Using Convolutional Neural Network for Early Breast Cancer Detection

By

Anna Rusnak

B.S. University of New Orleans, 2020

Thesis

Submitted in partial fulfillment of the requirements for the Degree of Master of Science in

Biomedical Engineering at Brown University

PROVIDENCE, RHODE ISLAND

MAY 2022

AUTHORIZATION TO LEND AND REPRODUCE THE THESIS

As the sole author of this thesis, I authorize Brown University to lend it to other institutions for the purpose of scholarly research.

Date 4/22/22

Anna Rusnak

Anna Rusnak, Author

I further authorize Brown University to reproduce this thesis by photocopying or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Date 4/22/22

Anna Rusnak

Anna Rusnak, Author

This thesis by Anna Rusnak is accepted in its present form by the Center of Biomedical Engineering as satisfying the thesis requirement for the degree of Master of Science.

Date _____ Signature:_____

Dr. Vikas Srivastava, Advisor

Recommended to the Graduate Council

Date _____ Signature:_____

Dr. Anubav Tripathi, Reader

Date _____ Signature:_____

Dr. Michelle Dawson, Reader

Approved by the Graduate Council

Date _____ Signature:_____

Dr. Andrew G. Campbell,

Dean of the Graduate School

Acknowledgements

I would like to thank the members of my research group for their support. In particular, Sijun Niu, who greatly helped me with developing my model and ultrasound imaging. Ruth Sullivan for creating PDMS ultrasound phantoms for experimental validation. Zahra Ahmed for her troubleshooting skills and polymer expertise. My adviser Dr. Vikas Srivastava for excellent mentorship and guidance. And finally, a radiologist Dr. Hemali Desai for providing invaluable consultation on the state-of-the-art breast cancer imaging techniques. Special thanks to Dr. Marissa Gray for being informative and accommodating program director and Dr. Michelle Dawson and Dr. Anubhav Tripathi for serving as members of my committee.

Table of Contents

Acknowledgments **4**

Table of Figures **6**

Table of Tables **7**

Abstract **8**

1 Introduction **10**

1.1 Breast Cancer 10

1.2 Breast Cancer Diagnostics 11

1.3 Machine Learning 14

1.4 Conclusion 17

2 Materials and Methods **19**

2.1 Abaqus Model 19

2.1 Convolutional Neural Network 23

2.1 Experimental Validation 26

3 Results and Discussion **29**

3 Abaqus Model 29

3 Experimental Validation 32

4 Conclusion and Future Work **38**

5 References **39**

6 Appendix A: Abaqus Simulation Code **46**

7 Appendix B: CNN Code **64**

Table of Figures

Introduction

1 Kaplan-Meier curves of breast cancer survival in relation to tumor size. . . . 12

2 A. Mammogram of a breast abnormality. B. Ultrasound of the same breast abnormality. . . . 13

3 The relationship between artificial intelligence, machine learning, and deep learning. . . . 14

4 A general architecture of a convolutional neural network. . . . 16

Materials and Methods

5 An overview of the computational and experimental workflows. . . . 19

6 Size and corresponding elastic modulus of simulated tumors. . . . 20

7 2D axisymmetric model of a breast with a tumor created in Abaqus. . . . 21

8 Cosine function of an ultrasound signal used in the simulation. . . . 22

9 The ultrasound output of an Abaqus simulation showing an initial pulse and signal reflected off the tumor. . . . 22

10 The architecture of our neural network. . . . 24

11 The aluminum molds used to create PDMS phantoms. . . . 26

12 PDMS breast with a 2 cm tumor inside. . . . 28

13 Applying Olympus 1 ultrasound machine to the PDMS breast. . . . 28

Results and Discussion

14 Ultrasound wave propagation in Abaqus simulation. . . . 29

15-19 Tumors simulation signals. 29-30

20 Training and testing loss over a 1,000 epochs. 31

21 Actual vs predicted size. 31

22 Actual vs predicted elastic modulus. 31

23-41 Tumors experimental signals. 33-37

Table of Tables

Introduction

1 Description of TNM classification based on tumor size, lymph node involvement and metastases. 11

2 Description of TNM cancer stage classification. 11

3 Comparison and contrast of ultrasound and mammography. 13

Materials and Methods

4 Description of our CNN layer-by-layer. 23

5 Elastic moduli of different Sylgard 527 to Sylgard 184 combinations 27

6 Sizes and correlated elastic moduli of PDMS tumors 28

Results and Discussion

7 Training loss over 1,000 epochs. 31

8 MAPE for size and elastic modulus for test and train data. 32

Abstract of Using Convolutional Neural Network for Breast Cancer Detection

by Anna Rusnak, ScM, Brown University, May 2022.

Objective: The objective of this study was to create a convolutional neural network (CNN) capable of predicting the size, elastic modulus, and the location of flaws embedded in soft materials. One potential application for this methodology is early breast cancer detection. This would make breast cancer screening and diagnostic safe for those who cannot undergo ionizing radiation and more accessible to those who do not have access to mammography.

Methods: Simulations of a breast containing a tumor of various elastic moduli and sizes were created. A pressure load simulating an ultrasound signal emitted from a probe was applied to the outside surface of the breast. The resulting signal, along with known elastic modulus and size of the tumor, was used to train a convolutional neural network. Convolutional neural networks consist of convolution layers where the input data is being transform by element-wise multiplication with convolution kernel, pooling layers that down sample the resulting data, and fully connected (FC) layers that connect the input and output. To experimentally validate the model, tissue mimicking phantoms will be created and imaged using an Olympus-1 ultrasound machine.

Results: MAPE for the training data set was 0.733 and 5.66 for size and elastic modulus, respectively. MAPE for the testing data set was 1.11 and 7.83 for size and elastic modulus, respectively.

Conclusions: The convolutional neural network trained on simulation data was evaluated to be highly effective and accurate. Future work is required to experimentally validate

and expand the model to be able to characterize smaller tumors and tumor density.

Introduction

Breast Cancer

Breast cancer is the prevailing type of cancer in women worldwide [1]. According to the American Cancer Society, 1 in 8 females is diagnosed with breast cancer. Moreover, the prediction is for breast cancer to reach 3.2 million new cases per year by 2050 [2]. While being the most frequent type of cancer in women, it is also the second deadliest [3]. American Cancer Society reports that for women under 50, breast cancer death rates have not gone down since 2007. In women over 50, breast cancer rate has been decreasing by 1% per year. This decrease is contributed to not only to better treatments but also to improved diagnostics [4].

TNM staging system is used to describe the progression of breast cancer. TNM stands for Tumor, Node, and Metastasis. Doctors evaluate the size of the tumor, if cancer has invaded nearby lymph nodes, and if cancer has become metastatic. Tumor (T) is broken down into T0, T1, T2, T3, and T4. T0 describes no evidence of tumor, T1 is a tumor of 20 mm or smaller, T2 is a tumor between 20 mm and 50 mmm, T3 is a tumor larger than 50 mm, and finally T4 describes a tumor that has grown into chest wall, skin, or both. Node (N) and metastasis (M) are classified similarly. The details of TNM classification are described in Table 1. Breast cancer is classified further, combining tumor, node, and metastasis characterization, into five stages: non-invasive stage 0, and invasive stages I-IV [5]. Table 2 describes the classification of cancer stages based on the TNM system. Stages 0 through IIA are described as early breast cancer.

T	T1	T2	T3	T4
	< 20 mm	20 mm - 50 mm	> 50 mm	Extension into skin or chest wall
N	N0	N1	N2	N3
	No lymph node metastases	Regional movable metastases	Non-movable regional metastases	Internal mammary lymph nodes involvement
M	M0	M1		
	No distant metastases	Distant metastases		

Table 1: Description of TNM classification based on tumor size, lymph node involvement and metastases.

Stage	T Tumor size	N Lymph node	M Metastasis
Stage 0	Tis	N0	M0
Stage IA	T1	N0	M0
Stage IB	T0-T1	N1mi	M0
Stage IIA	T0-T1	N1	M0
	T2	N0	M0
Stage IIB	T2-T3	N0-N1	M0
Stage IIIA	T0-T3	N2	M0
	T3	N1	M0
Stage IIIB	T4	N0-N2	M0
Stage IIIC	Any size	N3	M0
Stage IV	Any size	Any involvement	M1

Table 2: Description of TNM cancer stage classification.

Breast Cancer Diagnostics

Mammography is the state-of-the-art technology for breast cancer screening and diagnostics [6]. Mammography is a low-energy X-ray of the breast, where the breast is compressed between two plates for imaging [7]. It is fast and not complicated to read. However, there are certain shortcomings. Mammography is less sensitive in dense breasts, patients under 40 years of age, premenopausal women, and it has relatively high false-positive and false-negative rates. In fact, mammography alone did not have a drastic effect on breast cancer death rates, reducing it only by 0.0004%. Exposure to ionizing radiation can be harmful, and mammography does not provide any information on the disease outcome. Last, mammography is not widely accessible and affordable [8]. Indeed, less developed countries (LDCs) have

higher breast cancer death rate and mortality-to-incidence (M/I) ratio. In 2012, 62.1% of all breast cancer deaths occurred in LDCs. M/I ratio in LDCs was 0.37, in contrast to that of 0.2 in more developed countries. One reason for this phenomenon is failure to detect breast cancer at earlier stages [9].

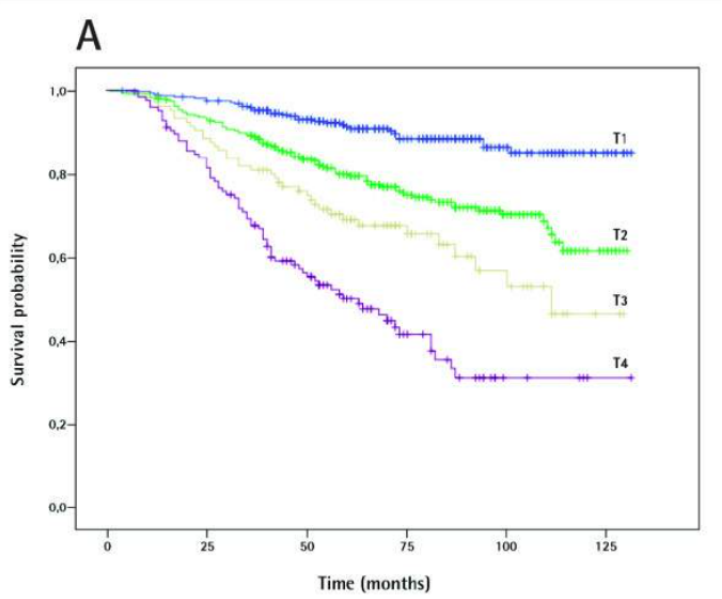

Figure 1: Kaplan-Meier curves of breast cancer survival in relation to tumor size [10].

As shown by Balabram et al. in Figure 1, the chances of patient survival decrease significantly as tumor grows in size [10]. Thus, there is a apparent need to improve breast cancer detection at the earlier stages of tumor development.

Some of the other not so popular diagnostic tools are MRI, CT, and PET scans. These tools are not widely used because of complicated procedures, ionizing radiation, and lack of cost effectiveness. On the other hand, ultrasound is cost-effective, accessible, and does not involve ionizing radiation. Ultrasound device contains a piezoelectric element that converts electrical signal to mechanical vibration and vice versa [11]. Ultrasound has an A-mode and a B-mode. In A-mode, a single transducer sends a signal and the received echoes are plotted on screen as a function of depth. In B-mode a linear array of transducers simultaneously scans a plane that can be viewed as a 2D image [12]. However, some of the limitations of ultrasound are the need of an experienced radiologist to read the results and low sensitivity. Table 3 summarizes the specification and advantages and disadvantages of ultrasound [8].

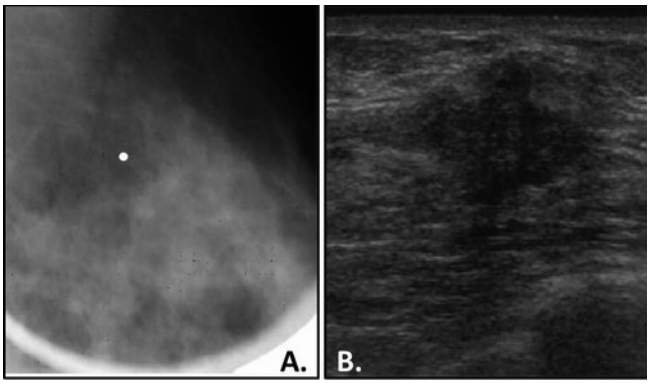

Figure 2: A. Mammogram of a breast abnormality. B. Ultrasound of the same breast abnormality. [13].

Figure 2 shows the mammography and an ultrasound of a palpable breast lump. No abnormality is detected on a mammogram, while solid mass with an irregular shape was detected on the ultrasound [13]. This image highlights the differences in images produced by mammography and ultrasound.

	Sensitivity	Specificity	Advantages	Disadvantages
Ultrasound	83.0%	34.0%	Accessible and affordable No ionizing radiation Comfortable examination	Low sensitivity Requires an experienced sonographer
Mammography	67.8%	75.0%	Fast, high specificity	Ionizing radiation Costly and not widely accessible

Table 3: Comparison and contrast of ultrasound and mammography [8].

In recent years, there has been a rise in development of portable ultrasound devices (PUD). PUDs are time-efficient, easy to use in remote locations and doctor's offices, and they can be used for various diagnostic purposes [14], including breast cancer. Study by Sood et al. revealed that ultrasound has a high potential to become a primary imaging modality for early breast cancer detection in places where mammography is not widely accessible [15]. However, PUDs have a lower resolution and still require an experienced examiner [14].

