquisition of calibration data from one machine, the probe seamlessly transitioned to the other machine without altering its position on the calibration phantom by simply moving the connector from one machine to the other. This calibration procedure assumes that the same transducer is being used even though the machine is different. In the second method, free-hand acquisition was used, involving free-hand motion to record a video of 1000 ultrasound frames from the calibration phantom using both testing and training machines. Free-hand acquisition for calibration data is essential when using different transducers between training and testing conditions. Additionally, two different types of calibration phantoms, each with uniform scattering properties, were utilized to investigate the effect of calibration phantom selection on calibration performance.

Implementation details of the M2M transfer function are identical to the previous chapter. Wiener implementation is used with SNR calculation follows the same recipe. Filters obtained and implemented depth-wise. The calibration techniques are depicted in Algorithm 1 and Algorithm 2, in the

previous chapter.

The only difference in the "M2M Transfer Function" compared to the "Setting Transfer Function" is the operation mode of BN layers. It has been observed in the literature that BN layers play a critical role in addressing data mismatch issues, leading to techniques that involve modifications in BN layers. Inspired by these methods, during both training and evaluation for the "M2M Transfer Function," BN layers were modified to operate with batch statistics. Specifically, in PyTorch, the running statistics of BN layers were set to None. Under this condition, BN layers have learnable affine parameters but utilize batch statistics instead of updating running statistics. Hence, in the experiments with the "M2M Transfer Function," affine parameters in BN layers were learned utilizing batch statistics.

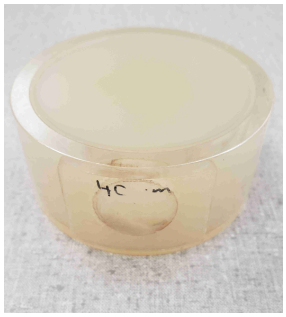
4.2.2 Phantoms

The experiments utilized two distinct tissue-mimicking phantoms as classification phantoms, shown in Fig. 4.2. Additionally, two distinct calibration phantoms were used to obtain the M2M transfer function, shown in Fig. 4.3 and summarized in Table 4.1. In Table 4.1, speed of sound (S0S) is given in terms of $m \times s^{-1}$, attenuation (Atten) is given in terms of $\mathrm{dB} \times \mathrm{cm}^{-1} \times \mathrm{MHz}^{-1}$, bead diameter (BD) is given in terms of μm and bead concentration (BC) is given in terms of $\mathrm{g} \times \mathrm{L}^{-1}$. The beads were glass spheres with selected mean sizes and distributions to act as scatterers in the phantoms.

Classification Phantom 1 mimics the characteristics of the human liver [130] and the construction details were given in [131]. The attenuation coefficient slope for Classification Phantom 1 was measured as 0.4 $\mathrm{dB} \times \mathrm{cm}^{-1} \times \mathrm{MHz}^{-1}$. It exhibited macroscopic uniformity. The speckle pattern in Classification Phantom 1 stemmed from the random distribution of microscopic glass bead scatterers, ranging in diameter from 82.5 $\pm$ 7.5 μm. Its speed of sound was 1540 $m \times s^{-1}$.

Classification Phantom 2 was characterized as a low-attenuation phantom [132] and the construction details were given in [133]. The same weakly-scattering agar, serving as the background material, was utilized in Classification Phantom 2 but included glass-bead scatterers of varying sizes, ranging from 41 $\pm$ 2 μm in diameter. Its speed of sound was 1539 $m \times s^{-1}$. The

attenuation coefficient slope was measured as 0.1 dB×cm^{-1}×MHz $^{-1}$.

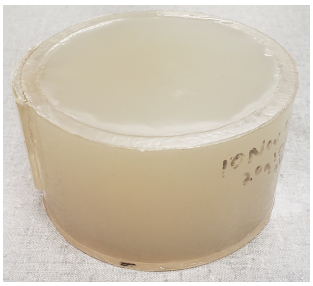



Classification Phantom 1 Classification Phantom 2

Figure 4.2: Visuals of The Classification Phantoms

The Calibration Phantom 1 was a commercial QUS reference phantom (part no. 14090502, serial no. 221447541) from CIRS, Inc., Norfolk, VA. It had an attenuation coefficient slope of 0.74 dB×cm^{-1}×MHz $^{-1}$. Its speed of sound was 1545 $m \times s^{-1}$.

The Calibration Phantom 2 was characterized as a low-attenuation phantom [132]. We used the approach by Anderson et al. [132] who utilized the arrival time difference [146], the insertion loss techniques [130], and the narrowband through-transmission technique [133] to characterize speed of sound and attenuation. It was constructed with a 2% agar background having weakly scattering properties. This phantom included glass beads with diameters measuring 160 ± 60 μm. The distribution of glass beads, occurring spatially randomly within the phantom's volume, was at a concentration of 20 g/L. The attenuation coefficient slope for Classification Phantom 2 measured as 0.6 dB×cm^{-1}×MHz $^{-1}$. Its speed of sound was 1535 $m \times s^{-1}$.

Calibration Phantom 1 Calibration Phantom 2

Figure 4.3: Visuals of The Calibration Phantoms

Table 4.1: Phantoms

Phantoms	SOS	Atten	BD	BC
Classification Phantom 1	1540	0.4	82.5±7.5	3.8
Classification Phantom 2	1539	0.1	41±2	2.23
Calibration Phantom 1	1545	0.7	Unknown	Unknown
Calibration Phantom 2	1535	0.6	160±60	2.23

4.2.3 Ultrasound Machines

The phantoms were scanned using both a SonixOne system and a Verasonics Vantage 128. An Ultrasonix L9-4 transducer and a Verasonics L11-5 transducer were utilized throughout the experiments. The datasets are summarized in Table 4.2. The SonixOne system captured post-beamformed radio-frequency (RF) data with a sampling rate of 40 MHz. Under the hood, the raw radio-frequency (RF) channel data had a 16-bit bit depth, and the beamforming operation was performed with a 64-bit bit depth in the SonixOne machine. In contrast, the Verasonics system acquired raw channel data and did not return post beamformed data. The Verasonics data was sampled at a rate of 50 MHz with a 16-bit bit depth. Subsequently, delay and sum beamforming were implemented on the Verasonics data using a 64-bit depth, similar to the SonixOne system. Following this, a multirate FIR filter was designed with an interpolation factor of 4 and a decimation factor of 5 to convert the sampling rate to 40 MHz. Beamforming and sampling rate conversion were implemented using MATLAB (version: R2023a) functions. Specifically, the 'designMultirateFIR' function was used, which computes the filter coefficients based on the interpolation and decimation factors, while the 'dsp.FIRRateConverter' function was used to implement a combined anti-aliasing FIR filter using these filter coefficients, the decimation factor, and the interpolation factor. After these preprocessing steps, post-beamformed RF data at a matching sampling frequency of 40 MHz was obtained from both machines for DL operations. Matching the sampling rate between systems was critical to being able to implement the M2M transfer function.

As training data, the SonixOne data, acquired with L9-4 transducer, was utilized during the experiments, positioning the SonixOne as the "training machine" where the model development occurred. On the other hand, the Verasonics data, acquired with both L9-4 and L11-5 transducers, was utilized

Table 4.2: Imaging Conditions

No	Machine	Transducer	Type	Frequency	Sampling	Amplitude
1	SonixOne	L9-4	training	9MHz	50MHz	0dB
2	Verasonics	L9-4	testing	5MHz	40MHz	45.2V
3	Verasonics	L11-5	testing	5MHz	40MHz	45.2V

as testing data during the experiments, positioning it as the "testing machine" where the machine data is assumed to be unavailable during model development. Testing machine data was only used to measure calibration success during inference time. For training data and testing data, free-hand data acquisition was utilized with Classification Phantom 1 and Classification Phantom 2, i.e., the transducer was moved across the phantom surface by hand. During this acquisition, by recording a video of 1000 frames, we captured a large amount of ultrasound data for each phantom.

For calibration data, both the SonixOne and Verasonics machines were utilized in two scanning procedures using Calibration Phantom 1 and Calibration Phantom 2. In the first procedure, similar to the training and testing data, free-hand acquisition was utilized, which provided 1000 independent frames from each calibration phantom. The second procedure, termed stable acquisition, involved securing the transducer using a bar clamp holder. Subsequently, ten identical frames were captured using both the SonixOne and Verasonics machines from precisely the same position on the calibration phantoms. These procedures facilitated the acquisition of calibration data necessary for computing the M2M transfer function.

As imaging settings, line by line acquisition with 2 cm axial focal point at transmission with dynamic focusing during reception was used for both machines. In the SonixOne, the center pulse frequency was set at 9 MHz and its output power level was set at 0 dB. In the Verasonics, the center pulse frequency was set at 5 MHz and its output power level was set at 45.2 Voltage. These settings were configured to evaluate the proposed method under combined hardware and acquisition-related mismatches. Note that the free-hand calibration was required for the change in the probe, i.e., going from the L11-5 to the L9-4.

4.2.4 Data Preparation

Patch extraction has been implemented as described in earlier chapters. Therefore, "patch-wise" data of z-score-normalized RF signals were used at neural networks' input.

From the training machine, 2000 ultrasound frames were acquired, with 1000 frames from each classification phantom, to be used in training, resulting in 162,000 patches. From the testing machine, 1000 ultrasound frames were acquired, with 500 frames from each classification phantom, to be used in testing, leading to 81,000 patches. Regarding calibration data, through stable acquisition, 10 frames from a fixed point were acquired for each machine, and through free-hand acquisition, 1000 frames were acquired for each machine.

In the experiment setup, a balanced configuration was investigated in terms of both class identify and depth location. During partitioning the data into training and validation sets, data points were uniformly and randomly grouped into two sets, with the training set representing 4/5 of the total patches. The setup ensured equal representation of patches from different depths and different classes in training, validation and calibration.

4.2.5 Training

The DL algorithms were trained utilizing a workstation equipped with four NVIDIA RTX A4000. Each experiment was conducted using all four RTX A4000s in parallel. The PyTorch library [134] was utilized for all experiments.

In all experiments, we utilized the Adam algorithm [135] as the optimizer. Hyper-parameters, including epoch numbers and learning rates, were determined aiming for "asymptotic test accuracy". The batch size was selected as 2048 maximize memory utilization. During training, a standard method for data augmentation involved applying a horizontal flip with a 50% probability by default. As training loss, cross-entropy loss was utilized. In the training phase, the data patches were split into training and validation sets using a uniform random approach, with a ratio of 4:1.

Each experiment, i.e., the training, was repeated 10 times. Next, the average of the classification accuracies, the average area under the receiver operator characteristic (ROC) curve (AUC) and their respective standard deviations were computed using the test sets. The results were obtained

patch-wise. The variance in the results was caused by the random initialization of network parameters at each repetition. In the code, random seed was included, ensuring that the results were reproducible.

4.2.6 Network Structure

We employed two established CNN architectures, as in the previous chapter, in this chapter: ResNet-50 [96] and DenseNet-201 [142]. We made minor adjustments to the CNN architectures to customize their input-output relationship to suit our specific problem. The first convolutional layers, which originally took three input feature channels, were replaced with a single input-channel convolution layer. Additionally, the last layer, a fully connected layer, was also modified to output a single probability corresponding to two classes. For network parameter initialization, pretrained weights were used, except for the first convolutional layer and the last fully connected layer, which were initialized using the default method in PyTorch. During training, all the parameters were unfrozen and fine-tuned through backpropagation.

4.2.7 Fine-Tuning

Pretraining is a significant concept in DL. Pretraining on a larger dataset could provide robust representations for downstream tasks in scenarios with low data availability. In this case, pretraining on the training machine data was followed by fine-tuning with a smaller dataset from the test machine, offering an efficient approach to overcoming machine-level data mismatches. Therefore, as a baseline method, the proposed approach was compared with the fine-tuning approach. In the fine-tuning approach, after the pretraining stage, 100 ultrasound frames from the testing machine were utilized to fine-tune the pretrained model. It is important to note that this approach requires diverse frames from the testing conditions, and the data must be from the classification phantoms. Moreover, during the fine-tuning stage, for ResNet, the last three Conv2D layers and the linear layer were unfrozen. For DenseNet, the last four Conv2D layers and the linear layers were unfrozen.