Machine Learning

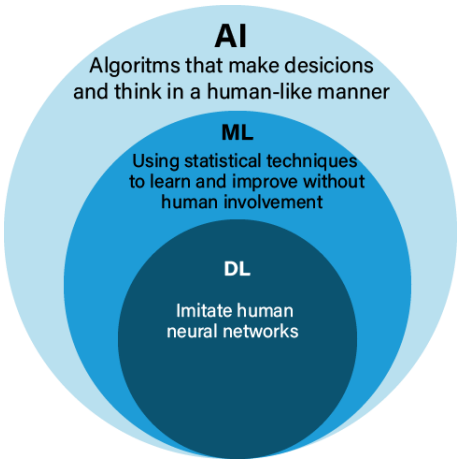

Figure 3: The relationship between artificial intelligence, machine learning, and deep learning.

We offer a machine learning (ML) solution that could possibly circumvent this ultrasound limitation, which would make breast cancer screening more accessible in locations with no access to mammography or for patients who cannot undergo ionizing radiation. As tumors grow, the amount of cancer cells, stromal cells, and extracellular matrix constituents increases. This causes tumor to stiffen [16]. Since sound velocity in solids depends on the elastic modulus, the acoustic properties of a tumor tissue would be different from that of healthy tissue [17]. Thus, the ultrasound signal should differ considerably between tumors of different elastic moduli and sizes. We used this property to develop a convolutional neural network (CNN) that would identify the presence of an early stage tumor in a range between 1 to 2 cm, and also its size and stiffness based on an input ultrasound signal. Further, tumor stiffness has been linked to cancer recurrence and metastasis. In study by Fenner at el. mice with more stiff tumors had minimal metastases compared to mice with the the least stiff tumors [18]. Considering mammography not providing any input on the eventual disease outcome, this can be another advantage of using CNN enhanced ultrasound screening for early breast cancer detection.

Machine learning is a subset of artificial intelligence (AI) algorithms that use statistical

techniques to learn and improve from data without human involvement. ML can be used in diagnostic radiology, pathology, ophthalmology, cardiology, and many other disciplines. According to the research firm Frost&Sullivan, AI has the potential to improve patient outcomes by 30-40% and reduce the health care costs by 50% [19]. The artificial intelligence in healthcare market size was valued at USD 10.4 billion in 2021 is predicted to expand by 38.4% from 2022 to 2030 [20]. Given this, there is a lot of interest of using AI in medical imaging, including ultrasound. Deep learning (DL), a subset of machine learning, is extremely effective in image-recognition tasks. The accuracy of radiology diagnoses can be improved by incorporating DL into the diagnostic process [21]. However, medical images have traits that make them difficult to analyze, such as imbalanced data sets, noisy labels, numerous disease patterns, and many more [22]. To circumvent this, we focus not on the resulting ultrasound image (B-mode), but on the raw ultrasound signal (A-mode).

The architecture of the convolutional neural network has been inspired by that of a visual cortex in the brain. In CNN, the data is passed from layer to layer, being transformed along the way, much like a signal is being passed from neuron to neuron. CNN consists of an input layer, multiple hidden layers which include convolution layers and pooling layers, fully connected layer, and an output layer. Convolution layers perform feature extraction, pooling layers downsample the extracted features, and fully connected layer maps the features to the output layer, which is classification or a value prediction. The architecture of a CNN is shown in Figure 4.

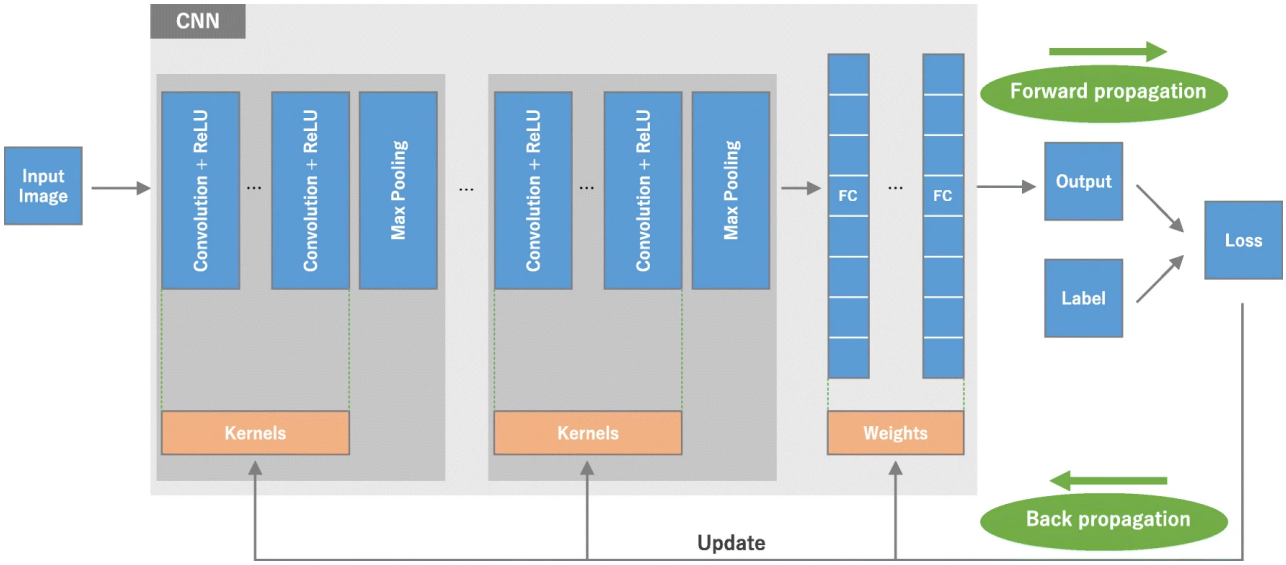

Figure 4: A general architecture of a convolutional neural network [23].

First, the input image is represented as an array of numbers. Kernel, also an array of numbers is applied over the input array in steps. Sum of element-wise multiplication of kernel and input is computed. This is the output of the convolution layer, called a feature map. Pooling layer is averaging the numbers under the pooling filter or taking their maximum. Between the hidden layers of a neural network, the data is passed through an activation function to introduce non-linearity into the model. After the data is passed through the hidden layers, it is flattened and passed to fully connected (FC) layers. In FC layers, every output is connected to every output, with an assigned weight. After the forward pass of data through the neural network, the difference between the prediction and the ground truth, referred to as loss, is computed. The kernel values and the weights are then updated in a back propagation step to minimize the loss. One forward-backward pass is called an epoch. Since kernel is applied to each image position, CNN is highly effective at recognizing features that can occur anywhere in the image [23]. The reflection of the tumor can occur in different

positions on the ultrasound signal, depending on the tumor size and material properties. Therefore, CNN is a fitting tool that can help accurately detect and characterize a tumor.

Conclusion

Less developed countries have higher incidence of breast cancer death due to cancer being diagnosed at later stages. Mammography, the state-of-the-art technique, is costly, involves ionizing radiation, and is not accessible globally. Ultrasound is more accessible and affordable than mammography and does not involve ionizing radiation. However, ultrasound needs an experienced radiologist to analyze the image and make a diagnosis. Additionally, the image resolution of PUDs, meant to make ultrasound more accessible, is lower, which complicates ultrasound image analysis further. Artificial intelligence comes into play, with a prognosis to have a generous impact on medicine. We can take advantage of the fact that tumor tissue has different material properties than healthy tissue to train CNN to detect and describe abnormalities. Analyzing the A-mode signal instead of a B-mode image lessens the need for experienced radiologist as well as alleviates the issue of low resolution with PUDs. Also, converting the signal into an image requires post processing, during which some information about tumor characteristics can be lost. The ultrasound raw time signal based analysis avoids any dependence on human generated data processing.Additionally, the reported elastic modulus of a tumor could aid in creating cancer prognosis, which gives it another advantage over mammography. This would allow ultrasound, the more affordable and accessible technology, to be more widely used for breast cancer screenings and diagnosis.

Moreover, this proposed methodology does not have to stay limited to breast tumor detection and characterization. There are a lot of cases where the need to characterize the

size and the stiffness of an inclusion arises. Other types of cancer, crystal build-ups in kidneys and bladder, ovarian cysts, and many others come to mind. Thus, this technology would also be beneficial for several other medical and biomedical fields.

Materials and Methods

The computational part of our experiment involves training and testing a CNN on ultrasound signals generated in finite element analysis software Abaqus from tumors of knows elastic moduli and sizes, shown in Figure 5 top. The experimental part consists of creating polydimethylsiloxane (PDMS) breasts with tumors of known elastic moduli and sizes and using them to test the validity of our model's predictions in real-life setting, as shown in Figure 5 bottom.

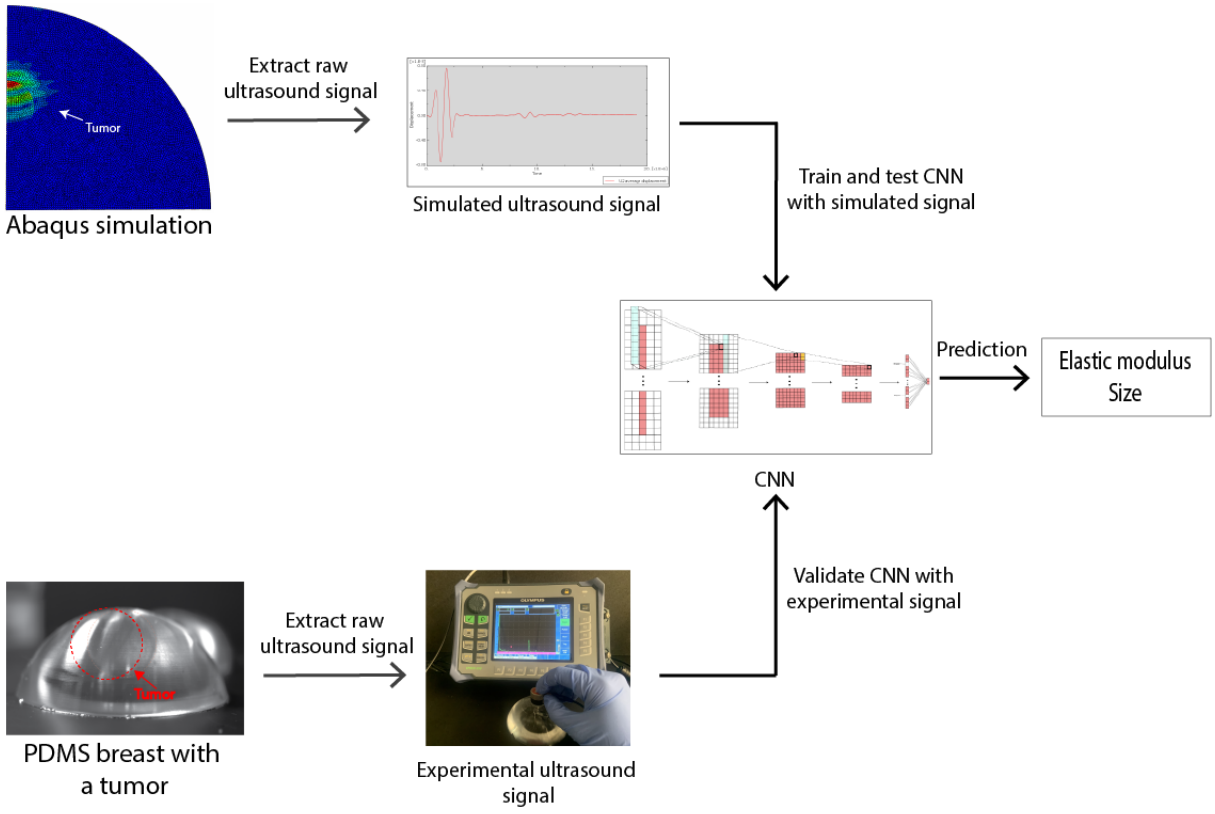

Figure 5: An overview of the computational and experimental workflows.

Abaqus Model

We created a 1,000 standard explicit models of various sizes and material properties in finite element analysis software Abaqus for training the CNN. Elastic modulus of tumors ranged from 0.1 MPa to 2MPa, and the size ranged from 10 mm to 20 mm. This tumor size range is classified as T1c. T1a and T1b, which describe tumors of size 0.1 - 10 mm, and T2, which describes tumors larger than 20 mm but smaller than 50 mm, will be included in the future work. This would cover the size range of early breast cancer tumors. According to a study by Samani, Judit and Plewes, breast tumor tissue can have elastic modulus from approximately 3 to 13 times higher than normal breast tissue [24]. In our simulation, we use tumor tissue with 1 to 20 times higher elastic modulus than breast tissue. Figure 6 shows size and corresponding elastic modulus of a 1,000 of randomly generated points used to train the neural network.

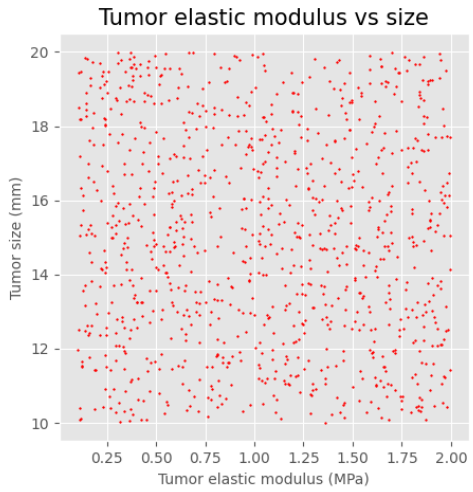

Figure 6: Size and corresponding elastic modulus of simulated tumors.