4.2.8 AUC Analysis

Another interesting baseline involves conducting AUC analysis on the pretrained model. In this approach, instead of fine-tuning the pretrained model, AUC analysis was performed with a small dataset from the testing machine to identify the optimal threshold for the last logit (after the sigmoid) outputted by the DL network. It is assumed that under the mismatch, the pretrained model remains robust in terms of separability, indicated by a high AUC score in testing conditions. The main degradation is expected to arise from shifting the threshold to determine the class identity in the last logit. Therefore, the proposed method was compared with the AUC analysis. It is important to note once again that this approach has similar limitations to fine-tuning, requiring diverse frames from the testing conditions, and the data must be from the classification phantoms. In the AUC analysis approach, 100 ultrasound frames were utilized.

4.2.9 BN Freezing

It has been demonstrated that BN layers play a critical role in preserving domain information [147]. Subsequently, in one line of domain adaptation techniques, BN layers were modified and adapted to learn domain invariant features. In ultrasound imaging, Tehrani et al. demonstrated that freezing BN layers during training improves domain adaptation [113]. Therefore, as a third baseline method, the proposed calibration approach was compared to BN freezing. However, it is important to note that even though there is no direct usage of any information related to the test data during training, at evaluation time, test statistics were needed to properly apply z-score normalization. Hence, the only limitation of this approach, in comparison to the other two baseline methods, is the assumption that z-score normalization statistics at test time are given.

4.2.10 No Calibration

The proposed method was also compared to the scenario where no calibration or adaptation method was implemented. In the results, three experiments were conducted under the no-calibration case: 1) No calibration with train

statistics, 2) No calibration with calibrated statistics, and 3) No calibration with test statistics. Different statistics represent different z-score normalization statistics at the evaluation stage. "No calibration with train stats" utilizes $\mathbb{E}X_{train}$ and σX_{train}. This experiment represents the worst case, where no mismatch mitigation is applied. "No calibration with calibrated stats" utilizes $\mathbb{E}X_{train \to test}$ and $\sigma X_{train \to test}$. This experiment represents the use of the proposed calibration method in its simplest form by altering z-score normalization statistics only. "No calibration with test stats" utilizes $\mathbb{E}X_{test}$ and σX_{test}. This experiment represents the perfect adaptation of the normalization layer only, relying on true information of test data statistics which refers to an idealized experiment, serving as a reference for comparison.

4.2.11 Statistical Test

As a statistical test, the Wilcoxon signed-rank test was utilized using the scipy.stats.wilcoxon function from the SciPy library. In the results, p-values for each pair are presented in tabular format. In the tables, the right upper side was reserved for ResNet, and the lower left side was reserved for DenseNet. Moreover, $*$ represents not applicable scenarios and < 0.01 represents statistically different pairs at threshold $p = 0.01$.

4.3 Results

4.3.1 Transfer Function vs Baseline Methods

We compare train-time calibration, test-time calibration, no calibration experiments and the baseline methods in Tables 4.3 and 4.3. Statistical tests are given in Tables 4.11 and 4.12. In these experiments, stable acquisition with Calibration Phantom 1 was utilized to obtain the M2M transfer function. We set the learning rate to $1e-5$ and the number of epochs to 50 for both train-time and test-time calibration. For the no calibration experiments, we set the learning rate to $5e-6$, and we ran the training for 25 epochs to facilitate early stopping. For the fine-tuning, after the pretraining stage, which was the no calibration case, we set the learning rate to $1e-5$ and the number of epochs to 50. For the AUC analysis, there was no extra training over the

Table 4.3: Transfer Function vs Baseline Methods in terms of Classification Accuracy

No	*Experiment*	*ResNet-Acc.*	*DenseNet-Acc.*
1	Train-Time Calibration	93.69±0.70	94.36±0.97
2	Test-Time Calibration	**98.73±0.30**	**98.96±0.32**
3	No Calib. with Train Stats	50.01±0.01	47.19±5.94
4	No Calib. with Calib. Stats	70.59±4.73	68.40±2.86
5	No Calib. with Test Stats	75.11±4.67	72.65±3.23
6	Fine-Tuning	86.32±1.12	86.82±1.06
7	AUC Analysis	87.94±0.90	85.96±2.24
8	BN Freezing	87.97±1.41	89.46±0.72

Table 4.4: Transfer Function vs Baseline Methods in terms of AUC Scores

No	*Experiment*	*ResNet-AUC*	*DenseNet-AUC*
1	Train-Time Calibration	0.983±0.003	0.984±0.004
2	Test-Time Calibration	**0.999±4e-4**	**0.999±4e-4**
3	No Calib. with Train Stats	0.405±0.073	0.488±0.157
4	No Calib. with Calib. Stats	0.936±0.021	0.919±0.017
5	No Calib. with Test Stats	0.949±0.011	0.936±0.139
6	Fine-Tuning	0.946±0.009	0.945±0.008
7	AUC Analysis	0.950±0.010	0.937±0.014
8	BN Freezing	0.951±0.011	0.963±0.005

no calibration case. Moreover, for the BN freezing, we set the learning rate to $1e-5$ and the number of epochs to 50. The results reveal a significant gain achieved by the proposed method. Furthermore, the proposed method offers additional advantages in comparison to the baseline methods, such as not requiring any data from classification samples and needing only a single stable view from a calibration phantom to calculate the transfer function.

4.3.2 Different Calibration Phantoms

We investigated the effects of using different calibration phantoms, Calibration Phantom 1 and Calibration Phantom 2, on the success of calibration, as shown in Tables 4.5 and 4.6. The results of statistical tests are provided in Tables 4.13 and 4.14. For train-time calibration and test-time calibra-

Table 4.5: Different Calibration Phantoms in terms of Classification Accuracy

No	*Experiment*	*ResNet-Acc.*	*DenseNet-Acc.*
1	Train-Time with Calib. Ph. 1	93.69±0.70	94.36±0.97
2	Train-Time with Calib. Ph. 2	93.36±0.60	93.85±0.98
3	Test-Time with Calib. Ph. 1	**98.73±0.30**	**98.96±0.32**
4	Test-Time with Calib. Ph. 2	97.36±0.43	97.62±0.38

Table 4.6: Different Calibration Phantoms in terms of AUC Scores

No	*Experiment*	*ResNet-AUC*	*DenseNet-AUC*
1	Train-Time with Calib. Ph. 1	0.983±0.003	0.984±0.004
2	Train-Time with Calib. Ph. 2	0.981±0.003	0.982±0.004
3	Test-Time with Calib. Ph. 1	**0.999±4e-4**	**0.999±4e-4**
4	Test-Time with Calib. Ph. 2	0.997±0.001	0.997±0.001

tion, we set the learning rate to $1e-5$, and ran the training for 50 epochs. Stable acquisition was utilized in these results because we only used a single transducer, the L9-4, for both machines. Utilizing different calibration phantoms led to statistically different behavior for both train-time and test-time calibration.

4.3.3 Stable vs Hand-Free Calibration

We investigated the effects of using different acquisitions for the calibration data, stable and free-hand, on the success of calibration, as shown in Tables 4.7 and 4.8. Statistical tests are provided in Tables 4.15 and 4.16. For train-time calibration and test-time calibration, we set the learning rate to $1e-5$, and ran the training for 50 epochs. Calibration Phantom 1 was utilized in these results. Utilizing free-hand calibration did not result in performance improvement over stable calibration.

4.3.4 Different Transducers

We investigated the effects of using different transducers for the test machine on the success of calibration, as shown in Tables 4.9 and 4.10. Statistical tests are provided in Tables 4.17 and 4.18. For the training machine, the L9-4 was

Table 4.7: Stable vs Hand-Free Calibration in terms of Classification Accuracy

No	*Experiment*	*ResNet-Acc.*	*DenseNet-Acc.*
1	Train-Time with Stable	93.69±0.70	94.36±0.97
2	Train-Time with Hand-Free	93.57±0.70	94.27±0.99
3	Test-Time with Stable	**98.73±0.30**	**98.96±0.32**
4	Test-Time with Hand-Free	98.66±0.33	98.90±0.35

Table 4.8: Stable vs Hand-Free Calibration in terms of AUC Scores

No	*Experiment*	*ResNet-AUC*	*DenseNet-AUC*
1	Train-Time with Stable	0.983±0.003	0.984±0.004
2	Train-Time with Hand-Free	0.982±0.004	0.984±0.004
3	Test-Time with Stable	**0.999±4e-4**	**0.999±4e-4**
4	Test-Time with Hand-Free	0.999±6e-4	0.999±5e-4

utilized while the L11-5 was utilized for the testing machine. For train-time calibration, this effect did not yield any statistically significant differences, while for test-time calibration, it led to statistically different results. When using two separate transducers, only free-hand calibration could be used.

4.4 Discussion

The M2M transfer function has the potential to be implemented in practice as it does not rely on the acquisition of test domain data from classification samples to calibrate the classifier. The approach can provide a practical means to transfer DL models between imaging machines and to transfer data from different sources to the desired domain, thereby significantly reducing model development costs. In this chapter, an M2M transfer function

Table 4.9: Different Transducers in terms of Classification Accuracy

No	*Experiment*	*ResNet-Acc.*	*DenseNet-Acc.*
1	Train-Time with L11-5 vs L9-4	93.49±3.01	95.33±1.70
2	Test-Time with L11-5 vs L9-4	97.51±0.76	97.57±0.72
3	Train-Time with L9-4 vs L9-4	93.57±0.70	94.27±0.99
4	Test-Time with L9-4 vs L9-4	**98.66±0.33**	**98.90±0.35**

Table 4.10: Different Transducers in terms of AUC Scores

No	*Experiment*	*ResNet-AUC*	*DenseNet-AUC*
1	Train-Time with L11-5 vs L9-4	0.977±0.019	0.988±0.007
2	Test-Time with L11-5 vs L9-4	0.996±0.002	0.997±0.002
3	Train-Time with L9-4 vs L9-4	0.982±0.004	0.984±0.004
4	Test-Time with L9-4 vs L9-4	**0.999±6e-4**	**0.999±5e-4**

Table 4.11: p-values for Table 4.3 based on classification accuracies

No	1	2	3	4	5	6	7	8
1	*	<.01	<.01	<.01	<.01	<.01	<.01	<.01
2	<.01	*	<.01	<.01	<.01	<.01	<.01	<.01
3	<.01	<.01	*	<.01	<.01	<.01	<.01	<.01
4	<.01	<.01	<.01	*	<.01	<.01	<.01	<.01
5	<.01	<.01	<.01	<.01	*	<.01	<.01	<.01
6	<.01	<.01	<.01	<.01	<.01	*	<.01	.06
7	<.01	<.01	<.01	<.01	<.01	.43	*	.77
8	<.01	<.01	<.01	<.01	<.01	<.01	<.01	*

was investigated using different calibration phantoms and different acquisition strategies for acquiring calibration data. Along with the M2M transfer function, BN layers were modified to utilize batch statistics for training and evaluation. We observed that the M2M transfer function was effective at calibrating a DL model between imaging machines, increasing mean classification accuracy from 50.01% to 98.73% and mean AUC from 0.405 to 0.999 for ResNet architecture; increasing mean accuracy from 47.19% to 98.96% and mean AUC from 0.488 to 0.999 for DenseNet architecture.

In Tables 4.3 and 4.4, we mainly observe that the M2M transfer function provided perfect calibration under machine-level data mismatches. Compared to the baseline methods, the proposed method achieved a significant performance improvement. Additionally, there were multiple interesting observations that can be derived from the tables. First, in the case of no calibration, the use of training statistics resulted in very poor performance, as expected. However, the utilization of calibrated statistics and test statistics led to a significant improvement in accuracy and AUC. The accuracy improved from 50% to the range of 70-75%, and the AUC increased from 0.5 to above 0.9. It is worth noting that a significant improvement was achieved by using calibrated statistics even without calibrating input data.

Table 4.12: p-values for Table 4.4 based on AUC scores

No	1	2	3	4	5	6	7	8
1	*	<.01	<.01	<.01	<.01	<.01	<.01	<.01
2	<.01	*	<.01	<.01	<.01	<.01	<.01	<.01
3	<.01	<.01	*	<.01	<.01	<.01	<.01	<.01
4	<.01	<.01	<.01	*	<.01	.38	<.01	.04
5	<.01	<.01	<.01	<.01	*	.56	.13	.43
6	<.01	<.01	<.01	<.01	.01	*	.32	.37
7	<.01	<.01	<.01	<.01	.08	.02	*	.63
8	<.01	<.01	<.01	<.01	<.01	<.01	<.01	*

Table 4.13: p-values for Table 4.5 based on classification accuracy

No	1	2	3	4
1	*	<.01	<.01	<.01
2	<.01	*	<.01	<.01
3	<.01	<.01	*	<.01
4	<.01	<.01	<.01	*

This improvement was observed solely by implementing calibration for the statistics used in the normalization step, demonstrating the potential of the M2M transfer function.

Another important observation from Tables 4.3 and 4.4 is that the test-time calibration performed significantly better than train-time calibration in terms of accuracy and AUC. This observation differs from the previous chapter on the setting transfer function and was achieved after BN layer modification. However, further study is needed to understand the performance differences between train-time and test-time. In terms of network architecture, it was found that the proposed method was effective for both ResNet and DenseNet and test-time calibration provided perfect calibration for both architectures. In comparison to the baselines, the proposed method provided significant performance improvement in terms of both accuracy and AUC. It is noteworthy that the performance of Experiment 4, fine-tuning, and BN freezing were statistically insignificant in terms of AUC for the ResNet architecture, verifying the usefulness of the proposed method.

Given the best-performing approach, the calibration approach holds other advantages over the baseline methods. The proposed method does not require data acquisition from classification samples from testing machines. Instead, it only requires a single calibration view from a calibration phantom. This

Table 4.14: p-values for Table 4.6 based on AUC scores

No	1	2	3	4
1	*	<.01	<.01	<.01
2	<.01	*	<.01	<.01
3	<.01	<.01	*	<.01
4	<.01	<.01	<.01	*

Table 4.15: p-values for Table 4.7 based on classification accuracy

No	1	2	3	4
1	*	<.01	<.01	<.01
2	<.01	*	<.01	<.01
3	<.01	<.01	*	<.01
4	<.01	<.01	<.01	*

aspect of the proposed method, especially, positions it as a practical and clinically viable method.

In Tables 4.5 and 4.6, Calibration Phantom 1 and Calibration Phantom 2 were used for both train-time and test-time calibration. We observed that Calibration Phantom 1 resulted in better calibration for both train-time calibration, and test-time calibration in terms of accuracy and AUC. These results suggest that the selection of a calibration phantom was relevant to performance. Even though the proposed method did not rely on the acquisition of real samples from the test domain, one could hypothesize that the calibration phantom selection should align with the classification samples. As known from standard QUS [138], the properties of the calibration phantom, specifically speed of sound and attenuation, should resemble those of the test and training domain samples to enhance the calibration process.

Another interesting aspect of calibration phantom selection is the SNR values obtained for the Wiener implementation. Through qualitative investigation of the overall SNR values obtained from ultrasound signals from the calibration phantoms, we found that Calibration Phantom 2 had a higher SNR for lower frequencies, while Calibration Phantom 1 had a higher SNR for higher frequencies. Therefore, one might conclude that the calibration of higher frequencies is more important for calibration performance than the calibration of lower frequencies. However, further studies in DL-based approaches should be conducted to develop a systematic approach for selecting a calibration phantom based on properties of the real sample in the training

Table 4.16: p-values for Table 4.8 based on AUC scores

No	1	2	3	4
1	*	<.01	<.01	<.01
2	<.01	*	<.01	<.01
3	<.01	<.01	*	<.01
4	<.01	<.01	.07	*

Table 4.17: p-values for Table 4.3 based on classification accuracy

No	1	2	3	4
1	*	<.01	.49	<.01
2	<.01	*	<.01	<.01
3	.13	<.01	*	<.01
4	<.01	<.01	<.01	*

domain, which is known.

In Tables 4.7 and 4.8, stable acquisition and free-hand acquisition were investigated in terms of calibration performance. In stable acquisition, the M2M transfer function was calculated using a single fixed view from the calibration phantom. In free-hand acquisition, a video of ultrasound frames was recorded, and the M2M transfer function was calculated by averaging it over the frames. The results indicate that free-hand calibration does not have any advantage over stable calibration. This may sound counter intuitive at first as free-hand acquisition uses more frames to calculate a M2M transfer function; however, this observation actually verifies the robustness of the Wiener-inspired implementation against noise. Apparently, the Wiener-inspired implementation provides a robust method to calibrate data mismatches using just a single, fixed ultrasound frame. That being said, as a future direction, utilizing multiple calibration views to enhance calibration performance still remains an attractive avenue.

In Tables 4.9 and 4.10, the calibration method was tested against a mismatch of the transducer at the testing machine. It was observed that, during

Table 4.18: p-values for Table 4.10 based on AUC scores

No	1	2	3	4
1	*	<.01	.84	<.01
2	<.01	*	<.01	<.01
3	.13	<.01	*	<.01
4	<.01	<.01	<.01	*

train-time calibration, the use of a different transducer at the testing machine did not yield any statistically relevant change assuming the bandwidths were similar. However, for test-time calibration, a 1-1.5% accuracy loss and a 0.002-0.003 AUC score loss occurred. This verifies the robustness of the proposed approach under transducer mismatch; however, further study will be needed to enhance our understanding.

The results of this chapter highlight several potential future directions. First, the findings suggest that the selection of a calibration phantom can significantly impact performance. Specifically, the speed of sound and attenuation of the calibration phantom are expected to play a critical role. In this paper, speed of sound characteristics varied between 1535-1545 $\mathrm{m}\times\mathrm{s}^{-1}$ and attenuation characteristics varied between 0.1-0.7 $\mathrm{dB}\times\mathrm{cm}^{-1}\times\mathrm{MHz}^{-1}$. As a future direction, expanding these ranges and observing their effects on calibration performance would lead to a systematic approach for choosing a calibration phantom. Therefore, developing a systematic procedure for choosing a calibration phantom remains an important problem. Second, enhancing calibration performance by leveraging multiple calibration views remains unsolved. Although in this work, free-hand calibration couldn't provide additional performance improvement, utilizing multiple views from a calibration phantom could be effective in certain scenarios. Third, the impact of using different transducers, and hence, different usable frequency ranges on calibration, and devising solutions to address potential challenges arising from variations in transducer bandwidth, requires additional study. Even though the proposed approach works effectively between an L9-4 and an L11-5 transducer, it is expected that the technique will break down as the bandwidth overlap between systems and their respective transducers is reduced. One could try to match the ranges and/or implement the transfer function over the intersecting frequencies. Similarly, the effects of different acquisition techniques, such as plane wave imaging versus line by line imaging or even changes in sampling rate, on the calibration may also affect the ability to transfer classification models from one machine to another. Another intriguing aspect is the digitization bit. In this chapter, digitization bits were matched between domains. However, in cases of strong quantization artifacts or highly scattering materials the calibration performance may be affected. Finally, from a security perspective, in scenarios where DL model transferability is not desired, it may be possible to develop de-

fense mechanisms, such as introducing sampling rate mismatches along with mismatched digitization bits. Moreover, if data acquired from multiple machines can be combined through a M2M transfer function, the increase in data availability, i.e., incorporating the data from multiple machines, could lead to improved DL models. The code for the implementation of training, testing, and calibration can be accessed at the following repository: https://github.com/usoylu2/m2m. The dataset is available for use via the following link: https://uofi.box.com/s/d9ecw002ree6gj9tlplz7t0i2f1ojbk7.

CHAPTER 5

THE ART OF THE STEAL: PURLOINING DEEP LEARNING MODELS DEVELOPED FOR AN ULTRASOUND SCANNER TO A COMPETITOR MACHINE

5.1 Motivation

The transfer function approach has proven effective for calibrating deep learning (DL) algorithms in quantitative ultrasound (QUS) in the previous chapters, addressing data shifts at both the acquisition and machine levels. Expanding on this approach, we develop a strategy to 'steal' the functionality of a DL model from one ultrasound machine and implement it on another, in the context of QUS. This demonstrates the ease with which the functionality of a DL model can be transferred between machines, highlighting the security risks associated with deploying such models in a commercial scanner for clinical use.

The integration of DL-powered biomedical ultrasound has the potential to enhance the quality of medical services through automation and efficiency. Therefore, the adoption of DL-powered biomedical ultrasound imaging in a clinical setting has become the coveted goal for both industry and academia. Thanks to substantial efforts [103, 111–115, 125, 141] dedicated to overcoming challenges posed by DL-based models in clinical settings, this goal now can be within reach. Although ongoing efforts persist in addressing these challenges, DL-powered algorithms have reached a stage of maturity and increasingly adoption of such algorithms will occur clinically. Observing the pace of the development in this field, it is reasonable to claim that DL-based QUS approaches in clinical setting are becoming a reality rather than a distant goal.

Given the advancement, there could still be concerns or undiscovered challenges on the path that can delay the wide adoption of DL-based QUS methods in clinic. In this chapter, we identify one such concern: the ease with which the functionality of DL-based QUS methods can be stolen from one

machine and implemented on another, competitor machine. There can be great expense associated with the development of DL models for medical diagnostics. Specifically, a major expense is in the accumulation of large amounts of data with expert annotation and labelling. The cost of labelling copious amounts of data for DL training can be expensive and take years of investment. Companies investing in the development of such functionalities may be deterred from deploying their DL-based models to their own machine and making it available for clinical use, if competitors can effortlessly access their machine and transfer their functionality. Therefore, as an outcome of this chapter, we highlight the necessity of improving DL-model security in the context of biomedical ultrasound imaging by demonstrating the ability to purloin a DL-model from one machine, i.e., the victim machine, and deploy the DL-model on a different machine, i.e., the perpetrator machine.

The most relevant framework to this study is black-box unsupervised domain adaptation (UDA). In black-box UDA, we do not have access to the model's internals which means that its weights, its architecture, training process, and training data are unknown. However, we have access to input-output interface and there is unlabeled data available for the machine to which we aim to transfer functionality. This is the most relevant framework for security threats in deployment of DL-based models in the clinic. From the perspective of competitors, the process of stealing the model can be accomplished through a simple stepwise process. First, the competitor acquires the victim machine, gaining access to its input-output interface. Next, the competitor acquires unlabeled data utilizing their own machine in the pursuit of their own model development, making unlabeled data abundant. The unlabelled data or images from the perpetrator machine are transferred to the victim machine, i.e., test-time calibration. Labels are given to each image from the DL model on the victim machine. These labels are then used in the perpetrator machine to train the model.

The proposed method, depicted in Fig. 5.1, utilizes a transfer function at test-time to calibrate mismatches between the victim and the perpetrator machines. Subsequently, pseudolabels can be obtained for unlabeled data from the perpetrator machine through the input-output interface of the victim machine. These labels can be still noisy. Therefore, we propose utilizing IterLNL to further refine the labels. Then, a new DL model can be trained for the perpetrator machine utilizing these refined labels, which copies the

functionality of DL-based model from the victim to the perpetrator machine. The proposed method accesses only the input-output interface without accessing model internals of the victim machine, which are presumed not to be available. Another note is that the transfer function method requires the acquisition of calibration data using the victim and the perpetrator machines with a calibration phantom.