The breast was modeled as a hemisphere of 8 cm in diameter with a spherical tumor in the center of the breast. We used 2D axisymmetric model for computational efficiency. While an 8 cm in diameter 3D model with a mesh size of 0.4 mm uses 357,028 elements, an identical 2D axisymmetric model uses only 8,874 elements. Figure 7 shows one of the models created in Abaqus of a breast containing a tumor. The x-axis of the breast was assigned boundary conditions U1=0, U2=0, UR3=0 to imitate the breast being fixed to the rib cage. Y-axis was assigned to be an axis of symmetry, with boundary con-

ditions U1=0 and UR3=0. This way, when rotated about the axis of symmetry, the model would create a hemispherical breast with a spherical tumor inside. A partition of a breast surface from (0, 4, 0) to (6.25, 39.5, 0) served as an ultrasound probe 1.2 cm in size. Linear elastic isotropic material was used both for breast and the tumor. The elastic modulus of a breast was constant at 0.1 MPa, with a Poisson's ratio of 0.499 for both breast and the tumor. We used the density of 0.97 kg/m^3 for both.

Ultrasound was simulated as a pressure wave of 1 MHz frequency applied on the ultrasound probe partition of the breast surface. We used the formula for the speed of sound in three-dimensional solids to calculate the speed of sound in the breast:

$$c = \sqrt{\frac{E(1-\upsilon)}{\rho(1+\upsilon)(1-2\upsilon)}} \qquad (1)$$

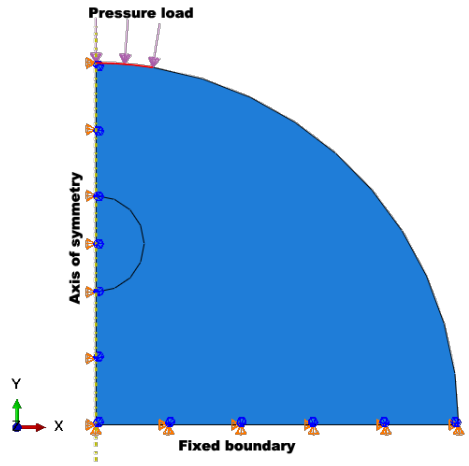

Figure 7: 2D axisymmetric model of a breast with a tumor created in Abaqus.

Where E is the elastic modulus of the material, υ is the Poisson's ratio, and ρ is the density. Using equation 1, the speed of sound propagating through the breast was 4,151 m/s. Mesh size was calculated using the formula to accommodate 10 elements per wavelength. Since the speed and the frequency of the sound were known, trivial calculations were performed to calculate the wavelength, and subsequently the mesh size. The mesh size of 0.415 mm was used. Quad element shape with free technique and advancing front algorithm were assigned. Element type CAX4R from the explicit element library was chosen.

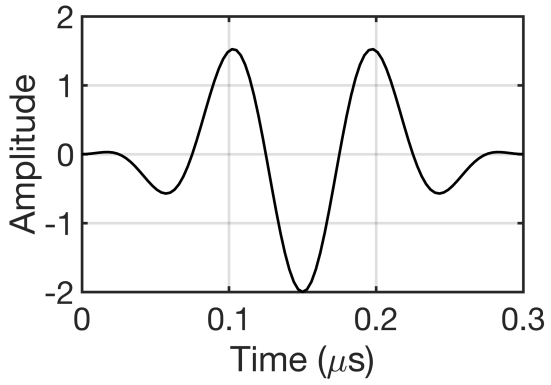

Figure 8: Cosine function of an ultrasound signal used in the simulation.

1 MHz raised-cosine pressure wave was used to simulate an ultrasound signal. This amplitude of this wave is described by equation [25]

$$A = \begin{cases} \cos(2\pi ft)[1 - \frac{2\pi ft}{m}], & 0 \leq t \leq \frac{m}{f} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The raised-cosine pressure wave used in the simulation is shown in the Figure 8. As mentioned before, a pressure load of 0.01 magnitude on the ultrasound probe region was applied to simulate the ultrasound wave. Dynamic explicit step was used. Time increment had to be small enough to capture the smallest natural period of the wave and was chosen to be 0.005 μs. Total time of the simulation was that for signal to travel through the breast and return to the receiver, 19.3 μs.

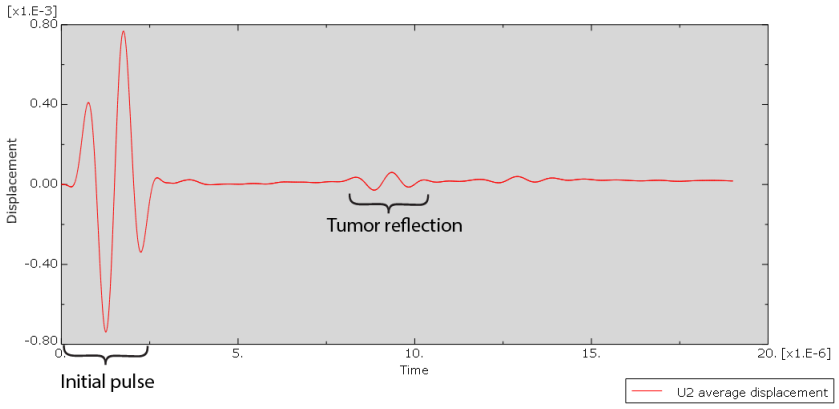

Figure 9: The ultrasound output of an Abaqus simulation showing an initial pulse and signal reflected off the tumor.

History output of a probe displacement in the y-direction, defined in Figure 7, was extracted. The average displacement of a probe node results from a returning ultrasound signal, which depends on the material properties of a tumor. Figure 9 shows the displacement of the probe nodes as the ultrasound signal is applied (initial pulse) and as it is reflected off of the tumor and returns to the probe (tumor reflection). Thus, this displacement is correlated to the elastic modulus and size of the tumor and can be used to train a neural network.

Convolutional Neural Network

The data was pre-processed for the neural network using Python: the initial pulse was removed, and the amplitude of the signal was normalized between -1 and 1 by dividing by the maximum value. The elastic modulus and the size, or labels, were normalized between 0 and 1. The details of our CNN are shown in Table 4 below.

The formula for min-max normalization, which normalizes values to between 0 and 1 is given by the formula:

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{3}$$

Where z_i is the normalized value in the dataset, and x_i is the value in the dataset.

Layer	Data in	Channel in	Channel out	Padding	Kernel	Stride	Dropout	Data out
Convolution 1	1×3296	1	4	2	8	4	-	4×824
Convolution 2	4×824	4	8	2	8	4	-	8×206
Max Pooling	8×206	8	8	0	2	2	-	8×103
Fully Connected Layer	8×103	8	1	-	-	-	0.2	1×824
Output	1×824	-	-	-	-	-	-	1× 2

Table 4: Description of our CNN layer-by-layer.

PyTorch was used to build a neural network. Rectified linear unit (ReLU) activation function was used for both convolution layers. ReLU is the most popular activation function

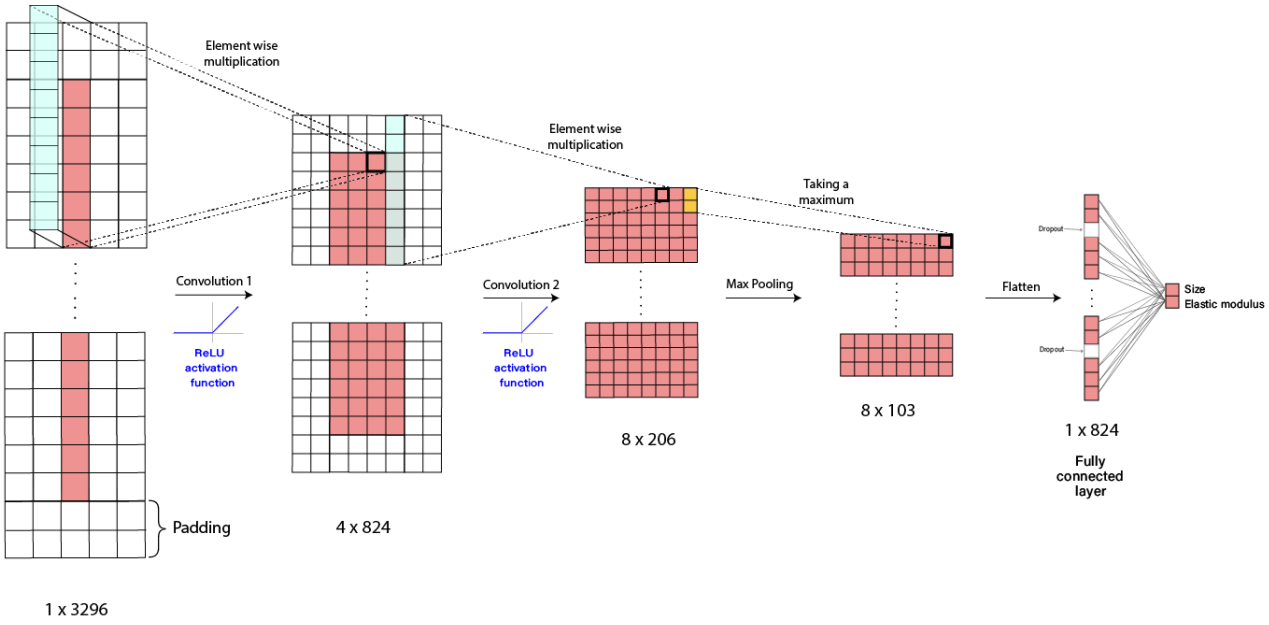

Figure 10: The architecture of our neural network.

in neural networks [26]. On account of not all neurons being activated at the same time, ReLU is more efficient than other functions [27] It is given by the formula:

$$ReLU(x) = max(0, x) \tag{4}$$

Max pooling operation with a stride of 2 and no padding was used to downsample the output of convolutional layers. The output was flattened with a 0.2 dropout, activated by ReLU function, and passed to a fully connected layer (FC). FC in turns passed the data to an output layer consisting of two neurons, which is the prediction of elastic modulus and size. Figure 10 shows the architecture of our neural network.

The model was iterated for 1000 epochs, with a 0.001 learning rate. Adaptive moment estimator optimizer (Adam) algorithm with a mean squared error (MSE) loss function was used to train the model. MSE is given by equation 5:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y})^2 \tag{5}$$

Where N is the number of data points, Y_i are the observed values and $\hat{Y}_i$ are the predicted

values.

Adam optimizer combines the advantages of RMSProp and Stochastic Gradient Descent with momentum. It is computationally efficient, requires little memory, and is well-suited for data sets with noisy gradients. Adam is an adaptive learning rate optimizer, meaning it assigns different learning rates to different parameters. It uses estimation of first and second moments of gradient to compute the learning rate for each weight [28]. The moving averages are estimated using the formulas:

$$m_t = (1 - \beta_1) \sum_{i=0}^{t} \beta_1^{t-i} g_i \tag{6}$$

and

$$v_t = (1 - \beta_2) \sum_{i=1}^{t} \beta_2^{t-i} g_i^2 \tag{7}$$

Where β is the hyper-parameters, and g is the gradient on a current mini-batch. The formulas for bias-corrected first moment and second raw moment estimates are:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{8}$$

and

$$\hat{v_t} = \frac{v_t}{1 - \beta_2^t} \tag{9}$$

Lastly, the parameters are updated using the formula [28]:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v_t} + \epsilon}} \tag{10}$$

The performance of the model was evaluated using mean absolute error (MAE) and mean

absolute percent error (MAPE), which are given by:

$$MAE = \frac{\sum_{i=1}^{N} |y_i - x_i|}{N} \tag{11}$$

and

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} \frac{|y_i - x_i|}{y_i} \tag{12}$$

where N is the total number of data points, y_i is the actual value, and x_i is the predicted value.

Experimental Validation

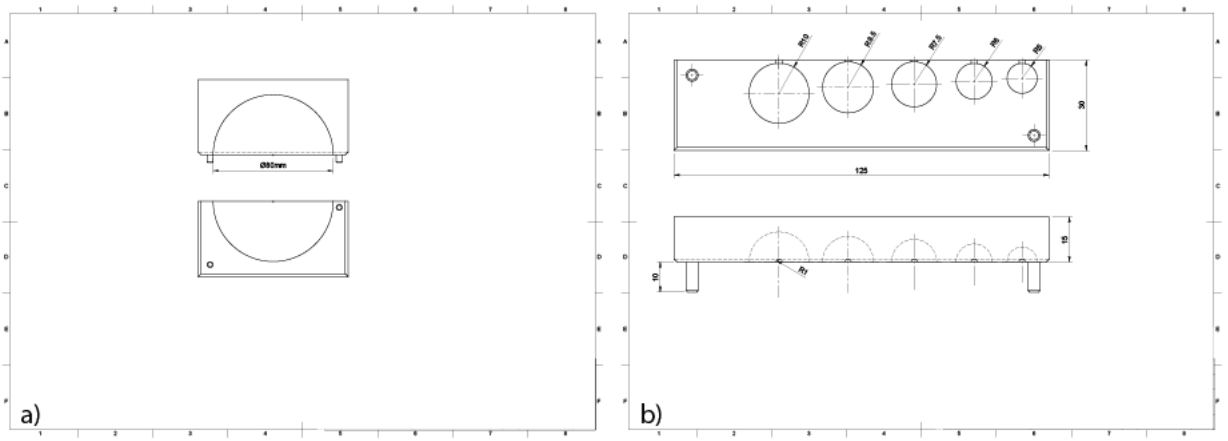

Figure 11: The aluminum molds used to create PDMS phantoms. a) breast mold, b) tumor molds

The reflection and scattering of ultrasound depend on the difference in acoustic impedance of between the tissues. The acoustic impedance of a material, in its turn, depends on its density and bulk and elastic modulus [29] [30]. Polydimethylsiloxane (PDMS) is widely used in biomedical science due to its broad applicability and versatility [31] [32]. It is chemically

inert, optically transparent, thermally stable, low cost and easy to fabricate. It also has proved to perform well in studies involving ultrasound [33]. Additionally, elastic modulus of PDMS can be fine-tuned from 5kPa to 10 MPa [34] [32]. This makes PDMS an ideal material for our application.