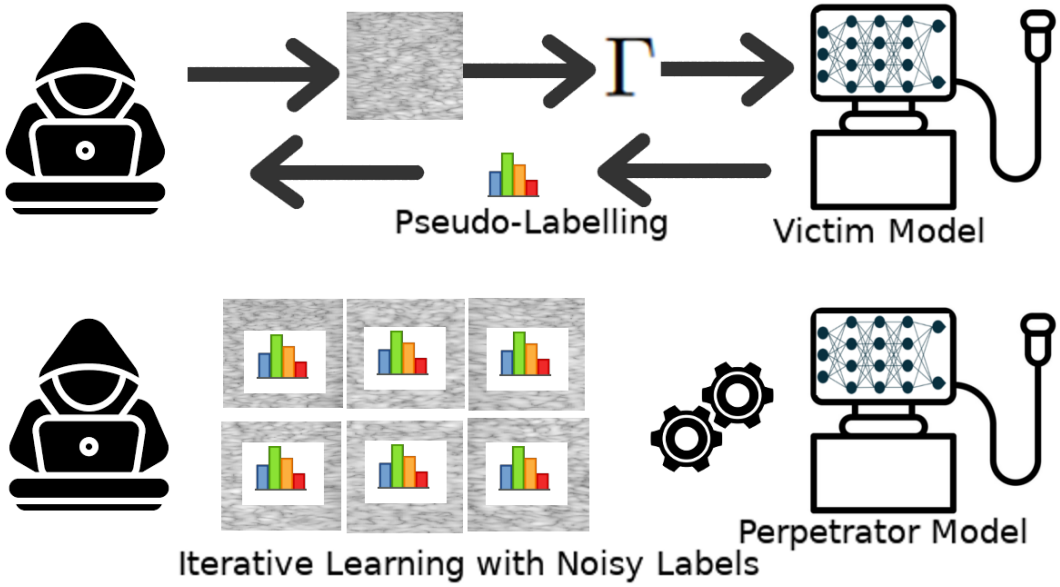


Figure 5.1: The proposed method

5.2 Methods

5.2.1 Proposed Approach

The proposed approach combines the transfer function with IterLNL [119]. The transfer function is able to calibrate data mismatches between ultrasound machines over a defined bandwidth. In the previous chapter, we discussed changing the operation mode of batch normalization layers along with the transfer function. However, in this chapter, we assume that we have no access to the model internals, and therefore, we cannot change the mode of operation in batch normalization layers. Given these circumstances, it has

been observed that utilizing the transfer function between two machines at test-time, i.e., converting data acquired on one machine to the DL model developed on another machine, achieved approximately an AUC scores of 0.99 and accuracy of 80 percent for a QUS task. Therefore, as a refinement step, IterLNL, which conducts iterative learning with noisy labels, is performed to further refine the labels. Following that, a new learning algorithm can be developed using these refined labels for the perpetrator machine. Therefore, the propose approach at a high level involves 5 steps:

1. Gather calibration data using the victim machine and the perpetrator machine.

2. Calculate the transfer function between the perpetrator and the victim machines.

3. Obtain pseudo-labels for the unlabeled data from the perpetrator machine utilizing the transfer function and the input-output interface of the victim machine.

4. Implement iterative learning with pseudo-labels for refinement.

5. Obtain the learning algorithm for the perpetrator machine using the refined pseudo-labels.

The development of the transfer function is identical to the previous chapter; however, in this chapter, we utilize it at test-time only. We obtained 10 calibration views from a fixed location of the calibration phantom. Additionally, following the methodology of the original transfer function work, we computed the transfer function at various depths and employed the Wiener implementation of it.

For simplicity, Γ will be referred to the Wiener implementation. The key idea in IterLNL is that given noisy labels, a new model is learned iteratively based on anchors, which are the data points exhibiting the lowest training loss. Let's denote the victim model F_{victim}, unlabeled data $X_{perpetrator}$, the perpetrator model $F_{perpetrator}$ and anchors $U_{perpetrator}$. Then, pseudo-labels $y_{perpetrator}$ can be obtained by applying the input-output interface of F_{victim} to $\Gamma(X_{perpetrator}) = X_{perpetrator \rightarrow victim}$. At each iteration, pseudo-labels will be acquired again using the learnt perpetrator model $F_{perpetrator}$. An important parameter in IterLNL is the noise rate, denoted as ϵ. In the original work,

it was estimated using $y_{perpetrator}$ and $X_{perpetrator \to victim}$. However, in this chapter, we leveraged our prior knowledge regarding the transfer function, establishing the noise rate at approximately 20 percent. Then, the proposed approach that combines the transfer function and IterLNL can be depicted in Algorithm 3. One note is that after the iterative learning process, $F_{perpetrator}$ can generate highly accurate labels for the unlabeled data $X_{perpetrator \to victim}$, which has been transformed using the transfer function Γ. As a final step, we may update $F_{perpetrator}$ so that it operates directly within the perpetrator machine domain. This involves training it directly on $X_{perpetrator}$ without the application of the transfer function Γ.

Algorithm 3: Proposed Approach

Input: Victim Model F_{victim}, Unlabeled data $X_{perpetrator}$, Transfer Function Γ, $\epsilon = 20$

Output: Perpetrator Model $F_{perpetrator}$

Step1: Apply Transfer Function $X_{perpetrator \to victim} \leftarrow \Gamma(X_{perpetrator})$;

Step2: Standardization $X_{perpetrator \to victim} \leftarrow (X - \mathbb{E}X_{perpetrator \to victim})/\sigma X_{perpetrator \to victim}$;

Step3: Set $F = F_{victim}$ as the pseudo-labeling model;

Training: Iterative Learning

while *iteration* **do**

  Acquire noisy labels $y_{perpetrator}$ using F;

  Initialize $F_{perpetrator}$;

  Obtain anchors $U_{perpetrator}$ (100-ϵ percent of $X_{perpetrator \to victim}$ returning lowest $Loss(F_{perpetrator}(X_{perpetrator \to victim}), y_{perpetrator})$;

  while *epoch* **do**

    Update $F_{perpetrator}$ using anchors $U_{perpetrator}$;

    Update anchors $U_{perpetrator}$;

  end

  Update F as $F_{perpetrator}$

end

Final Step: Obtain refined pseudo-labels $y_{perpetrator} = F(X_{perpetrator \to victim})$ and retrain $F_{perpetrator}$ using $(X_{perpetrator}, y_{perpetrator})$ which works directly on the perpetrator machine without Γ

5.2.2 Phantoms

The experiments utilized two distinct tissue-mimicking phantoms as classification phantoms and one calibration phantom, shown in Fig. 5.2. The QUS

task in this study was to determine the class identity of the ultrasound data, identical to previous chapters. Clinically, this could correspond to a binary tissue classification problem, e.g., is the tissue diseased or not diseased. The objective of the study is to steal this functionality from the victim machine and implement it on the perpetrator machine.

Classification Phantom 1 mimics the characteristics of the human liver [130] and the construction details are provided in [131]. The attenuation coefficient slope for Classification Phantom 1 was measured around 0.4 dB$\times$cm^{-1} $\times$MHz $^{-1}$. Its speed of sound was 1540 $m \times s^{-1}$. It exhibited macroscopic uniformity. The sole source of non-uniformity in Classification Phantom 1 stemmed from the random distribution of microscopic glass bead scatterers, ranging in diameter from 75 to 90 μm.

Classification Phantom 2 was characterized as a low-attenuation phantom [132] and the construction details are provided in [133]. The same weakly-scattering agar, serving as the background material, was utilized in Classification Phantom 2 but included glass-bead scatterers of varying sizes, ranging from 39 to 43 μm in diameter. Its speed of sound was 1539 $m \times s^{-1}$. The attenuation coefficient slope was measured around 0.1 dB$\times$cm$^{-1}\times$MHz $^{-1}$.

The Calibration Phantom was a commercial QUS reference phantom (part no. 14090502, serial no. 221447541) from CIRS, Inc., Norfolk, VA. It had an attenuation coefficient slope of approximately equal to 0.74 dB$\times$cm$^{-1}\times$MHz $^{-1}$. Its speed of sound was 1545 $m \times s^{-1}$.

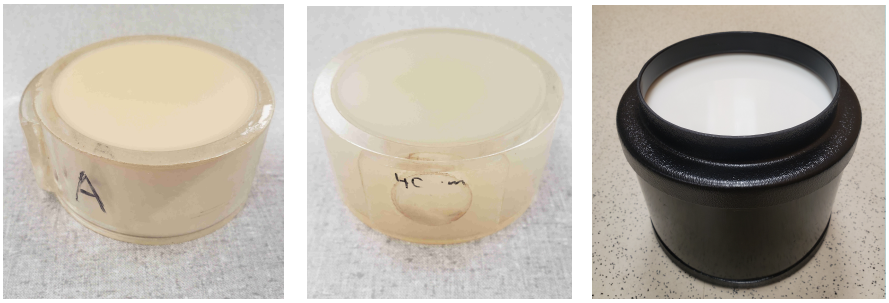


Figure 5.2: Classification Phantoms and Calibration Phantom

5.2.3 Ultrasound Machines

The victim model was trained using data from the SonixOne system. The perpetrator model aimed to replicate the same functionality as the victim model but for a Verasonics Vantage 128. Throughout the experiments, an L9-4 transducer was consistently utilized. One note is that there was sampling rate mismatches between these sources: the SonixOne system captured post-beamformed radio-frequency (RF) data at 40 MHz and the Verasonics system acquired raw channel data a sampling rate of 50 MHz. These differences were mitigated by preprocessing the Verasonics data. First, delay and sum beamforming was implemented on the Verasonics data. Following that, a multirate FIR filter was designed with an interpolation factor of 4 and a decimation factor of 5 to convert the sampling rate to 40 MHz. All these preprocessing operations were done on Matlab, specifically using the 'designMultirateFIR' and the 'dsp.FIRRateConverter' functions. After these preprocessing steps, post-beamformed RF data at a matching sampling frequency of 40 MHz was obtained from the Verasonics system.

The victim model was trained using 1,000 frames per classification phantom, totaling 2,000 frames, acquired using the SonixOne machine. Similarly, for the perpetrator model, 2,000 frames were acquired using the Verasonics machine. These acquisitions involved recording a video of frames while moving the transducer across the phantom surface. For calibration data, both the SonixOne and Verasonics machines were utilized. By securing the transducer using a bar clamp holder, ten identical frames were captured from precisely the same position on the calibration phantom. Multiple frames were recorded to mitigate or reduce any potential effects of electrical noise through averaging.

As imaging settings, line by line acquisition with 2 cm axial focal point was used for both machines. In the SonixOne, the center pulse frequency was set at 9 MHz and its output power level was set at 0 dB. In the Verasonics, the center pulse frequency was set at 5 MHz and its output power level was set at 45.2 Voltage.

5.2.4 Data Preparation

Patch extraction process was implemented, as described in earlier chapters. For the victim model, 2000 frames from the SonixOne machine were split into a training and a validation set with a ratio of 4:1. On the other hand, for the perpetrator model, 2000 frames from the Verasonics machine were split into a training and a testing set with a ratio of 1:1. Therefore, in the experiments, the perpetrator model was trained using the training set consisting of 1000 frames, and it was evaluated using the test set consisting of 1000 ultrasound frames.

5.2.5 Training

The DL training was done with a workstation equipped with four NVIDIA RTX A4000. Each experiment was conducted using all four RTX A4000s in parallel. The PyTorch library [134] was utilized for all experiments.

In training, we utilized the Adam algorithm [135] as the optimizer. The batch size was 2,048 for all experiments to maximize memory utilization. During training, a standard method for data augmentation involved applying a horizontal flip with a 50% probability by default. As training loss, cross-entropy loss was utilized. Z-score normalization/standardization was carried out at the patch level as a data preprocessing step. This process includes subtracting the mean patch, then dividing by the standard deviation patch.

In iterative learning, during each iteration, ten percent of the training data was allocated as a validation set. This allocation allowed for the observation of training metrics and the determination of hyperparameters, such as setting the epoch number to 10 and the learning rate to 1×10^{-5}. In results, the experiments were repeated ten times starting from the iterative learning. The performance of the final $F^{perpetrator}$ on the test data from the Verasonics machine was reported in terms of the average of the classification accuracies, the average area under the receiver operator characteristic (ROC) curve (AUC) and their respective standard deviations. The results were obtained patch-wise, and the observed variance was caused by the random initialization of network parameters at each iteration.

5.2.6 Network Structure

The victim model was a DenseNet-201 [142] with minor tweaks to their input-output relationships tailored to our specific problem. The perpetrator model was chosen as ResNet-34 [96] with similar tweaks. These tweaks involve adjusting the first convolutional layer to accommodate single-layer inputs and modifying the final output layer to output a single score for a binary classification setup. For network parameter initialization, pretrained weights were used, except for the first convolutional layers and the last fully connected layers, which were initialized using the default method in PyTorch. During training, all the parameters were unfrozen and updated through backpropagation. The necessity of selecting different model architectures between the victim and the perpetrator arose from the black-box nature of the victim setup, resulting in inevitable differences in architectural design. As long as the selected architecture has enough complexity to solve the task, the attack should be successful. In practice, network selection may require search and hyperparameter tuning, especially in a black-box setup. However, if the perpetrator knows the complexity of the victim model or its architecture, the perpetrator can use the victim's architecture directly.

5.3 Results

The victim model scored an accuracy of 99.93 percent and an AUC of 0.999 on its validation set. This performance serves as the upper bound achievable with the proposed method, acknowledging an anticipated performance degradation due to machine-level mismatches between the victim and perpetrator machines.

Our investigation primarily focused on three aspects. First, we explored the impact of the transfer function in our proposed approach. To examine this, we compared two cases: utilizing the transfer function and omitting it, essentially skipping step 1 in Algorithm 3. Second, our exploration delved into the importance of the priors utilized in the proposed method. These priors covered both the label distribution and the noise rate parameter ϵ. Third, we explored the number of ultrasound frames from the perpetrator machine needed for successful attack.

Table 5.1: Classification Accuracy for Ablation Study of Transfer Function in Model Extraction

No	*Experiment*	*Accuracy (Iter=2)*	*Accuracy (Iter=5)*	*Accuracy (Iter=10)*
1	Iterative Learning with TF	**97.83±0.28**	**97.78±0.25**	**97.69±0.36**
2	Iterative Learning without TF	89.98±3.88	89.22±7.39	85.51±1.60

Table 5.2: AUC Scores for Ablation Study of Transfer Function in Model Extraction

No	*Experiment*	*AUC (Iter=2)*	*AUC (Iter=5)*	*AUC (Iter=10)*
1	Iterative Learning with TF	**0.9976±5e-4**	**0.9975±5e-4**	**0.9974±7e-4**
2	Iterative Learning without TF	0.9618±3e-2	0.9401±6e-2	0.9020±1e-1

5.3.1 The Effect of the Transfer Function

We conducted an ablation study for the proposed method consisting of two experiments. The first experiment involved using the transfer function within the iterative learning schema. In the second experiment, we omitted the transfer function and solely implemented the iterative learning schema. These experiments were designed to gauge the contribution of the transfer function method. The results for different iteration numbers (2, 5, and 10) are presented in Tables 5.1 and 5.2.

5.3.2 Priors

We conducted a robustness study for the proposed method consisting of two sets of experiments. The proposed method utilized two information priors: label distribution and noise rate ϵ. We utilized the prior for the label distribution to obtain noisy labels through the pseudo-labeling model. In our balanced binary classification setup, where half of the unlabeled data was expected to belong to one class and the other half to the other, we used this prior knowledge when assigning class identities in pseudo-labeling. Scores were thus thresholded at the median score within the training set. In

Table 5.3: Classification Accuracy for various Label Distribution

No	*Experiment*	*Accuracy (Iter=2)*	*Accuracy (Iter=5)*	*Accuracy (Iter=10)*
1	40^{th} percentile	97.35±1.44	97.71±0.33	97.70±0.51
2	50^{th} percentile	**97.83±0.28**	**97.78±0.25**	97.69±0.36
3	60^{th} percentile	97.55±0.47	97.48±0.46	**97.82±0.38**

Table 5.4: AUC Scores for various Label Distribution

No	*Experiment*	*AUC (Iter=2)*	*AUC (Iter=5)*	*AUC (Iter=10)*
1	40^{th} percentile	0.9975±7e-4	0.9974±6e-4	0.9975±7e-4
2	50^{th} percentile	**0.9976±5e-4**	**0.9975±5e-4**	0.9974±7e-4
3	60^{th} percentile	0.9974±5e-4	0.9972±7-e4	**0.9976±7e-4**

Tables 5.3 and 5.4, we explored scenarios where the threshold value deviated from the exact median. In other words, we considered cases where the prior information about the label distribution was vague and not precisely known. On the other hand, we utilized the prior of noise rate in determining anchors to be used in training. Based on previous findings, the average accuracy for the transfer function at test-time was approximately 80 percent, indicating a noise rate of 20 percent. In Tables 5.5 and 5.6, we examined scenarios where the noise rate deviated from this established percentage.

5.3.3 Data-set Size

We examined the number of ultrasound frames needed from the perpetrator machines to achieve a successful attack on the victim model. In Fig. 5.3, the y-axis represents the classification accuracy achieved on the perpetrator machine using the proposed approach and x-axis represents the number of ultrasound frames utilized from the perpetrator machine. In this result, we performed two iterations to refine pseudolabels, setting ϵ to 0.8.

Table 5.5: Classification Accuracy for various Noise Rate ϵ

No	*Experiment*	*Accuracy (Iter=2)*	*Accuracy (Iter=5)*	*Accuracy (Iter=10)*
1	10 percent	**98.25±0.29**	**98.13±0.28**	**98.25±0.28**
2	20 percent	97.83±0.28	97.78±0.25	97.69±0.36
3	30 percent	97.60±0.82	97.02±1.57	96.45±2.46

Table 5.6: AUC Scores for various Noise Rate ϵ

No	*Experiment*	*AUC (Iter=2)*	*AUC (Iter=5)*	*AUC (Iter=10)*
1	10 percent	**0.9983±5e-4**	**0.9982±4e-4**	**0.9983±5e-4**
2	20 percent	0.9976±5e-4	0.9975±5e-4	0.9974±7e-4
3	30 percent	0.9969±2e-3	0.9954±4e-3	0.9932±1e-2

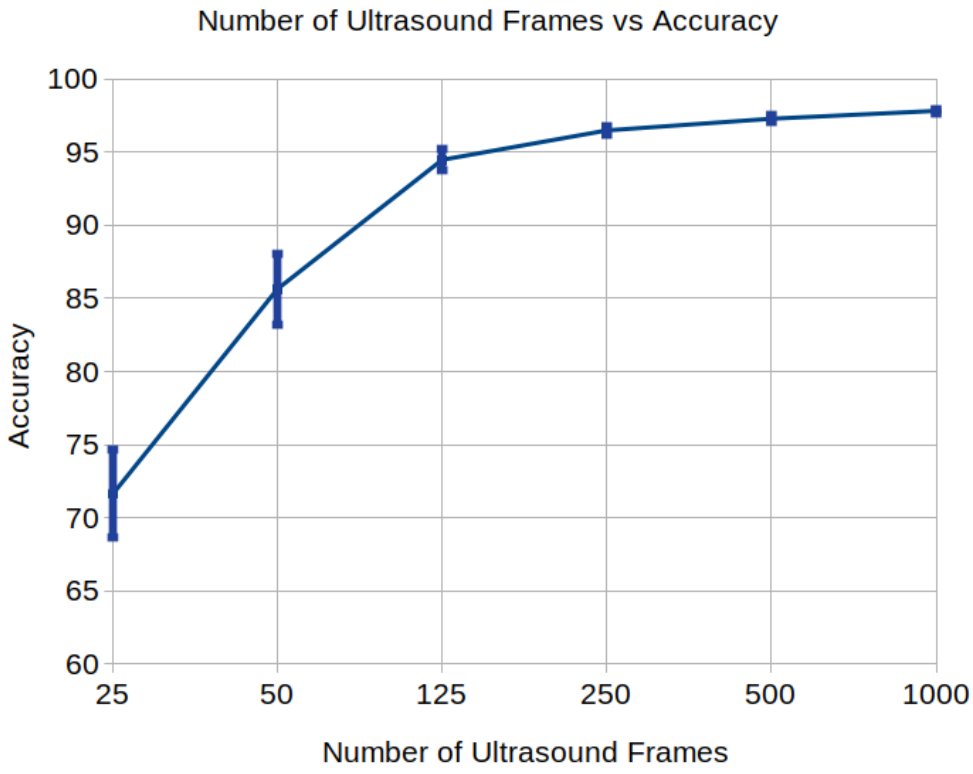


Figure 5.3: Number of Ultrasound Frames Needed for Successful Attack. Error bars represent one standard deviation

5.4 Discussion

The proposed approach utilized the transfer function within an iterative learning schema. The transfer function aids in decreasing noise in the pseudo-

labeling process, while the iterative schema further refines the labels. The study's key observation highlights the ease with which the functionality of DL-based QUS methods can be stolen, raising valid concerns regarding model security. The ease with which a competitor company can transfer this functionality to its own machines poses a risk, potentially causing delays in deploying DL-based QUS methods in clinical settings. Therefore, as an outcome of the study, we emphasize the need for a deeper understanding of model security and the development of secure deployment methods tailored for ultrasound imaging contexts. Developing techniques to safeguard machine learning models emerges as a critical challenge in the adoption of wider clinical applications of machine learning-based algorithms. As of today, there is no known defense against such model extraction attacks [120].

The study primarily aligns with the black-box UDA framework, wherein we lacked access to the internal workings of the victim model and solely utilized its input-output interface. Leveraging this interface, we obtained pseudo-labels from the unlabeled data of the perpetrator machine. Using these pseudo-labels we were able to train the DL model on the perpetrator machine without the need for annotation and labelling of the data by an outside expert. Our primary aim was to demonstrate the ease of replicating the functionality of the victim machine. While there might be technical differences such as model architecture between the victim and perpetrator machines, our focus remained on the functionality, considering it as the pivotal aspect of our investigation.

In Tables 5.1 and 5.2, we examined the impact of integrating the transfer function within the iterative schema. The findings highlighted a significant positive impact of the transfer function on the stealing performance. In the absence of the transfer function, the iterative schema yielded accuracy scores ranging between 85-90 percent. However, upon incorporating the transfer function, the performance consistently surged to the 98 percent range across all iteration numbers. Its impact on the AUC score also demonstrated a positive effect, elevating the AUC score to 0.998. These tables illustrated the substantial contribution brought by the transfer function to the iterative schema. While the study focused on the utilization of the IterLNL schema, the transfer function holds the potential to similarly benefit all other black-box UDA methods.

In Tables 5.3 and 5.4, we explored the effects of the prior about the label

distribution. In our experimental setup, the DL models generated a single score. When the class score approached 0, the likelihood of it being class 0 increased. When the score approached 1, the likelihood of it being class 1 increased. Following that, in Algorithm 3, during the acquisition of noisy labels $y_{perpetrator}$ using F, we employed our prior knowledge of the label distribution to set the threshold between class 0 and 1. As we operated within a balanced binary classification setup, we determined the threshold as the median value (50^{th} percentile). In Tables 5.3 and 5.4, we investigated scenarios involving inaccurate or approximate prior knowledge of the label distribution such as setting threshold at 40^{th} or 60^{th} percentile during the acquisition of noisy labels. We observed that the proposed method demonstrated robust performance across various scenarios, maintaining an accuracy score consistently above 97 percent and an AUC score of 0.997. As a future direction, it is possible to explore various problem scenarios, including imbalanced classification, to increase the impact of the proposed approach. Moreover, the proposed approach can be scaled to a multi-class problem setting, as in certain clinical tasks, there may be multiple types of tissue states. As long as we have access to the input-output interface and obtain pseudo-labels in a multi-class setting, we expect the proposed approach to scale well.

In Tables 5.5 and 5.6, we investigated the effects of the prior about the noise rate ϵ. The noise rate plays a crucial role in the iterative schema, influencing the selection of anchors at each epoch based on this parameter. Given the prior work on transfer function, the noise rate was estimated at 20 percent so that 80 percent of the data was utilized as anchors during the training. In Tables 5.5 and 5.6, we investigated scenarios involving inaccurate or approximate prior knowledge of the noise rate such as 10 and 30 percent during the acquisition of anchors. We observed that the proposed method demonstrated robust performance across various scenarios, maintaining an accuracy score consistently above 97 percent and an AUC score of 0.993.