We used SYLGARD™ 184 Silicone Elastomer Kit and SYLGARD™ 527 A&B Silicone Dielectric Gel and aluminum molds, shown in Figure 11, coated in Rain-X to create breast and tumor phantoms. SYLGARD™ 184 and curing agent were mixed in a 10:1 ratio. Then, SYLGARD™ 527 was added to SYLGARD™ 184 mixture in different proportions and cured for 24 hours at 60°C. Table 5 shows the ratio of SYLGARD™ 527 to SYLGARD™ 184 and the resulting elastic moduli, provided by fellow graduate student Gavin Mays.

Sylgard 527:184	E trial 1 (kPa)	E trial 2 (kPa)	E trial 3 (kPa)	E average (kPa)	σ
0:1	1541.87	2113.14	2484.97	2299.05	262.93
0.2:1	1599.14	1804.11	2793.75	1701.63	144.93
1:1	704.64	379.02	855.83	780.24	106.91
4:1	216.34	197.46	191.20	201.67	13.09
5:1	120.63	97.96	121.45	113.35	13.33

Table 5: Elastic moduli of different SYLGARD™ 527 to SYLGARD™ 184 combinations

Ten PDMS breasts with an elastic modulus of 0.1 MPA were created for experimental validation of the model. One of the breasts with a tumor is pictured in Figure 12, being imaged with ultrasound in Figure 13. Olympus 1 ultrasonic transducer was used to ultrasound a breast. The raw ultrasound signal was extracted and analyzed. The elastic moduli and sizes of created PDMS tumors are shown in Table 6.

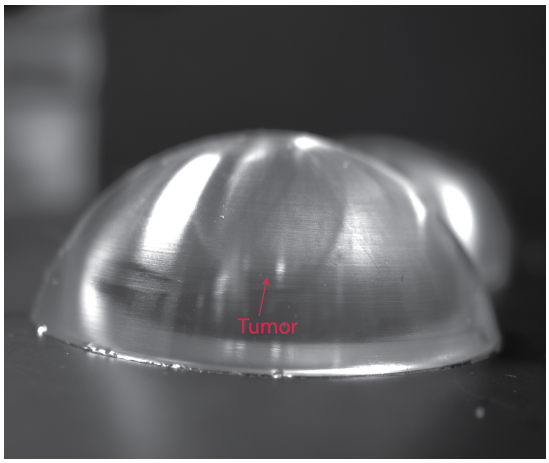

Figure 12: PDMS breast with a 2 cm tumor inside.

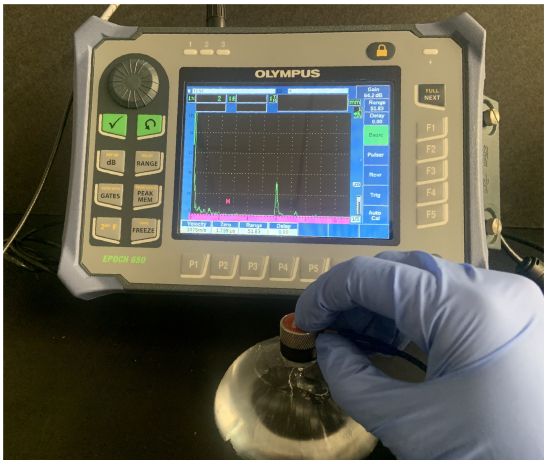

Figure 13: Applying Olympus 1 ultrasound machine to the PDMS breast.

Tumor size	Tumor elastic modulus
No tumor	
10 mm	133 kPa
10 mm	780 kPa
12 mm	202 kPA
12 mm	780 kPa
15 mm	780 kPa
15 mm	780 kPa
17 mm	780 kPa
17 mm	1702 kPa
20 mm	2299 kPa
20 mm	2299 kPa

Table 6: Sizes and correlated elastic moduli of PDMS tumors.

Results and Discussion

Abaqus Model

Figure 14 shows ultrasound wave propagation in Abaqus simulation. In Figure 14 a) the initial pulse is shown; the signal is going though the tumor in Figure 14 b), and the signal reflects off the edge of the breast in Figure 14 c).

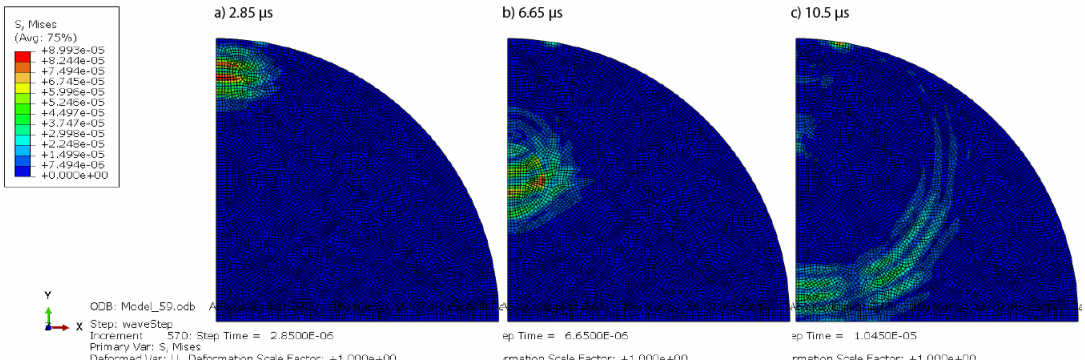

Figure 14: Ultrasound wave propagation in Abaqus simulation at a) 2.85 μs, b)6.65 μs, and c) 10.5 μs

Figures 17-21 show the Abaqus simulation signal of tumors with material properties close to those of PDMS tumors we molded.

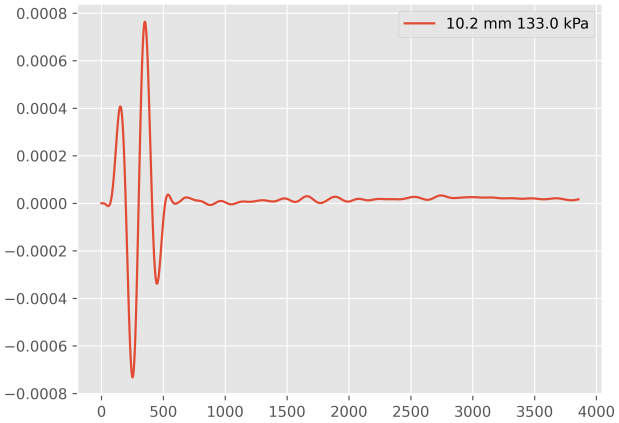

Figure 15: 10.2 mm 133 kPa tumor simulation signal.

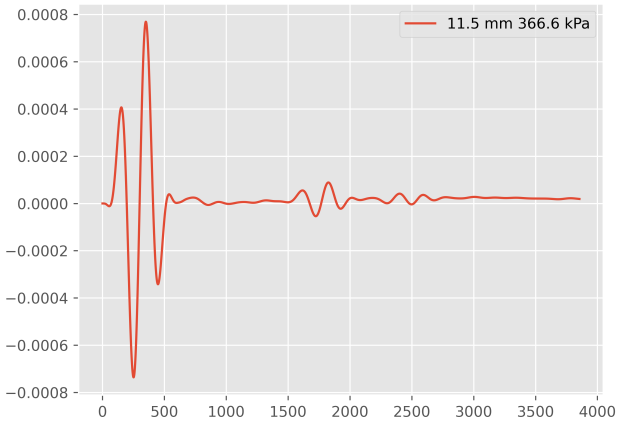

Figure 16: 11.5 mm 366.6 kPa tumor simulation signal.

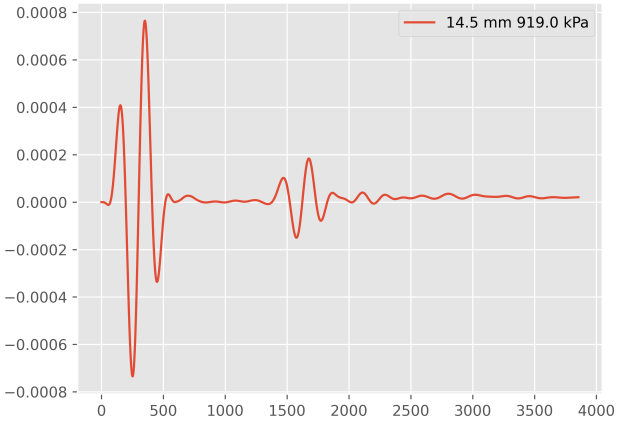

Figure 17: 14.5 mm 919 kPa tumor simulation signal.

Figure 18: 17.5 mm 1562.1 kPa tumor simulation signal.

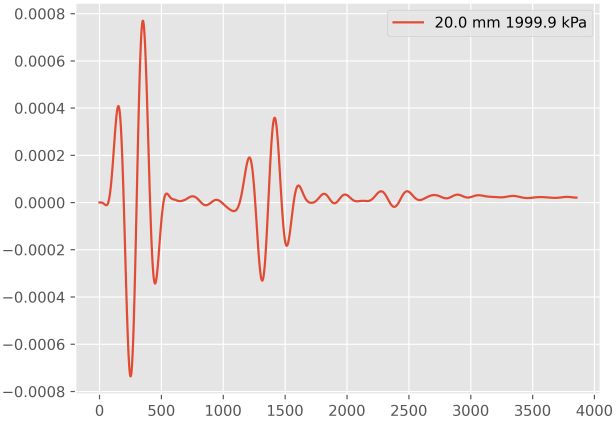

Figure 19: 20 mm 1999.9 kPa tumor simulation signal.

From the Abaqus simulation signal, we expected a trend where an amplitude of a signal reflected from a tumor is increasing as the elastic modulus of the tumor increases, and the signal is shifting closer to the initial pulse as the tumor size increases. This can be explained by the fact that inside the simulation, the signal is more readily reflected by tumors with higher elastic moduli, and shorter travel distance from larger tumors makes the signal return to the transducer faster.

The MSE loss over a 1,000 epochs in training and testing data sets is shown in Figure 15.

Table 7 shows training loss every 100 epochs over a 1,000 epochs. Where training set is 667 randomly selected Abaqus simulations, and testing set is remaining 333 Abaqus simulations.

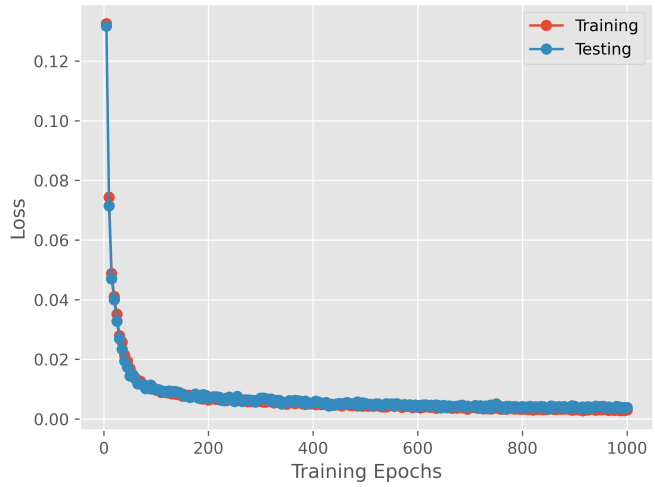

Figure 20: Training and testing loss over a 1,000 epochs.

Epoch	Loss
100	0.0099
200	0.0064
300	0.0059
400	0.0053
500	0.0046
600	0.0043
700	0.0039
800	0.0033
900	0.0031
1000	0.0030

Table 7: Training loss over 1,000 epochs.

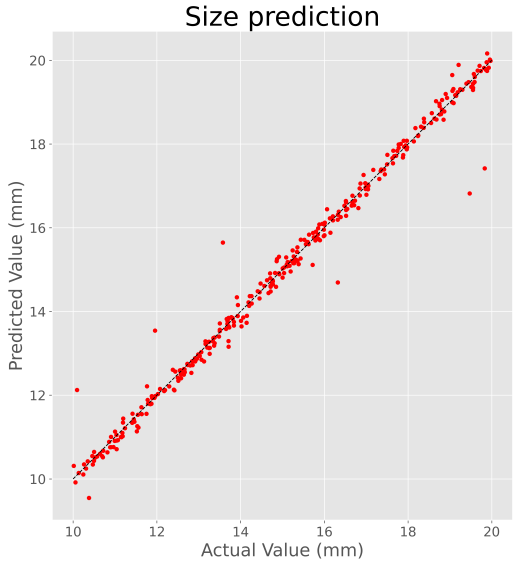

Figure 21: Actual vs predicted size.

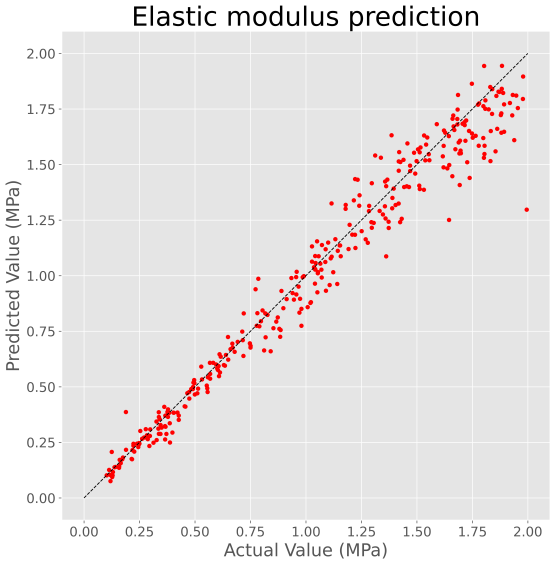

Figure 22: Actual vs predicted elastic modulus.