In Fig. 5.3, we investigated the number of ultrasound frames needed from the perpetrator machine to achieve a successful attack. We used 25, 50, 125, 250, 500 and 1000 frames in the figure. Classification accuracies on the perpetrator were calculated by first performing patch extraction, obtaining pseudolabels, refining the labels, and training a machine learning model for the perpetrator machine. Then, using a hidden set, we obtained classification accuracies. We observed that by just utilizing 125 ultrasound frames from the

perpetrator machine, the proposed approach achieved a 94.48% classification accuracy. Developing a more data-efficient attack could be another direction for research.

Although the proposed approach does not require any expert annotation, it assumes the availability of unlabeled data from the perpetrator machine. Depending on the clinical task, acquiring the unlabeled data could be as costly as acquiring expert annotation, as it requires finding the right type of patients, addressing privacy issues, and conducting lab experiments. Furthermore, as a follow-up research direction, combining data with labels acquired from the victim machine with a small subset of expert-annotated data from the perpetrator machine could be interesting and could lead to higher accuracy and more efficient attacks.

The proposed method achieved a remarkable 98% classification accuracy and 0.998 AUC score on the perpetrator machine by solely utilizing the input-output interface of the victim model and the unlabeled data from the perpetrator machine. It is important to note that our assumptions included the ease with which the perpetrator machine could utilize the input-output interface, converting its data into the appropriate format and generating pseudo-labels. Furthermore, under the hood, we assumed the perpetrator has complete knowledge about the clinical task that the victim model is trained for. Therefore, the perpetrator acquires meaningful unlabeled data for the victim model. For instance, if the victim model characterizes tissue for certain organs and for certain patient distribution, we assumed the perpetrator has the complete knowledge of it. Implementing security enhancements might involve measures like hiding or restricting access to the input-output interface. Nevertheless, there remains a risk of hacking, especially in a development environment if the competitor gains ownership of the victim machine. An intriguing approach to improve model security could involve integrating security measures into the model's behavior, aiming to deceive the perpetrator machine by presenting misleading functionalities. In essence, this study underscores the critical need for an enhanced understanding of model security within the domain of DL-based ultrasound imaging. Hence, we identify model security as a critical future direction encompassing two key aspects: first, the development of new strategies to transfer DL-based functionalities to the perpetrator machine; and second, the development of security measures to defend against these potential at-

tacks. The code for this study can be accessed at the following repository: https://github.com/usoylu2/theartsteal. The dataset is available for use via the following link: https://uofi.box.com/s/d9ecw002ree6gj9tlplz7t0i2f1ojbk7.

CHAPTER 6

CONCLUSION

The thesis focused on designing algorithmic solutions to major challenges in DL-based QUS imaging. Towards this end, *Zone Training* was proposed as a data-efficient solution, and the *Transfer Function* was proposed to mitigate data mismatches. Expanding on the *Transfer Function*, a strategy was developed to "steal" a DL functionality from a victim machine to the perpetrator machine.

It has been observed that *Zone Training* is effective, especially in low data regimes. It can require a factor of 2-3 less training data in low data regimes to achieve similar classification accuracies compared to conventional training strategies. Additionally, performance in DL-based QUS can be improved by integrating global coordinates into the training, as in the case of depth-aware training. Furthermore, a novel form of mismatch, "zonal mismatches," has been identified. When training a DL algorithm for DL-based QUS using only one zone, the resulting model performs poorly on other zones. This finding suggests that, in order to improve clinical outcomes, the data collection process for DL algorithms should cover all regions of clinical interest.

The *Transfer Function* has been developed utilizing systems and signal perspectives, posing the problem of data mismatch as matching system responses. It has been shown to be effective for acquisition-related and machine-related data mismatches. Results from the Transfer Function approach suggests a practical and economical way for calibrating DL-based QUS, as it does not require data from actual samples. Instead, a calibration phantom that aligns with the actual samples is enough to calibrate DL models in the clinic, separating the calibration process from patient-related data acquisition. Therefore, it suggests a patient-free calibration approach.

Building on the success of transferring data from one machine to another machine, an approach to steal the DL functionality from a victim machine was introduced. The approach achieved high accuracies on the perpetrator

machine with ease of stealing. This highlights the need for security measures before deploying DL models in clinical settings.

DL-based QUS has the potential to transform clinical ultrasound imaging by enabling more accurate, precise, and automated assessments of tissue properties. The pace in this field is impressive, and the technology for DL-based algorithms is advancing rapidly, with a focus on safety and reliability. While solutions to some of the major challenges in clinically adopted DL-based ultrasound imaging, such as data mismatch and data scarcity, have been developed, we expect to uncover unrealized challenges on the path to clinical applications. Model security is one such challenge, and it could be a blocker, as companies may be less willing to invest in and implement DL solutions without securing their DL models.

The findings from this thesis have led to additional avenues for future research. First, the proposed methods have proven to be effective for the QUS tissue characterization task. It would be of great interest to investigate these methods for different ultrasound image analysis tasks, such as object detection and image segmentation. Second, various convolutional neural networks were utilized. It would be of great interest to examine the proposed methods with different architecture types, such as vision transformers. Regarding the *Zone Training*, the most compelling research question to be answered is how to systematically define the zone definitions per imaging setting, grounded in the physics of diffraction. This should include determining the optimal number of zones and their widths. Regarding the *Transfer Function*, the most compelling research question to be answered is how to select the calibration phantom systematically. Factors such as speed of sound, attenuation coefficients, and SNR are expected to play critical roles. It would also be extremely useful to study the *Transfer Function* under strong bandwidth variations between training and testing conditions to enhance its robustness further. Additionally, aging in the context of calibration would be of great interest. In clinical use, the ultrasound machine and/or ultrasound transducer may degrade over time. This degradation could also affect DL-based solutions. *Transfer Function* could provide a way to calibrate for this aging problem. By conducting periodic checks and obtaining calibration datasets, DL-based algorithms can be calibrated to account for changes due to aging. Moreover, there is a need for standardization and quantification of DL challenges in the context of QUS. This should include the selection of datasets,

network architectures, DL training, and evaluation methodologies. Developing standard datasets with quantified measures, such as divergent measures, would benefit the research community by ensuring that the datasets are good representations of clinical tasks in terms of difficulty and complexity. However, the highest need uncovered by the thesis is model security. This could be the main blocker in the adoption of clinically based DL solutions, unless necessary security solutions will be developed. This research path consists of two fronts: developing model extraction attacks and developing model security mechanisms against such attacks.

CHAPTER 7

REFERENCES

[1] Z. Akkus, J. Cai, A. Boonrod, A. Zeinoddini, A. D. Weston, K. A. Philbrick, and B. J. Erickson, "A survey of deep-learning applications in ultrasound: Artificial intelligence–powered ultrasound for improving clinical workflow," *Journal of the American College of Radiology*, vol. 16, no. 9, pp. 1318–1328, 2019.

[2] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, pp. 1–74, 2021.

[3] P. Chea and J. C. Mandell, "Current applications and future directions of deep learning in musculoskeletal radiology," *Skeletal radiology*, vol. 49, no. 2, pp. 183–197, 2020.

[4] X. Wu, D. Sahoo, and S. C. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing*, vol. 396, pp. 39–64, 2020.

[5] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2020.

[6] G. Yolcu, I. Oztel, S. Kazan, C. Oz, and F. Bunyak, "Deep learning-based face analysis system for monitoring customer interest," *Journal of ambient intelligence and humanized computing*, vol. 11, pp. 237–248, 2020.

[7] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE access*, vol. 7, pp. 128 837–128 868, 2019.

[8] S. K. Zhou, H. Greenspan, C. Davatzikos, J. S. Duncan, B. Van Ginneken, A. Madabhushi, J. L. Prince, D. Rueckert, and R. M. Summers, "A review of deep learning in medical imaging: Imaging traits, technology trends, case studies with progress highlights, and future promises," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 820–838, 2021.

[9] G. Wang, Y. Zhang, X. Ye, and X. Mou, *Machine learning for tomographic imaging*. IOP Publishing, 2019.

[10] M. Gaillochet, K. C. Tezcan, and E. Konukoglu, "Joint reconstruction and bias field correction for undersampled mr imaging," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020, pp. 44–52.

[11] B. E. Dewey, C. Zhao, J. C. Reinhold, A. Carass, K. C. Fitzgerald, E. S. Sotirchos, S. Saidha, J. Oh, D. L. Pham, P. A. Calabresi et al., "Deepharmony: A deep learning approach to contrast harmonization across scanner changes," *Magnetic resonance imaging*, vol. 64, pp. 160–170, 2019.

[12] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J. N. Chiang, Z. Wu, and X. Ding, "Embracing imperfect datasets: A review of deep learning solutions for medical image segmentation," *Medical image analysis*, vol. 63, p. 101693, 2020.

[13] Y. Fu, Y. Lei, T. Wang, W. J. Curran, T. Liu, and X. Yang, "Deep learning in medical image registration: a review," *Physics in Medicine & Biology*, vol. 65, no. 20, p. 20TR01, 2020.

[14] H.-P. Chan, L. M. Hadjiiski, and R. K. Samala, "Computer-aided diagnosis in the era of deep learning," *Medical physics*, vol. 47, no. 5, pp. e218–e227, 2020.

[15] D. Liu, K. S. Zhou, D. Bernhardt, and D. Comaniciu, "Search strategies for multiple landmark detection by submodular maximization," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2831–2838.

[16] Z. Xu, Y. Huo, J. Park, B. Landman, A. Milkowski, S. Grbic, and S. Zhou, "Less is more: Simultaneous view classification and landmark detection for abdominal ultrasound images," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II 11*. Springer, 2018, pp. 711–719.

[17] B. Jing, P. Xie, and E. Xing, "On the automatic generation of medical imaging reports," *arXiv preprint arXiv:1711.08195*, 2017.

[18] S. Liu, Y. Wang, X. Yang, B. Lei, L. Liu, S. X. Li, D. Ni, and T. Wang, "Deep learning in medical ultrasound analysis: A review," *Engineering*, vol. 5, no. 2, pp. 261–275, 2019.

[19] X. Liu, J. Shi, and Q. Zhang, "Tumor classification by deep polynomial network and multiple kernel learning on small ultrasound image dataset," in *International Workshop on Machine Learning in Medical Imaging*. Springer, 2015, pp. 313–320.

[20] J. Shi, S. Zhou, X. Liu, Q. Zhang, M. Lu, and T. Wang, "Stacked deep polynomial network based representation learning for tumor classification with small ultrasound image dataset," *Neurocomputing*, vol. 194, pp. 87–94, 2016.

[21] T. N. Nguyen, A. S. Podkowa, T. H. Park, R. J. Miller, M. N. Do, and M. L. Oelze, "Use of a convolutional neural network and quantitative ultrasound for diagnosis of fatty liver," *Ultrasound in Medicine & Biology*, vol. 47, no. 3, pp. 556–568, 2021.

[22] A. Han, M. Byra, E. Heba, M. P. Andre, J. W. Erdman Jr, R. Loomba, C. B. Sirlin, and W. D. O'Brien Jr, "Noninvasive diagnosis of nonalcoholic fatty liver disease and quantification of liver fat with radiofrequency ultrasound data using one-dimensional convolutional neural networks," *Radiology*, vol. 295, no. 2, pp. 342–350, 2020.

[23] M. Byra, A. Han, A. S. Boehringer, Y. N. Zhang, W. D. O'Brien Jr, J. W. Erdman Jr, R. Loomba, C. B. Sirlin, and M. Andre, "Liver fat assessment in multiview sonography using transfer learning with convolutional neural networks," *Journal of Ultrasound in Medicine*, vol. 41, no. 1, pp. 175–184, 2022.

[24] Z. Cao, L. Duan, G. Yang, T. Yue, Q. Chen, H. Fu, and Y. Xu, "Breast tumor detection in ultrasound images using deep learning," in *International Workshop on Patch-based Techniques in Medical Imaging*. Springer, 2017, pp. 121–128.

[25] G. Carneiro, J. C. Nascimento, and A. Freitas, "The segmentation of the left ventricle of the heart from ultrasound data using deep learning architectures and derivative-based search methods," *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 968–982, 2011.