Tables 15 and 16 show actual values against values predicted by the model for tumor size and elastic modulus, respectively. The red line shows the perfect model. Table 8 shows MAPE for size and elastic modulus for test and train data. Both training and testing loss follow an appropriate pattern over the epochs. Testing loss starts lower than training loss, as expected, but soon training and testing losses even out. The final loss is 0.003, which is low. Size and elastic modulus actual versus predicted values follow the diagonal line of a perfect model very closely. Testing MAPEs for size and stiffness were 1.11 and 7.83, respectively. MAPE less than 5% indicates that the prediction is very accurate [35].

	Train	Test
Size MAPE	0.733	1.11
Elastic modulus MAPE	5.66	7.83

Table 8: MAPE for size and elastic modulus for test and train data.

Experimental Validation

Figures 22-40 show the experimental signal recorded from the ten PDMS breasts. The images on the left show the signal recorded when the probe is positioned right above the tumor, while the images on the right show the signal received when the probe is shifted around.

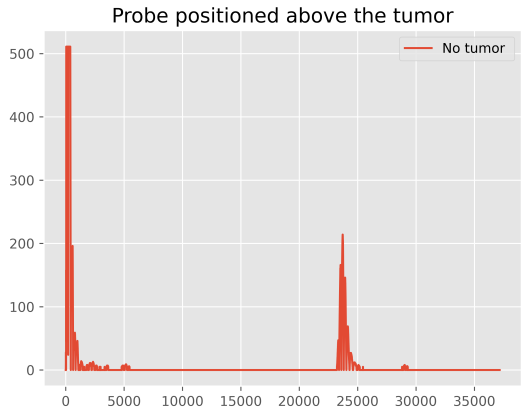

Figure 23: Breast with no tumor experimental signal.

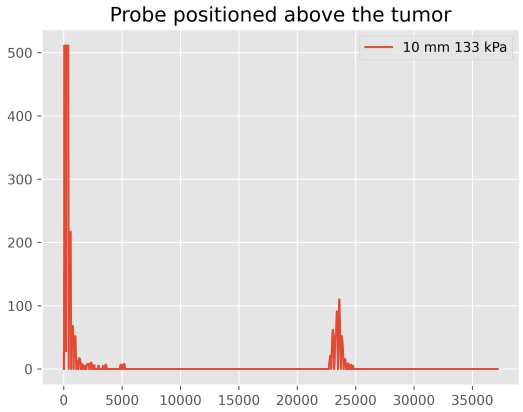

Figure 24: 10 mm 133 kPa tumor experimental signal with probe positioned above the tumor.

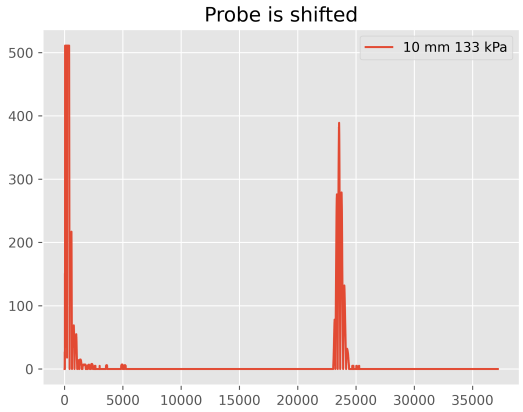

Figure 25: 10 mm 133 kPa tumor experimental signal with probe shifted.

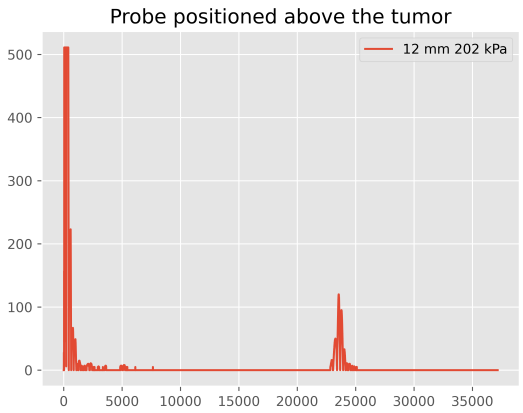

Figure 26: 12 mm 202 kPa tumor experimental signal with probe positioned above the tumor.

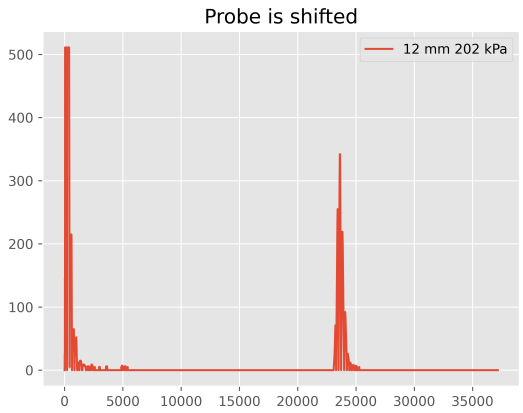

Figure 27: 12 mm 202 kPa tumor experimental signal with probe shifted.

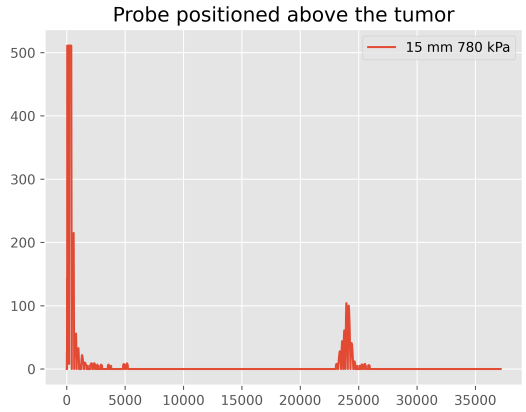

Figure 28: 15 mm 780 kPa tumor experimental signal with probe positioned above the tumor.

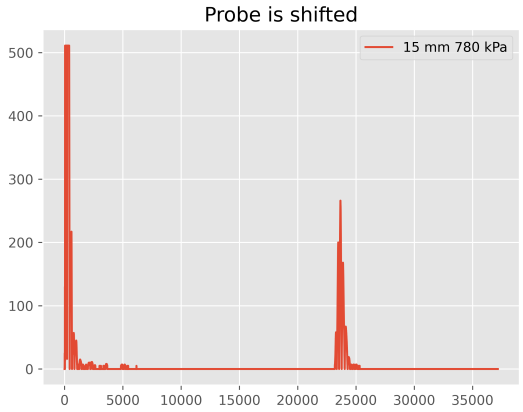

Figure 29: 15 mm 780 kPa tumor experimental signal with probe shifted.

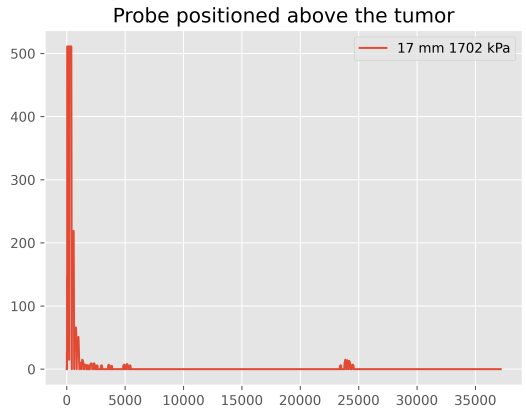

Figure 30: 17 mm 1702 kPa tumor experimental signal with probe positioned above the tumor.

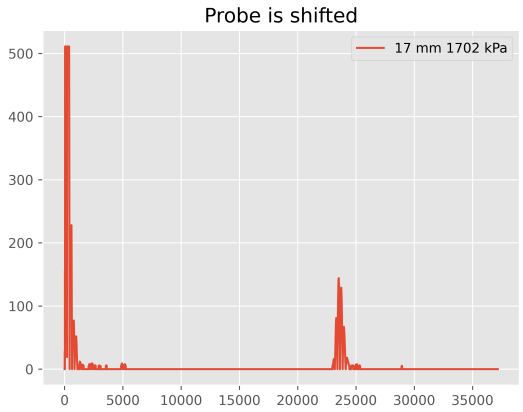

Figure 31: 17 mm 1702 kPa tumor experimental signal with probe shifted.

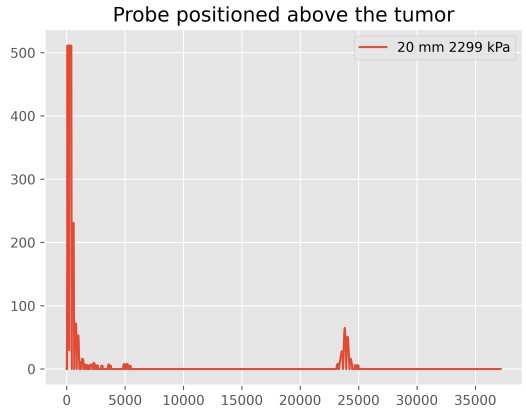

Figure 32: 20 mm 2299 kPa tumor experimental signal with probe positioned above the tumor.

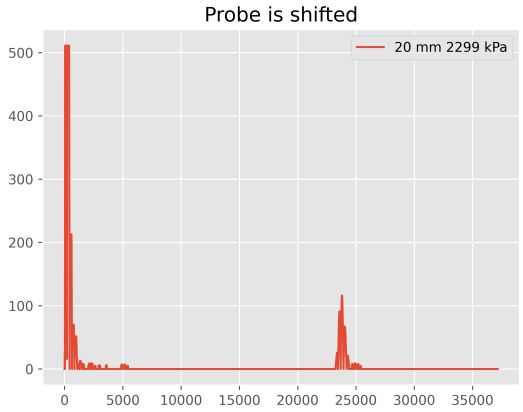

Figure 33: 20 mm 2299 kPa tumor experimental signal with probe shifted.

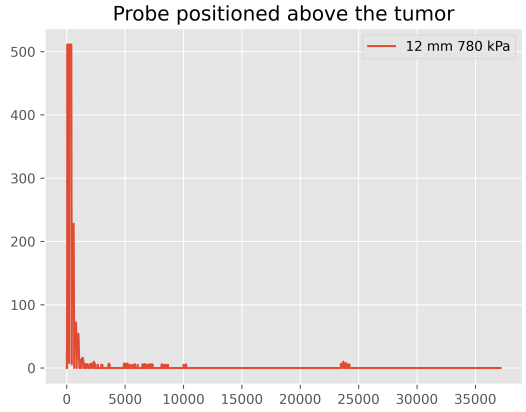

Figure 34: 12 mm 780 kPa tumor experimental signal with probe positioned above the tumor.

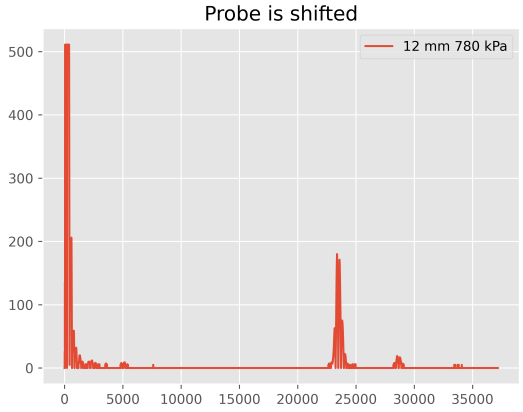

Figure 35: 12 mm 780 kPa tumor experimental signal with probe shifted.

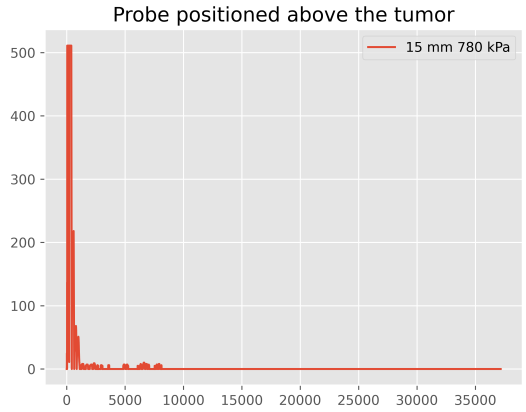

Figure 36: 15 mm 780 kPa tumor experimental signal with probe positioned above the tumor.

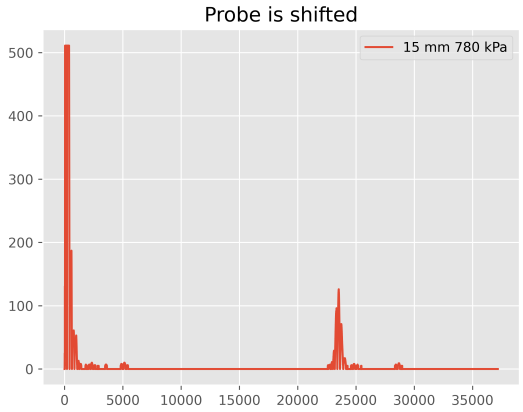

Figure 37: 15 mm 780 kPa tumor experimental signal with probe shifted.

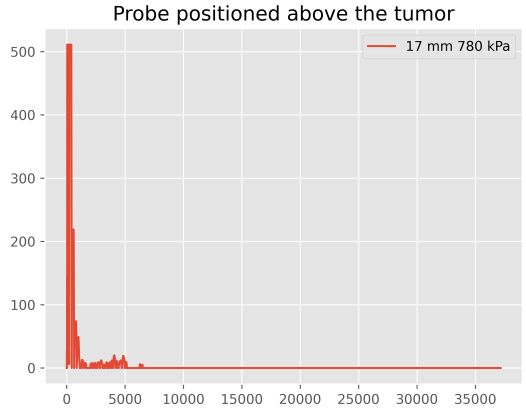

Figure 38: 17 mm 780 kPa tumor experimental signal with probe positioned above the tumor.

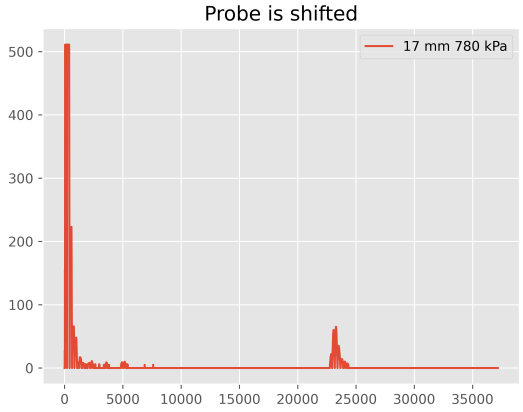

Figure 39: 17 mm 780 kPa tumor experimental signal with probe shifted.

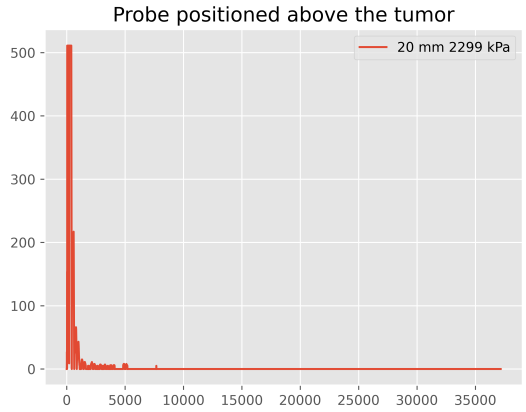

Figure 40: 20 mm 780 kPa tumor experimental signal with probe positioned above the tumor.

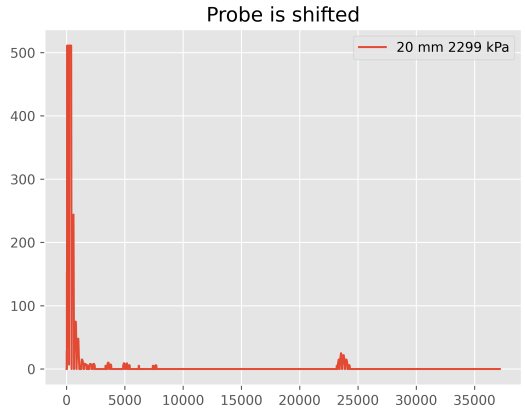

Figure 41: 20 mm 780 kPa tumor experimental signal with probe shifted.

A breast with no tumor showed one reflection peak at around 25,000. This was assumed to be a reflection off the desk surface the PDMS breast was placed on. Breasts with embedded high elastic modulus anomalies showed no additional peak of ultrasound signal being reflected off the tumor. However, we noticed that as the elastic modulus of the tumor and size increase, the amplitude of the desk surface reflection signal was decreasing. We theorized that instead of reflecting the signal, the tumor attenuates it. To test this, we shifted the probe to see if avoiding the tumor would increase the amplitude of the desk reflection peak. This indeed was the case. We hypothesize that while the tumor has elastic modulus different from that of breast, there is not enough of an acoustic impedance difference between the tumor and the breast. Study by Liu et al. shows that increasing the elastic modulus of PDMS does not change the density of this polymer, and only slightly changes the speed of sound inside the material [36]. Thus, different material shall be selected for experimental validation.

Conclusion and Future Work

Less developed countries exhibit higher incidence of breast cancer-related deaths, as patients being diagnosed at later stages. Current state-of-the-art technique, mammography, is not accessible and affordable, as well as exposes patients to harmful ionizing radiation. The alternative diagnostic tool, ultrasound, is more cost effective and safe. However, it has low specificity and requires an experienced radiologist. To circumvent this limitation, we developed a machine learning model that is able to process A-mode ultrasound signal, detect the tumor, and quantify its size and elastic modulus. The performance of the model was evaluated, with MAPE being under 5% and the actual and predicted values following a perfect model closely. We have started the experimental validation work. It requires correct choice of materials.

A different tissue mimicking material (TMM) with an easily adjustable attenuation coefficient has to be selected for breast and tumor phantoms. Polyurehtane gels, oil gels, and custom-made recipes for other TMM have been found in literature [37] [38]. The attenuation coefficients of these materials can be fine-tuned to suit our needs. Additionally, it would make sense to expand the model to detect tumors smaller than 1 mm, to cover the entirety of T1 stage of tumor development. Being able to experimentally validate the model that is able to detect and characterize tumors smaller than 1 mm would be a significant achievement for this study. Additionally, doing an in-depth literature review on the correlation between the tumor size and elastic modulus would be highly beneficial in building an accurate model. This correlation is linear in the simulation dataset, but some data suggest that this might not be the case [39].

References

[1] N. Harbeck, F. Penault-Llorca, J. Cortes, M. Gnant, N. Houssami, P. Poortmans, K. Ruddy, J. Tsang, and F. Cardoso, "Breast cancer," *Nature Reviews Disease Primers*, vol. 5, no. 1, Sep. 2019. Available doi:10.1038/s41572-019-0111-2

[2] Z. Q. Tao, A. Shi, C. Lu, T. Song, Z. Zhang, and J. Zhao, "Breast cancer: Epidemiology and etiology," *Cell Biochemistry and Biophysics*, vol. 72, no. 2, pp. 333–338, Jun. 2014. Available doi:10.1007/s12013-014-0459-6

[3] C. DeSantis, J. Ma, L. Bryan, and A. Jemal, "Breast cancer statistics, 2013," *CA: A Cancer Journal for Clinicians*, vol. 64, no. 1, pp. 52–62, Oct. 2013. [Online]. Available https://doi.org/10.3322/caac.21203

[4] The American Cancer Society, "Breast cancer statistics: How common is breast cancer?," *American Cancer Society*, 12-Jan-2022. [Online]. Available: https://www.cancer.org/cancer/breast-cancer/about/how-common-is-breast-cancer.html. [Accessed: 03-Apr-2022].

[5] M. B. Amin, F. L. Greene, S. B. Edge, C. C. Compton, J. E. Gershenwald, R. K. Brookland, L. Meyer, D. M. Gress, D. R. Byrd, and D. P. Winchester, "The eighth edition AJCC cancer staging manual: Continuing to build a bridge from a population-based to a more 'personalized' approach to cancer staging," *CA: A Cancer Journal for Clinicians*, vol. 67, no. 2, pp. 93–99, Mar. 2017. Available doi:10.3322/caac.21388

[6] A. Karellas and S. Vedantham, "Breast cancer imaging: A perspective for the next decade," *Medical Physics*, vol. 35, no. 11, pp. 4878–4897, Oct. 14, 2008. Available doi: 10.1118/1.2986144

[7] R. L. Egan, "Mammography," *The American Journal of Nursing*, vol. 66, no. 1, p. 108, Jan. 1966. [Online]. Available https://doi.org/10.2307/3419514

[8] L. Wang, "Early diagnosis of breast cancer," *Sensors*, vol. 17, no. 7, p. 1572, Jul. 2017. Available doi:10.3390/s17071572

[9] J. J. Li and Z. M. Shao, "Mammography screening in less developed countries," *SpringerPlus*, vol. 4, no. 1, Oct. 15, 2015. Available doi: 10.1186/s40064-015-1394-8

[10] D. Balabram, C. M. Turra, and H. Gobbi, "Survival of patients with operable breast cancer (stages I-III) at a Brazilian Public Hospital - a closer look into cause-specific mortality," *BMC* Cancer, vol. 13, no. 1, Sep. 24, 2013. Available doi:10.1186/1471-2407-13-434

[11] S. Niu and V. Srivastava, "Simulation trained CNN for accurate embedded crack length, location, and orientation prediction from ultrasound measurements," *International Journal of Solids and Structures*, vol. 242, no. 111521, May 1, 2022. [Online] Available https://doi.org/10.1016/j.ijsolstr.2022.111521

[12] A. Carovac, F. Smajlovic, and D. Junuzovic, "Application of ultrasound in medicine," *Acta Informatica Medica*, vol. 19, no. 3, pp. 168–171, Sep. 2011. Available doi: 10.5455/aim.2011.19.168-171

[13] C. D. Lehman, C. I. Lee, V. A. Loving, M. S. Portillo, S. Peacock, and W. B. DeMartini, "Accuracy and value of breast ultrasound for primary imaging evaluation of symptomatic women 30-39 years of age," *American Journal of Roentgenology*, vol. 199, no. 5, pp. 1169–1177, Nov. 2012. Available doi:10.2214/AJR.12.8842

[14] European Society of Radiology (ESR), "ESR statement on portable ultrasound devices," Insights into Imaging, vol. 10, no. 1, Sep. 16, 2019. Available doi:10.1186/s13244-019-0775-x

[15] R. Sood, A. F. Rositch, D. Shakoor, E. Ambinder, K.-L. Pool, E. Pollack, D. J. Mollura, L. A. Mullen, and S. C. Harvey, "Ultrasound for breast cancer detection globally: A systematic review and meta-analysis," Journal of Global Oncology, vol. 5, pp. 1–17, Aug. 27, 2019. Available doi: 10.1200/JGO.19.00127

[16] C. Voutouri and T. Stylianopoulos, "Accumulation of mechanical forces in tumors is related to hyaluronan content and tissue stiffness," *PLOS ONE*, vol. 13, no. 3, Mar. 21, 2018. [Online]. Available https://doi.org/10.1371/journal.pone.0193801

[17] J. M. Mansour, M. Motavalli, J. Bensusan, M. Li, S. Margevicius, and J. F. Welter, "The nonlinear relationship between speed of sound and compression in articular cartilage: Measurements and modeling," *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 110, no. 103923, Jun. 19, 2020. Available doi: 10.1016/j.jmbbm.2020.103923

[18] J. Fenner, A. C. Stacer, F. Winterroth, T. D. Johnson, K. E. Luker, and G. D. Luker, "Macroscopic stiffness of breast tumors predicts metastasis," *Scientific Reports*, vol. 4, no. 1, Jul. 1, 2014. [Online]. Available https://doi.org/10.1038/srep05512

[19] A. S. Ahuja, “The impact of Artificial Intelligence in medicine on the future role of the physician,” *PeerJ*, vol. 7, Oct. 4, 2019. Available doi: 10.7717/peerj.7702

[20] “Artificial Intelligence in healthcare market report, 2022-2030,” *Grand View Research*, Jan. 2022. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-ai-healthcare-market#. [Accessed: 05-Apr-2022].

[21] A. Hosny, C. Parmar, J. Quackenbush, L. H. Schwartz, and H. J. Aerts, “Artificial Intelligence in radiology,” *Nature Reviews Cancer*, vol. 18, no. 8, pp. 500–510, Aug. 2018. Available doi:10.1038/s41568-018-0016-5

[22] S. K. Zhou et al., "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, May 2021. Available doi: 10.1109/JPROC.2021.3054390

[23] R. Yamashita, M. Nishio, R. K. Do, and K. Togashi, “Convolutional Neural Networks: An overview and application in Radiology,” *Insights into Imaging*, vol. 9, no. 4, pp. 611–629, Jun. 22, 2018. [Online]. Available https://doi.org/10.1007/s13244-018-0639-9

[24] A. Samani, J, Zubovits, and D. Plewes. "Elastic moduli of normal and pathological human breast tissues: an inversion-technique-based investigation of 169 samples" *Physics in Medicine and Biology*, vol. 52, pp. 1565–1576, Feb. 16, 2007. Available doi:https://doi.org/10.1088/0031-9155/52/6/002

[25] Hanxin Chen, Kui Sun, Chanli Ke and Yunfei Shang, "Simulation of ultrasonic testing technique by Finite Element Method," *Proceedings of the IEEE 2012 Prognostics and*

System Health Management Conference (PHM-2012 Beijing), pp. 1-5, 2012. Available doi: 10.1109/PHM.2012.6228959.

[26] B. Ding, H. Qian and J. Zhou, "Activation functions and their characteristics in deep neural networks," *2018 Chinese Control And Decision Conference (CCDC)*, pp. 1836-1841. Jun. 2018. Available doi: 10.1109/CCDC.2018.8407425.

[27] S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *International Journal of Engineering Applied Sciences and Technology*, vol. 04, no. 12, pp. 310–316, Apr. 2020. Available doi: 10.33564/IJEAST.2020.v04i12.054

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv.org*, Jan. 30, 2017. [Online]. Available https://doi.org/10.48550/arXiv.1412.6980.