[26] G. Carneiro and J. C. Nascimento, "Combining multiple dynamic models and deep learning architectures for tracking the left ventricle endocardium in ultrasound data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2592–2607, 2013.

[27] Y. Zhang, M. T. Ying, L. Yang, A. T. Ahuja, and D. Z. Chen, "Coarse-to-fine stacked fully convolutional nets for lymph node segmentation in ultrasound images," in *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2016, pp. 443–448.

[28] N. Baka, S. Leenstra, and T. van Walsum, "Ultrasound aided vertebral level localization for lumbar surgery," *IEEE Trans. Med. Im.*, vol. 36, no. 10, pp. 2138–2147, 2017.

[29] R. J. Van Sloun, R. Cohen, and Y. C. Eldar, "Deep learning in ultrasound imaging," *Proceedings of the IEEE*, vol. 108, no. 1, pp. 11–29, 2019.

[30] A. C. Luchies and B. C. Byram, "Deep neural networks for ultrasound beamforming," *IEEE transactions on medical imaging*, vol. 37, no. 9, pp. 2010–2021, 2018.

[31] D. Xiao, W. M. Pitman, B. Y. Yiu, A. J. Chee, and C. Alfred, "Minimizing image quality loss after channel count reduction for plane wave ultrasound via deep learning inference," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 10, pp. 2849–2861, 2022.

[32] J. Zhang, Q. He, Y. Xiao, H. Zheng, C. Wang, and J. Luo, "Ultrasound image reconstruction from plane wave radio-frequency data by self-supervised deep neural network," *Medical Image Analysis*, vol. 70, p. 102018, 2021.

[33] J. Lu, F. Millioz, D. Garcia, S. Salles, D. Ye, and D. Friboulet, "Complex convolutional neural networks for ultrafast ultrasound imaging reconstruction from in-phase/quadrature signal," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 69, no. 2, pp. 592–603, 2021.

[34] Y. Chen, J. Liu, X. Luo, and J. Luo, "Apodnet: Learning for high frame rate synthetic transmit aperture ultrasound imaging," *IEEE Transactions on Medical Imaging*, vol. 40, no. 11, pp. 3190–3204, 2021.

[35] D. Perdios, M. Vonlanthen, F. Martinez, M. Arditi, and J.-P. Thiran, "Cnn-based image reconstruction method for ultrafast ultrasound imaging," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 4, pp. 1154–1168, 2021.

[36] M. Gasse, F. Millioz, E. Roux, D. Garcia, H. Liebgott, and D. Friboulet, "High-quality plane wave compounding using convolutional neural networks," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 64, no. 10, pp. 1637–1639, 2017.

[37] D. Hyun, L. L. Brickson, K. T. Looby, and J. J. Dahl, "Beamforming and speckle reduction using neural networks," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 66, no. 5, pp. 898–910, 2019.

[38] A. K. Tehrani, M. Sharifzadeh, E. Boctor, and H. Rivaz, "Bi-directional semi-supervised training of convolutional neural networks for ultrasound elastography displacement estimation," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 4, pp. 1181–1190, 2022.

[39] X. Wei, Y. Wang, L. Ge, B. Peng, Q. He, R. Wang, L. Huang, Y. Xu, and J. Luo, "Unsupervised convolutional neural network for motion estimation in ultrasound elastography," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 7, pp. 2236–2247, 2022.

[40] A. K. Tehrani, M. Ashikuzzaman, and H. Rivaz, "Lateral strain imaging using self-supervised and physically inspired constraints in unsupervised regularized elastography," *IEEE Transactions on Medical Imaging*, 2022.

[41] E. Evain, K. Faraz, T. Grenier, D. Garcia, M. De Craene, and O. Bernard, "A pilot study on convolutional neural networks for motion estimation from ultrasound images," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 67, no. 12, pp. 2565–2573, 2020.

[42] R. Delaunay, Y. Hu, and T. Vercauteren, "An unsupervised learning approach to ultrasound strain elastography with spatio-temporal consistency," *Physics in Medicine & Biology*, vol. 66, no. 17, p. 175031, 2021.

[43] X. Liu, T. Zhou, M. Lu, Y. Yang, Q. He, and J. Luo, "Deep learning for ultrasound localization microscopy," *IEEE Trans. Med. Im.*, vol. 39, no. 10, pp. 3064–3078, 2020.

[44] R. J. van Sloun, O. Solomon, M. Bruce, Z. Z. Khaing, H. Wijkstra, Y. C. Eldar, and M. Mischi, "Super-resolution ultrasound localization microscopy through deep learning," *IEEE Trans. Med. Im.*, 2020.

[45] X. Chen, M. R. Lowerison, Z. Dong, N. V. Chandra Sekaran, C. Huang, S. Chen, T. M. Fan, D. A. Llano, and P. Song, "Localization free super-resolution microbubble velocimetry using a long short-term memory neural network," *bioRxiv*, pp. 2021–10, 2021.

[46] J. Kim, Z. Dong, M. R. Lowerison, N. V. C. Sekaran, Q. You, D. A. Llano, and P. Song, "Deep learning-based 3d beamforming on a 2d row column addressing (rca) array for 3d super-resolution ultrasound localization microscopy," in *2022 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2022, pp. 1–4.

[47] T. N. Nguyen, A. S. Podkowa, A. Y. Tam, E. C. Arnold, R. J. Miller, T. H. Park, M. N. Do, and M. L. Oelze, "Characterizing fatty liver in vivo in rabbits, using quantitative ultrasound," *Ultrasound in Medicine & Biology*, vol. 45, no. 8, pp. 2049–2062, 2019.

[48] M. Byra, P. Jarosik, K. Dobruch-Sobczak, Z. Klimonda, H. Piotrzkowska-Wroblewska, J. Litniewski, and A. Nowicki, "Joint segmentation and classification of breast masses based on ultrasound radio-frequency data and convolutional neural networks," *Ultrasonics*, p. 106682, 2022.

[49] D. C. Castro, I. Walker, and B. Glocker, "Causality matters in medical imaging," *Nature Communications*, vol. 11, no. 1, pp. 1–10, 2020.

[50] L. Alzubaidi, J. Bai, A. Al-Sabaawi, J. Santamaría, A. S. Albahri, B. S. N. Al-dabbagh, M. A. Fadhel, M. Manoufali, J. Zhang, A. H. Al-Timemy et al., "A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications," *Journal of Big Data*, vol. 10, no. 1, p. 46, 2023.

[51] B. Maschler and M. Weyrich, "Deep transfer learning for industrial automation: a review and discussion of new techniques for data-driven machine learning," *IEEE Industrial Electronics Magazine*, vol. 15, no. 2, pp. 65–75, 2021.

[52] D. Karimi, S. K. Warfield, and A. Gholipour, "Transfer learning in medical image segmentation: New insights from analysis of the dynamics of model parameters and learned representations," *Artificial intelligence in medicine*, vol. 116, p. 102078, 2021.

[53] W. Mao, J. Chen, Y. Chen, S. S. Afshari, and X. Liang, "Construction of health indicators for rotating machinery using deep transfer learning with multiscale feature representation," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[54] L. Xun, J. Zhang, F. Yao, and D. Cao, "Improved identification of cotton cultivated areas by applying instance-based transfer learning on the time series of modis ndvi," *Catena*, vol. 213, p. 106130, 2022.

[55] J. C. Hung and J.-W. Chang, "Multi-level transfer learning for improving the performance of deep neural networks: Theory and practice from the tasks of facial emotion recognition and named entity recognition," *Applied Soft Computing*, vol. 109, p. 107491, 2021.

[56] G. Michau and O. Fink, "Unsupervised transfer learning for anomaly detection: Application to complementary operating condition transfer," *Knowledge-Based Systems*, vol. 216, p. 106816, 2021.

[57] M. Wilbur, A. Mukhopadhyay, S. Vazirizade, P. Pugliese, A. Laszka, and A. Dubey, "Energy and emission prediction for mixed-vehicle transit fleets using multi-task and inductive transfer learning," in *Machine Learning and Knowledge Discovery in Databases. Applied Data Science Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part IV 21*. Springer, 2021, pp. 502–517.

[58] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[59] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *IEEE transactions on knowledge and data engineering*, vol. 35, no. 1, pp. 857–876, 2021.

[60] S. Azizi, B. Mustafa, F. Ryan, Z. Beaver, J. Freyberg, J. Deaton, A. Loh, A. Karthikesalingam, S. Kornblith, T. Chen et al., "Big self-supervised models advance medical image classification," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 3478–3488.

[61] N. E. Khalifa, M. Loey, and S. Mirjalili, "A comprehensive survey of recent trends in deep learning for digital images augmentation," *Artificial Intelligence Review*, vol. 55, no. 3, pp. 2351–2377, 2022.

[62] N. Cauli and D. Reforgiato Recupero, "Survey on videos data augmentation for deep learning models," *Future Internet*, vol. 14, no. 3, p. 93, 2022.

[63] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[64] L. Brigato and L. Iocchi, "A close look at deep learning with small data," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 2490–2497.

[65] G. Miranda and C. Rubio-Manzano, "Image classification using deep and classical machine learning on small datasets: A complete comparative," 2022.

[66] M. R. Santander, J. H. Albarracin, and A. R. Rivera, "On the pitfalls of learning with limited data: a facial expression recognition case study," *Expert Systems with Applications*, vol. 183, p. 114991, 2021.

[67] A. M Javid, "Neural network architecture design: towards low-complexity and scalable solutions," Ph.D. dissertation, KTH Royal Institute of Technology, 2021.

[68] İ. Aydın and E. Kızılay, "Development of a new light-weight convolutional neural network for acoustic-based amateur drone detection," *Applied Acoustics*, vol. 193, p. 108773, 2022.

[69] T. Kim, J. Oh, N. Kim, S. Cho, and S.-Y. Yun, "Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation," *arXiv preprint arXiv:2105.08919*, 2021.

[70] J. Qi, J. Du, S. M. Siniscalchi, X. Ma, and C.-H. Lee, "On mean absolute error for deep neural network based vector-to-vector regression," *IEEE Signal Processing Letters*, vol. 27, pp. 1485–1489, 2020.

[71] Y. Zhou, X. Wang, M. Zhang, J. Zhu, R. Zheng, and Q. Wu, "Mpce: a maximum probability based cross entropy loss function for neural network classification," *IEEE Access*, vol. 7, pp. 146 331–146 341, 2019.

[72] B. M. Ozyildirim and M. Kiran, "Levenberg–marquardt multi-classification using hinge loss function," *Neural Networks*, vol. 143, pp. 564–571, 2021.

[73] B. Barz and J. Denzler, "Deep learning on small datasets without pre-training using cosine loss," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2020, pp. 1371–1380.

[74] M. Raissi, P. Perdikaris, and G. 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*, vol. 378, pp. 686–707, 2019.

[75] R. G. Patel, I. Manickam, N. A. Trask, M. A. Wood, M. Lee, I. Tomas, and E. C. Cyr, "Thermodynamically consistent physics-informed neural networks for hyperbolic systems," *Journal of Computational Physics*, vol. 449, p. 110754, 2022.

[76] A. Kovacs, L. Exl, A. Kornell, J. Fischbacher, M. Hovorka, M. Gusenbauer, L. Breth, H. Oezelt, M. Yano, N. Sakuma et al., "Conditional physics informed neural networks," *Communications in Nonlinear Science and Numerical Simulation*, vol. 104, p. 106041, 2022.

[77] J. Bai, T. Rabczuk, A. Gupta, L. Alzubaidi, and Y. Gu, "A physics-informed neural network technique based on a modified loss function for computational 2d and 3d solid mechanics," *Computational Mechanics*, vol. 71, no. 3, pp. 543–562, 2023.

[78] Y. Song, J. Zheng, L. Lei, Z. Ni, B. Zhao, and Y. Hu, "Ct2us: Cross-modal transfer learning for kidney segmentation in ultrasound images with synthesized data," *Ultrasonics*, vol. 122, p. 106706, 2022.