[29] P. N. Wells and H.-D. Liang, "Medical Ultrasound: Imaging of soft tissue strain and elasticity," *Journal of The Royal Society Interface*, vol. 8, no. 64, pp. 1521–1549, Jun. 2011. Available doi:10.1098/rsif.2011.0054

[30] K. E. Wilson, T. Y. Wang, and J. K. Willmann, "Acoustic and photoacoustic molecular imaging of cancer," *Journal of Nuclear Medicine*, vol. 54, no. 11, pp. 1851–1854, Nov. 2013. Available doi:10.2967/jnumed.112.115568

[31] S. Noimark, R. J. Colchester, R. K. Poduval, E. Maneas, E. J. Alles, T. Zhao, E. Z. Zhang, M. Ashworth, E. Tsolaki, A. H. Chester, N. Latif, S. Bertazzo, A. L. David, S. Ourselin, P. C. Beard, I. P. Parkin, I. Papakonstantinou, and A. E. Desjardins, "Polydimethylsiloxane composites for optical ultrasound generation and multimodality

imaging," *Advanced Functional Materials*, vol. 28, no. 9, p. 1704919, Jan. 15, 2018. [Online] Available https://doi.org/10.1002/adfm.201704919

[32] R. Seghir and S. Arscott, "Extended PDMS stiffness range for flexible systems," *Sensors and Actuators*, vol. 230, pp. 33–39, Jul. 1, 2015. [Online]. Available https://doi.org/10.1016/j.sna.2015.04.011

[33] G. Xu, Z. Ni, X. Chen, J. Tu, X. Guo, H. Bruus, and D. Zhang, "Acoustic characterization of Polydimethylsiloxane for Microscale Acoustofluidics," *Physical Review Applied*, vol. 13, no. 5, May, 2020. Available doi: 10.1103/PhysRevApplied.13.054069

[34] R. N. Palchesko, L. Zhang, Y. Sun, and A. W. Feinberg, "Development of polydimethylsiloxane substrates with tunable elastic modulus to study cell mechanobiology in muscle and nerve," *PLOS ONE*, vol. 7, no. 12, Dec. 11, 2012. Available https://doi.org/10.1371/journal.pone.0051499

[35] D. A. Swanson, "On the Relationship among Values of the Same Summary Measure of Error when it is used across Multiple Characteristics at the Same Point in Time: An Examination of MALPE and MAPE," *Review of Economics & Finance*, Better Advances Press, Canada, vol. 5, pp. 1-14, Aug. 2015. [Online]. Available https://escholarship.org/content/qt1f71t3x9/qt1f71t3x9.pdf?t=o5wul1

[36] N. N. Liu, Y. D. Cui, B. C. Khoo, and A. M. Zhang, "Damage characteristics of elastic material through a thin membrane using high-intensity focused ultrasound (HIFU)," *AIP Advances*, vol. 8, no. 115123, Nov. 20, 2018. [Online]. Available https://doi.org/10.1063/1.5050432

[37] T. Kondo, M. Kitatuji and H. Kanda, "New tissue mimicking materials for ultrasound phantoms," *IEEE Ultrasonics Symposium 2005*,pp. 1664-1667, Sept. 2005. Available doi: 10.1109/ULTSYM.2005.1603183

[38] L. M. Cannon, A. J. Fagan, and J. E. Browne, "Novel tissue mimicking materials for high frequency breast ultrasound phantoms," *Ultrasound in Medicine & Biology*, vol. 37, no. 1, pp. 122–135, Jan. 2011. Available doi:10.1016/j.ultrasmedbio.2010.10.005

[39] E. J. Song, Y.-M. Sohn, and M. Seo, "Tumor stiffness measured by quantitative and qualitative shear wave elastography of breast cancer," *The British Journal of Radiology*, vol. 91, no. 1086, Apr. 2018. Available doi:10.1259/bjr.20170830

Appendix A: Abaqus Simulation Code

```
from abaqus import *

from abaqusConstants import *

import sketch

import part

from mesh import *

import odbAccess

from odbAccess import *


import sys

import os

import random

import math

from math import sqrt

import xlsxwriter

import numpy as np


print(sys.version)


features = xlsxwriter.Workbook('C:/Users/arusnak/ABAQUS_breast/features.
   xlsx')
labels = xlsxwriter.Workbook('C:/Users/arusnak/ABAQUS_breast/labels.xlsx')


f_worksheet = features.add_worksheet()

l_worksheet = labels.add_worksheet()


#print(sys.path)
```

```
E = [0]*1000
tumorSize = [0]*1000
for i in range(0,1000):
    E[i] = random.uniform(0.1, 2)
    tumorSize[i] = random.uniform(10,20)



E=sorted(E)
tumorSize=sorted(tumorSize)

#E=[0.4, 0.8, 1, 1.5, 2, 2.5, 3, 3.5, 3.7, 4] #input in megapascals
#tumorSize=[14, 15, 16, 17, 17.5, 18, 19, 20, 21, 22] #imput in mm

#E=[2] #input in megapascals
#tumorSize=[20] #imput in mm

dens = 0.97 #define density for both, in kg/m3
v = 0.499 #define Poissons ratio for both
f=1e6 # frequency of the US

BreastStiff = 0.1 # in megapascals, stiffness of breast does not vary
stiffCalc = BreastStiff*(10**6) #convert breast stifness to pascal

c=sqrt((stiffCalc*(1-v))/(dens*(1+v)*(1-2*v))) # in m/s
lam = c/f #in m
meshSize = lam/10 # in m
```

```
timeIncr=(meshSize/c)/20 # in s
totalTime=((2*0.04)/c) # in s


meshSize=(lam/10)*1000


print("E", stiffCalc)
print("c ", c)
print("lambda", lam)
print("meshSize", meshSize, type(meshSize))
print("timeIncr", timeIncr)



#define values
breastDensity = (dens/(1e12),) #convert density to tonne/mm^3
tumorDensity = (dens/(1e12),)


col = -1 #start counting columns for xlsxwriter


for i in range(0,len(E)):


        try:
            os.remove('Model_'+str(i)+'.lck')
        except WindowsError:
            pass



        col += 1
```

```
 stiff=E[i]
 diam=tumorSize[i]

 breastE = (BreastStiff, v)
 tumorE = (stiff, v)

#meshSize=0.5

 myModel = mdb.Model(name ='Model_'+str(i)) #create a  model

 #sketching a breast
 breastSketch = myModel.ConstrainedSketch(name='breast', sheetSize
    =100.0) #create sketch
 breastSketch.ConstructionLine(point1=(0, -100), point2=(0, 100)) #
    create a construction line
 breastSketch.ArcByCenterEnds(center=(0,0),point1=(40,0),point2
    =(0,40)) #draw breast
 breastSketch.Line(point1=(0,0),point2=(0,40)) #draw breast
 breastSketch.Line(point1=(0,0),point2=(40,0))#draw breast
 breast = myModel.Part(name='breast', dimensionality=AXISYMMETRIC,
    type=DEFORMABLE_BODY)
 breast.BaseShell(sketch=breastSketch)

 #sketching a tumor
 tumorSketch = myModel.ConstrainedSketch(name='tumor', sheetSize
    =100.0) #create sketch
```

```
tumorSketch.ConstructionLine(point1=(0, -100), point2=(0, 100)) #
    create a construction line
tumorSketch.ArcByCenterEnds(center=(0,20),point1=(0,20-(diam/2)),
    point2=(0,20+(diam/2))) #draw tumor
tumorSketch.Line(point1=(0,20-(diam/2)),point2=(0,20+(diam/2))) #
    draw tumor

#partition breast to create tumor
breast.PartitionFaceBySketch(faces = breast.faces[0:1], sketch=
    tumorSketch)

#create materials
import material
#create breast material
BreastMaterial = myModel.Material(name='breast tissue') #create
    material
BreastMaterial.Elastic(type = ISOTROPIC, table = (breastE,)) #
    elasticity
BreastMaterial.Density(table=(breastDensity,)) # density
#create tumor material
TumorMaterial = myModel.Material(name='tumor')#create material
TumorMaterial.Elastic(type = ISOTROPIC, table = (tumorE,)) #
    elasticity
TumorMaterial.Density(table=(tumorDensity,)) # density

#create instances from parts
myAssembly = myModel.rootAssembly #create assembly
```

```
BreastInstance = myAssembly.Instance(name='breastinstance', part =
    breast, dependent = OFF) #create breast instance

#create and assign sections
import section

BreastSection = myModel.HomogeneousSolidSection(name='
   BreastSection', material = 'breast tissue')
TumorSection = myModel.HomogeneousSolidSection(name='TumorSection
   ', material = 'tumor')
import regionToolset
breastRegion = regionToolset.Region(faces = breast.faces.findAt
   (((15,20,0),)))
breast.SectionAssignment(region = breastRegion, sectionName = '
   BreastSection')
tumorRegion = regionToolset.Region(faces = breast.faces.findAt(((
   (diam/2)/2 ,20,0),)))
breast.SectionAssignment(region = tumorRegion, sectionName = '
   TumorSection')


#create a probe
breastEdge = BreastInstance.edges.findAt(((28.284271247462,
   28.284271247462, 0.0),)) #create an Edge object for breast edge
partition = myAssembly.PartitionEdgeByPoint(edge = breastEdge[0],
   point = (6.257379,39.507534,0)) #create a partition
probeEdge = BreastInstance.edges.findAt(((3.138364,39.876693,0),))
```

```
    #create an Edge object for probe edge
probeSurface = myAssembly.Surface(side2Edges=probeEdge, name='
   probe')
probeRegion = regionToolset.Region(edges = probeEdge)
probeSet = myAssembly.Set(name = 'probeSet', region = probeRegion)



#mesh the assembly
import mesh
myAssembly.seedPartInstance(regions = (BreastInstance, ), size =
   meshSize) #seed instance
myAssembly.setMeshControls(regions=(BreastInstance.faces[0],),
   elemShape = QUAD, technique = FREE) #set mesh controls for
   cutPart
myAssembly.setElementType(elemTypes=(ElemType( elemCode=CAX4R,
   elemLibrary=EXPLICIT), ElemType(elemCode=CAX4R, elemLibrary=
   EXPLICIT)), regions=(BreastInstance.faces[0],))
myAssembly.generateMesh(regions=(BreastInstance,))

#create step
import step
myModel.ExplicitDynamicsStep(name="waveStep", previous = "Initial
   ", timePeriod = totalTime, timeIncrementationMethod =
   FIXED_USER_DEFINED_INC, userDefinedInc = timeIncr)

#create and apply load
import load
```

```
myModel.TabularAmplitude(data=((0.0, 0.0),
(3e-08, 0.00193831950713421),
(6e-08, 0.00733156530200401),
(9e-08, 0.0149553688471836),
(1.2e-07, 0.0229018899099815),
(1.5e-07, 0.0287682579175257),
(1.8e-07, 0.0298997180897102),
(2.1e-07, 0.0236685495833722),
(2.4e-07, 0.0077667678223513 8),
(2.7e-07, -0.0195108844724957),
(3e-07, -0.0590169943749474),
(3.3e-07, -0.110556088450452),
(3.6e-07, -0.172761898856107),
(3.9e-07, -0.243060632353894),
(4.2e-07, -0.317727779806876),
(4.5e-07, -0.392039521920206),
(4.8e-07, -0.460513060657597),
(5.1e-07, -0.517223685156237),
(5.4e-07, -0.556180508961424),
(5.7e-07, -0.57173799851997),
(6e-07, -0.559016994374948),
(6.3e-07, -0.514307163396255),
(6.6e-07, -0.435422865745577),
(6.9e-07, -0.321986312142295),
(7.2e-07, -0.175615544492801),
(7.5e-07, -1.83697019872103e-16),
(7.8e-07, 0.199147084678648),
```

```
(8.1e-07, 0.414262793227062),
(8.4e-07, 0.636230724212417),
(8.7e-07, 0.854787048461122),
(9e-07, 1.05901699437495),
(9.3e-07, 1.23791610641207),
(9.6e-07, 1.38098581329584),
(9.9e-07, 1.47882977170031),
(1.02e-06, 1.52371634197136),
(1.05e-06, 1.5100735106701),
(1.08e-06, 1.43488558028085),
(1.11e-06, 1.29796585319769),
(1.14e-06, 1.10208608064127),
(1.17e-06, 0.852951259752979),
(1.2e-06, 0.559016994374949),
(1.23e-06, 0.231155582656112),
(1.26e-06, -0.117814271236274),
(1.29e-06, -0.473711224746334),
(1.32e-06, -0.821658865040433),
(1.35e-06, -1.14680224666742),
(1.38e-06, -1.43503536493284),
(1.41e-06, -1.67370048215685),
(1.44e-06, -1.8522214064745),
(1.47e-06, -1.96263618195024),
(1.5e-06, -2.0),
(1.53e-06, -1.96263618195024),
(1.56e-06, -1.8522214064745),
(1.59e-06, -1.67370048215685),
```

```
(1.62e-06, -1.43503536493284),
(1.65e-06, -1.14680224666742),
(1.68e-06, -0.821658865040435),
(1.71e-06, -0.473711224746336),
(1.74e-06, -0.117814271236276),
(1.77e-06, 0.231155582656111),
(1.8e-06, 0.559016994374947),
(1.83e-06, 0.852951259752979),
(1.86e-06, 1.10208608064127),
(1.89e-06, 1.29796585319769),
(1.92e-06, 1.43488558028085),
(1.95e-06, 1.5100735106701),
(1.98e-06, 1.52371634197136),
(2.01e-06, 1.47882977170031),
(2.04e-06, 1.38098581329584),
(2.07e-06, 1.23791610641207),
(2.1e-06, 1.05901699437495),
(2.13e-06, 0.854787048461124),
(2.16e-06, 0.636230724212417),
(2.19e-06, 0.414262793227062),
(2.22e-06, 0.199147084678649),
(2.25e-06, 5.51091059616309e-16),
(2.28e-06, -0.175615544492802),
(2.31e-06, -0.321986312142294),
(2.34e-06, -0.435422865745577),
(2.37e-06, -0.514307163396255),
(2.4e-06, -0.559016994374947),
```

```
(2.43e-06, -0.57173799851997),
(2.46e-06, -0.556180508961424),
(2.49e-06, -0.517223685156237),
(2.52e-06, -0.460513060657597),
(2.55e-06, -0.392039521920206),
(2.58e-06, -0.317727779806877),
(2.61e-06, -0.243060632353894),
(2.64e-06, -0.172761898856108),
(2.67e-06, -0.110556088450453),
(2.7e-06, -0.0590169943749476),
(2.73e-06, -0.019510884472496),
(2.76e-06, 0.00776676782235107),
(2.79e-06, 0.0236685495833723),
(2.82e-06, 0.0298997180897102),
(2.85e-06, 0.0287682579175258),
(2.88e-06, 0.0229018899099815),
(2.91e-06, 0.0149553688471836),
(2.94e-06, 0.00733156530200421),
(2.97e-06, 0.00193831950713421),
(3e-06, 0.0)), name = 'wave')

#create interaction
from interaction import *
interaction1 = myModel.ContactProperty(name = "intProperty-1")
interaction1.NormalBehavior()
interaction1.TangentialBehavior()
myModel.ContactExp(name = "intProperty-1", createStepName="
```

```
    waveStep", useAllstar=TRUE, contactPropertyAssignments=((GLOBAL
    , SELF, "intProperty-1"),))

#create load
load = myModel.Pressure(name="Wave", createStepName = "waveStep",
    region = probeSurface, magnitude = 0.01, amplitude = 'wave')

#apply boundary conditions
ribEdge = BreastInstance.edges.findAt(((1,0,0),)) #define the rib
    edge
ribRegion = regionToolset.Region(edges = ribEdge) #create a rib
    region from rib edge
ribBC = myModel.DisplacementBC(name = "ribBC", createStepName = "
    waveStep", region = ribRegion, u1=0,u2=0,ur3=0, amplitude="wave
    ") #define rib BC
symmetryEdge1 = BreastInstance.edges.findAt(((0,1,0),)) #define
    symmetry edge in three parts
symmetryEdge2 = BreastInstance.edges.findAt(((0,39,0),)) #define
    symmetry edge in three parts
symmetryEdge3 = BreastInstance.edges.findAt(((0,20,0),)) #define
    symmetry edge in three parts
symmetryRegion1 = regionToolset.Region(edges = symmetryEdge1) #
    make symmetry region from symm edge
symmetryRegion2 = regionToolset.Region(edges = symmetryEdge2) #
    make symmetry region from symm edge
symmetryRegion3 = regionToolset.Region(edges = symmetryEdge3) #
    make symmetry region from symm edge
```

```
symmetryBC1 = myModel.DisplacementBC(name="ysymm1", createStepName
   ="waveStep", region = symmetryRegion1, u1=0, ur3=0, amplitude="
   wave")
symmetryBC2 = myModel.DisplacementBC(name="ysymm2", createStepName
   ="waveStep", region = symmetryRegion2, u1=0, ur3=0, amplitude="
   wave")
symmetryBC3 = myModel.DisplacementBC(name="ysymm3", createStepName
   ="waveStep", region = symmetryRegion3, u1=0, ur3=0,amplitude="
   wave")

#add dispacement to historyOutputRequests
mdb.models['Model_'+str(i)].historyOutputRequests['H-Output-1'].
   setValues(frequency=1, rebar=EXCLUDE, region=mdb.models['Model_
   '+str(i)].rootAssembly.sets['probeSet'],sectionPoints=DEFAULT,
   variables=('U2',))

#run a job
import job
job = mdb.Job(name='Model_'+str(i), model = myModel,
   explicitPrecision = DOUBLE, nodalOutputPrecision=FULL)
job.submit()
job.waitForCompletion()

#print(session.odbs)

odb = openOdb('Model_'+str(i)+'.odb')
Odb_obj = session.odbs['Model_'+str(i)+'.odb']
```

```
disp_list = [0]*int(((totalTime/timeIncr)+2))


first_key = list(Odb_obj.steps['waveStep'].historyRegions.keys())
    [0]
times=Odb_obj.steps['waveStep'].historyRegions[first_key].
    historyOutputs['U2'].data
time_list = []


for m in range(0, len(times)):
    time_list.append(times[m][0])


print(len(disp_list))


for histObj in Odb_obj.steps['waveStep'].historyRegions.keys():
    U2_data=Odb_obj.steps['waveStep'].historyRegions[histObj].
        historyOutputs['U2'].data


    for num in range(0, len(U2_data)):
        disp_list[num] = disp_list[num] + U2_data[num][1]


for k in range(0, len(disp_list)):
    disp_list[k]=disp_list[k]/len(Odb_obj.steps['waveStep'].
        historyRegions)


#print(time_list[-10:])
#print(disp_list[-10:])
#region = step1.historyRegions['Node BREASTINSTANCE'+str(
```

```
    set_sensor[0].label)]
#dictionary = dict(zip(time_list, disp_list))

f_worksheet.write_column(0, 0, time_list)
f_worksheet.write_column(0, col+1, disp_list)
l_worksheet.write(0, col, stiff)
l_worksheet.write(1, col, diam)

try:
    os.remove('Model_'+str(i)+'.abq')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.inp')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.ipm')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.pac')
except WindowsError:
    pass
```

```
try:
    os.remove('Model_'+str(i)+'.prt')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.res')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.sel')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.stt')
except WindowsError:
    pass

try:
    os.remove('Model_'+str(i)+'.sta')
except WindowsError:
    pass

try:
```

```
    os.remove('Model_'+str(i)+'.mdl')
except WindowsError:
    pass


try:
    os.remove('Model_'+str(i)+'.log')
except WindowsError:
    pass


try:
    os.remove('Model_'+str(i)+'.msg')
except WindowsError:
    pass


try:
    os.remove('Model_'+str(i)+'.dat')
except WindowsError:
    pass


try:
    os.remove('Model_'+str(i)+'.com')
except WindowsError:
    pass



odb.close()
```

```
        try:
            os.remove('Model_'+str(i)+'.odb')
        except WindowsError:
            pass


features.close()
labels.close()
```

Appendix B: CNN Code

```
#Model


import torch.nn as nn


class MyModel(nn.Module):
    def __init__(self, fc_out):
        super(MyModel, self).__init__()


        # Convolution block
        self.conv_layer = nn.Sequential(
            nn.Conv1d(1, 4, 8, stride=4, padding=2),
            nn.ReLU(),
            nn.Conv1d(4, 8, 8, stride=4, padding=2),
            nn.ReLU(),
            nn.MaxPool1d(2, stride=2, padding=0)
        )


        # Fully connected block
        self.fc_layer = nn.Sequential(
            nn.Linear(8 * 103, 400),
            nn.Dropout(0.2),
            nn.ReLU(),
            nn.Linear(400, fc_out)
        )
```

```
    def forward(self, x):
        #print(x.size())
        x = self.conv_layer(x)
        #print(x.size())
        x = x.view(-1, 8 * 103)
        #print(x.size())
        x = self.fc_layer(x)
        #print(x.size())
        #print("_____________")
        return x

#Train and test the model

import torch
device = torch.device("cuda:0")

import torch
import torch.nn as nn
import numpy as np
import random
import pandas as pd
import matplotlib.pyplot as plt
from mymodel import MyModel

# Seeds the random generator
torch.manual_seed(123)
np.random.seed(123)
```

```
# Hyper-parameters
num_epochs = 1000
learning_rate = 0.001

# Variable to store epoch and loss for plotting
epoch_list = []
loss_train = []
loss_val = []

#load data
features = pd.read_excel('features329.xlsx', header=None, index_col=None)
labels = pd.read_excel('labels329.xlsx', header=None, index_col=None)

#drop the initial pulse from the dataframe
features = features.drop(features.index[0:560])
features = features.drop(features.columns[0], axis=1)

#parse in data
set = random.sample(range(0, len(features.columns)), len(features.columns)
   )
train = set[0 : int(round(len(features.columns)*2/3, 0))]
test  =set[int(round(len(features.columns)*2/3, 0)):]
train_f = features.iloc[:,train]
train_l = labels.iloc[:,train]
test_f = features.iloc[:,test]
test_l = labels.iloc[:,test]
```

```
#save min, max, and range for denormalization later
train_min = train_l.min(axis=1)
train_max = train_l.max(axis=1)
train_range = train_l.max(axis=1)-train_l.min(axis=1)
test_min = test_l.min(axis=1)
test_max = test_l.max(axis=1)
test_range = test_l.max(axis=1)-test_l.min(axis=1)

#normalize labels [0,1]
train_l=train_l.subtract(train_l.min(axis=1), axis=0).div(train_l.max(axis
   =1)-train_l.min(axis=1), axis=0)
test_l=test_l.subtract(test_l.min(axis=1), axis=0).div(test_l.max(axis=1)-
   test_l.min(axis=1), axis=0)

#normalize features
train_f=train_f.div(train_f.abs().max())
test_f=test_f.div(test_f.abs().max())

# Transform the data into tensors
inputs_train = torch.unsqueeze(torch.from_numpy(train_f.values).float(),
   1)
targets_train = torch.from_numpy(train_l.values).float()
inputs_test = torch.unsqueeze(torch.from_numpy(test_f.values).float(), 1)
targets_test = torch.from_numpy(test_l.values).float()

#set the number of output neurons
```

```
fc_out = 2


# CNN model
model = MyModel(fc_out)


# Loss and optimizer
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)


inputs_train = torch.transpose(inputs_train, 0,2)
inputs_test=torch.transpose(inputs_test, 0,2)
targets_train = torch.transpose(targets_train, 0,1)
targets_test = torch.transpose(targets_test, 0,1)


"""
print(inputs_train.size())
print(inputs_test.size())
print(targets_train.size())
print(targets_test.size())
"""


# Train the model
for epoch in range(num_epochs):


    # Forward pass
    outputs_train = model(inputs_train)
    outputs_test = model(inputs_test)
```

```
    loss = criterion(outputs_train, targets_train)
    loss_v = criterion(outputs_test, targets_test)

    # Backward and optimize
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    if (epoch + 1) % 5 == 0:
        epoch_list.append(epoch+1)
        loss_train.append(loss.item())
        loss_val.append(loss_v.item())

    if (epoch + 1) % 100 == 0:
        print('Epoch [{}/{}], Loss: {:.10f}'.format(epoch + 1, num_epochs,
            loss.item()))

torch.save(model, 'trained_model.pt')

model.eval()
with torch.no_grad():
    outputs_test = model(inputs_test)
    outputs_train = model(inputs_train)

outputs_test = outputs_test.detach().numpy()
targets_test = targets_test.numpy()
outputs_train = outputs_train.detach().numpy()
```

```
targets_train = targets_train.numpy()

# De-normalize the data for visualization (targets test and outputs test
    datasets)
outputs_test[:,0]  = (outputs_test[:,0]*test_range[0])+test_min[0]
outputs_train[:,0]  = (outputs_train[:,0]*train_range[0])+train_min[0]
outputs_test[:,1]  = (outputs_test[:,1]*test_range[1])+test_min[1]
outputs_train[:,1]  = (outputs_train[:,1]*train_range[1])+train_min[1]

targets_test[:,0]  = (targets_test[:,0]*test_range[0])+test_min[0]
targets_train[:,0]  = (targets_train[:,0]*train_range[0])+train_min[0]
targets_test[:,1]  = (targets_test[:,1]*test_range[1])+test_min[1]
targets_train[:,1]  = (targets_train[:,1]*train_range[1])+train_min[1]

print(targets_test)

# Calculate MAPE for test data
abs_error = targets_test - outputs_test
rel_error = abs_error / targets_test * 100
abs_rel_error = abs(rel_error)
MAPE = np.sum(abs_rel_error, 0) / abs_rel_error.shape[0]
print('MAPE for test data is', MAPE)

# Calculate MAPE for training data
abs_error = targets_train - outputs_train
rel_error = abs_error / targets_train * 100
abs_rel_error = abs(rel_error)
```

```
MAPE = np.sum(abs_rel_error, 0) / abs_rel_error.shape[0]
print('MAPE for training data is', MAPE)


# Plot training epochs
plt.style.use('ggplot')
fig, ax = plt.subplots()
ax.plot(epoch_list, loss_train, 'o-')
ax.plot(epoch_list, loss_val, 'o-')
ax.set_xlabel('Training Epochs')
ax.set_ylabel('Loss')
ax.legend(['Training', 'Testing'])
plt.savefig('Loss.png', format='png', dpi=600)
plt.show()
plot_index = 0


# Plot P-A for stiffness
plt.style.use('ggplot')
PAline = np.linspace(min(outputs_test[0]), max(outputs_test[0]), num=20)
fig, ax = plt.subplots()
ax.scatter(targets_test[:, plot_index], outputs_test[:, plot_index],s=20,
   marker='o', color='red')
ax.plot([0, 1], [0, 1], transform=ax.transAxes)
ax.set_xlabel('Actual Value (MPa)', fontsize=20)
ax.set_ylabel('Predicted Value (MPa)', fontsize=20)
ax.tick_params(labelsize=15)
ax.set_aspect('equal')
fig.set_size_inches(10, 10)
```

```
plt.title('Elastic modulus prediction', fontsize=30)
plt.savefig('Stiffness_pred.png', format='png', dpi=600)
plt.show()
plot_index += 1


# Plot P-A for size
plt.style.use('ggplot')
PAline = np.linspace(min(outputs_test[1]), max(outputs_test[1]), num=20)
fig, ax = plt.subplots()
ax.scatter(targets_test[:, plot_index], outputs_test[:, plot_index],s=20,
   marker='o', color='red')
ax.plot([0, 1], [0, 1], transform=ax.transAxes)
ax.set_xlabel('Actual Value (mm)', fontsize=20)
ax.set_ylabel('Predicted Value (mm)', fontsize=20)
ax.tick_params(labelsize=15)
ax.set_aspect('equal')
fig.set_size_inches(10, 10)
plt.title('Size prediction', fontsize=30)
plt.savefig('Size_pred.png', format='png', dpi=600)
plt.show()
```