[79] G. Ayana, J. Park, J.-W. Jeong, and S.-w. Choe, "A novel multistage transfer learning for ultrasound breast cancer image classification," *Diagnostics*, vol. 12, no. 1, p. 135, 2022.

[80] D. Cheng and E. Y. Lam, "Transfer learning u-net deep learning for lung ultrasound segmentation," *arXiv preprint arXiv:2110.02196*, 2021.

[81] G. Ayana, K. Dese, and S.-w. Choe, "Transfer learning in breast cancer diagnoses via ultrasound imaging," *Cancers*, vol. 13, no. 4, p. 738, 2021.

[82] E. C. Constantinescu, A.-L. Udristoiu, S. C. Udristoiu, A. V. Iacob, L. G. Gruionu, G. Gruionu, L. Săndulescu, and A. Săftoiu, "Transfer learning with pre-trained deep convolutional neural networks for the automatic assessment of liver steatosis in ultrasound images," *Medical Ultrasonography*, vol. 23, no. 2, pp. 135–139, 2021.

[83] P. F. Wilson, M. Gilany, A. Jamzad, F. Fooladgar, M. N. N. To, B. Wodlinger, P. Abolmaesumi, and P. Mousavi, "Self-supervised learning with limited labeled data for prostate cancer detection in high frequency ultrasound," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2023.

[84] Z. Xiang, Q. Zhuo, C. Zhao, X. Deng, T. Zhu, T. Wang, W. Jiang, and B. Lei, "Self-supervised multi-modal fusion network for multi-modal thyroid ultrasound image diagnosis," *Computers in Biology and Medicine*, vol. 150, p. 106164, 2022.

[85] J. Jiao, R. Droste, L. Drukker, A. T. Papageorghiou, and J. A. Noble, "Self-supervised representation learning for ultrasound video," in *2020 IEEE 17th international symposium on biomedical imaging (ISBI)*. IEEE, 2020, pp. 1847–1850.

[86] X. Dai, Y. Lei, T. Wang, M. Axente, D. Xu, P. Patel, A. B. Jani, W. J. Curran, T. Liu, and X. Yang, "Self-supervised learning for accelerated 3d high-resolution ultrasound imaging," *Medical Physics*, vol. 48, no. 7, pp. 3916–3926, 2021.

[87] A.-R. Ali, A. E. Samir, and P. Guo, "Self-supervised learning for accurate liver view classification in ultrasound images with minimal labeled data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3086–3092.

[88] S. Zama, T. Fujioka, E. Yamaga, K. Kubota, M. Mori, L. Katsuta, Y. Yashima, A. Sato, M. Kawauchi, S. Higuchi et al., "Clinical utility of breast ultrasound images synthesized by a generative adversarial network," *Medicina*, vol. 60, no. 1, p. 14, 2023.

[89] M. Mendez, S. Sundararaman, L. Probyn, and P. N. Tyrrell, "Approaches and limitations of machine learning for synthetic ultrasound generation: A scoping review," *Journal of Ultrasound in Medicine*, vol. 42, no. 12, pp. 2695–2706, 2023.

[90] S. Katakis, N. Barotsis, A. Kakotaritis, P. Tsiganos, G. Economou, E. Panagiotopoulos, and G. Panayiotakis, "Generation of musculoskeletal ultrasound images with diffusion models," *BioMedInformatics*, vol. 3, no. 2, pp. 405–421, 2023.

[91] X. Liu and M. Almekkawy, "Ultrasound computed tomography using physical-informed neural network," in *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2021, pp. 1–4.

[92] S. Alkhadhr and M. Almekkawy, "Modeling of the wave propagation of a multi-element ultrasound transducer using neural networks," in *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2021, pp. 1–4.

[93] S. Alkhadhr, X. Liu, and M. Almekkawy, "Modeling of the forward wave propagation using physics-informed neural networks," in *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2021, pp. 1–4.

[94] O. Wiles et al., "A fine-grained analysis on distribution shift," *CoRR*, vol. abs/2110.11328, 2021. [Online]. Available: https://arxiv.org/abs/2110.11328

[95] I. Gulrajani and D. Lopez-Paz, "In search of lost domain generalization," *CoRR*, vol. abs/2007.01434, 2020. [Online]. Available: https://arxiv.org/abs/2007.01434

[96] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[97] D. Hendrycks et al., "Augmix: A simple data processing method to improve robustness and uncertainty," *arXiv preprint arXiv:1912.02781*, 2020.

[98] E. D. Cubuk et al., "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 702–703.

[99] E. D. Cubuk et al., "Autoaugment: Learning augmentation strategies from data," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 113–123.

[100] K. Goel et al., "Model patching: Closing the subgroup performance gap with data augmentation," *arXiv preprint arXiv:2008.06775*, 2020.

[101] Y. Choi et al., "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8789–8797.

[102] J.-Y. Zhu et al., "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[103] J. Tierney, A. Luchies, C. Khan, B. Byram, and M. Berger, "Domain adaptation for ultrasound beamforming," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020, pp. 410–420.

[104] M. Arjovsky et al., "Invariant risk minimization," *arXiv preprint arXiv:1907.02893*, 2019.

[105] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*. Springer, 2016, pp. 443–450.

[106] Y. Ganin et al., "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[107] S. Schneider et al., "Improving robustness against common corruptions by covariate shift adaptation," *Advances in neural information processing systems*, vol. 33, pp. 11 539–11 551, 2020.

[108] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 806–813.

[109] H. Azizpour, A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 36–45.

[110] M. Amiri, R. Brooks, and H. Rivaz, "Fine-tuning u-net for ultrasound image segmentation: different layers, different outcomes," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 67, no. 12, pp. 2510–2518, 2020.

[111] A. K. Tehrani, I. M. Rosado-Mendez, and H. Rivaz, "Robust scatterer number density segmentation of ultrasound images," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 69, no. 4, pp. 1169–1180, 2022.

[112] M.-G. Kim, S. Oh, Y. Kim, H. Kwon, and H.-M. Bae, "Learning-based attenuation quantification in abdominal ultrasound," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2021, pp. 14–23.

[113] A. K. Tehrani, M. Amiri, I. M. Rosado-Mendez, T. J. Hall, and H. Rivaz, "Ultrasound scatterer density classification using convolutional neural networks and patch statistics," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 68, no. 8, pp. 2697–2706, 2021.

[114] U. Soylu and M. L. Oelze, "Calibrating data mismatches in deep learning-based quantitative ultrasound using setting transfer functions," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 70, no. 6, pp. 510–520, 2023.

[115] M. Sharifzadeh, A. K. Tehrani, H. Benali, and H. Rivaz, "Ultrasound domain adaptation using frequency domain analysis," in *2021 IEEE International Ultrasonics Symposium (IUS)*. IEEE, 2021, pp. 1–4.

[116] S. Oh, M.-G. Kim, Y. Kim, G. Jung, H. Kwon, and H.-M. Bae, "Sensor geometry generalization to untrained conditions in quantitative ultrasound imaging," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2022, pp. 780–789.

[117] A. K. Tehrani, G. Cloutier, A. Tang, I. M. Rosado-Mendez, and H. Rivaz, "Homodyned k-distribution parameter estimation in quantitative ultrasound: Autoencoder and bayesian neural network approaches," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2024.

[118] J. Liang, R. He, and T. Tan, "A comprehensive survey on test-time adaptation under distribution shifts," *arXiv preprint arXiv:2303.15361*, 2023.

[119] H. Zhang, Y. Zhang, K. Jia, and L. Zhang, "Unsupervised domain adaptation of black-box source models," *arXiv preprint arXiv:2101.02839*, 2021.

[120] V. Chandrasekaran, K. Chaudhuri, I. Giacomelli, S. Jha, and S. Yan, "Exploring connections between active learning and model extraction," in *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, Aug. 2020. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation /chandrasekaran pp. 1309–1326.

[121] H. Jia, C. A. Choquette-Choo, V. Chandrasekaran, and N. Papernot, "Entangled watermarks as a defense against model extraction," in *30th USENIX security symposium (USENIX Security 21)*, 2021, pp. 1937–1954.

[122] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia conference on computer and communications security*, 2018, pp. 159–172.

[123] Y. Nagai, Y. Uchida, S. Sakazawa, and S. Satoh, "Digital watermarking for deep neural networks," *International Journal of Multimedia Information Retrieval*, vol. 7, pp. 3–16, 2018.

[124] U. Soylu and M. L. Oelze, "A data-efficient deep learning strategy for tissue characterization via quantitative ultrasound: Zone training," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2023.

[125] U. Soylu and M. L. Oelze, "Machine-to-machine transfer function in deep learning-based quantitative ultrasound," *arXiv preprint arXiv:2311.16028*, 2023.

[126] H. Su, V. Jampani, D. Sun, O. Gallo, E. Learned-Miller, and J. Kautz, "Pixel-adaptive convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 166–11 175.

[127] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu, "Attention mechanisms in computer vision: A survey," *Computational Visual Media*, pp. 1–38, 2022.

[128] A. Adadi, "A survey on data-efficient algorithms in big data era," *Journal of Big Data*, vol. 8, no. 1, pp. 1–54, 2021.

[129] B. Shneiderman, "Human-centered ai," *Issues in Science and Technology*, vol. 37, no. 2, pp. 56–61, 2021.

[130] K. A. Wear, T. A. Stiles, G. R. Frank, E. L. Madsen, F. Cheng, E. J. Feleppa, C. S. Hall, B. S. Kim, P. Lee, W. D. O'Brien Jr et al., "Interlaboratory comparison of ultrasonic backscatter coefficient measurements from 2 to 9 mhz," *Journal of Ultrasound in Medicine*, vol. 24, no. 9, pp. 1235–1250, 2005.

[131] E. L. Madsen, G. R. Frank, and F. Dong, "Liquid or solid ultrasonically tissue-mimicking materials with very low scatter," *Ultrasound in Medicine & Biology*, vol. 24, no. 4, pp. 535–542, 1998.

[132] J. J. Anderson et al., "Interlaboratory comparison of backscatter coefficient estimates for tissue-mimicking phantoms," *Ultrasonic Imaging*, vol. 32, no. 1, pp. 48–64, 2010.

[133] E. L. Madsen, J. A. Zagzebski, R. A. Banjavie, and R. E. Jutila, "Tissue mimicking materials for ultrasound phantoms," *Medical Physics*, vol. 5, no. 5, pp. 391–394, 1978.

[134] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026–8037, 2019.

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

[136] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, and V. K. Asari, "The history began from alexnet: A comprehensive survey on deep learning approaches," *arXiv preprint arXiv:1803.01164*, 2018.

[137] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[138] J. Mamou and M. L. Oelze, *Quantitative ultrasound in soft tissues*. Springer, 2013.

[139] K. Nam, I. M. Rosado-Mendez, L. A. Wirtzfeld, V. Kumar, E. L. Madsen, G. Ghoshal, A. D. Pawlicki, M. L. Oelze, R. J. Lavarello, T. A. Bigelow et al., "Cross-imaging system comparison of backscatter coefficient estimates from a tissue-mimicking material," *The Journal of the Acoustical Society of America*, vol. 132, no. 3, pp. 1319–1324, 2012.

[140] E. L. Madsen, M. A. Hobson, H. Shi, T. Varghese, and G. R. Frank, "Stability of heterogeneous elastography phantoms made from oil dispersions in aqueous gels," *Ultrasound in medicine & biology*, vol. 32, no. 2, pp. 261–270, 2006.

[141] U. Soylu and M. L. Oelze, "A data-efficient deep learning training strategy for biomedical ultrasound imaging: Zone training," *arXiv preprint arXiv:2202.00547*, 2022.

[142] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[143] J. S. Lim and A. V. Oppenheim, "Enhancement and bandwidth compression of noisy speech," *Proceedings of the IEEE*, vol. 67, no. 12, pp. 1586–1604, 1979.

[144] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[145] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825*, 2017.

[146] I. A. Hein and W. D. O'Brien, "Current time-domain methods for assessing tissue motion by analysis from reflected ultrasound echoes-a review," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 40, no. 2, pp. 84–102, 1993.

[147] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognition*, vol. 80, pp. 109–117, 2018.