

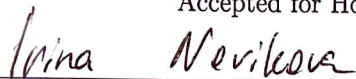
Optical Fiber-Linked Magnetometer Employing Artificial Intelligence for Magnetic Field Measurement

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelor of Science with Honors in
Physics from the College of William and Mary in Virginia,


by

Sofia Brown

Accepted for Honors



Advisor: Prof. Irina Novikova



Prof. David Armstrong



Prof. Stephen Trefethen

Williamsburg, Virginia

May 1 2022

Contents

Acknowledgments	iv
List of Figures	vii
List of Tables	viii
Abstract	v
1 Introduction	1
1.1 Background	1
1.2 Motivation and Theory	1
2 Experimental Design: Optical Setup, Data Acquisition, and Image Classification Algorithm	7
2.1 Experimental Setup	7
2.1.1 Preliminary Waveplate Setup	7
2.1.2 The Faraday Apparatus and Setup	9
2.2 The Image Classification Algorithm	12
2.2.1 Image Classification Theory	12
2.2.2 The Algorithm	17
3 Preliminary Work to Achieve Image Recognition	21

4	Current Work to Reduce Algorithm Sensitivity to Intensity	24
4.1	Basis Algorithm Tests	24
4.2	Exploring Algorithm Sensitivity to Laser Power	25
4.2.1	Maximum Difference in Power of 4 mW	25
4.2.2	Maximum Difference in Power of 1.5 mW.	27
4.2.3	Training on Mixed Power Datasets	29
4.2.4	Training on Modulated Power	32
4.3	Training the Algorithm to Distinguish between Images Taken at Different Powers	35
4.4	Fixing Image Saturation Issues	38
4.4.1	Distinguishing between 0° and 1° , Train on One Power Level, Test on Another	41
4.4.2	Distinguishing between 0° and 1° , Train on Mixed Power Levels	41
4.4.3	Train the Algorithm to Distinguish between Two Intensities for a Fixed Polarization	43
5	Improving Angular Sensitivity with the Waveplate	46
6	Implementing a Faraday Magnetometer	49
6.1	New Setup Replacing the Waveplate with a Solenoid	49
6.2	Faraday Magnetometer Benchmark Angular Sensitivity Tests	49
6.3	Imposing Masks on the Image Data	50
6.4	Exploring Performance Dependence on Mask Size and Location	52
6.5	Testing Smaller Angles	54
6.6	Improving Angular Sensitivity of the Faraday Rotation Magnetometer	54
7	Conclusion	59

A	Image Processing Code and Image Classification Algorithm	65
A.1	Image Acquisition and Rescaling Code Sample	65
A.2	Image Classification Algorithm	68

Acknowledgments

Thank you to Professor Irina Novikova for guiding me through this thesis project and for her stalwart mentorship and advocacy on my part throughout my undergraduate experience. Thank you to Savannah Cuzzo for always being there to help with coding and camera problems and to Nic DeStefano for his solid company and help in the lab. Thank you to Professors Armstrong and Trefethen for serving on my thesis committees and for their comments and critiques.

List of Figures

1.1	A Visualization of Faraday Rotation	3
1.2	Comparison of Gaussian Laser Profile and Multimode Fiber Output Profile	4
1.3	Comparison of Multimode Fiber Output Patterns for 0° and 45° polarization	5
2.1	The Experimental Setup	8
2.2	Experimental Calibration of the Faraday Apparatus	10
2.3	Experimental Setup Including the Faraday Apparatus	11
2.4	Suboptimal and Optimal Training Plots	19
2.5	The Algorithm	20
3.1	Sample Image Data of Speckle Patterns	23
4.1	Training and Testing Combinations: Maximum Difference in Power 4 mW.	26
4.2	Learning Curve for 0° vs. 1° Polarization at 1 mW Image Intensity .	27
4.3	Learning Curve for 0° vs. 1° Polarization, at 5 mW Image Intensity .	27
4.4	Training and Testing Combinations: Maximum Difference in Power 1.5 mW.	28
4.5	Learning Curves for Training on 0.5, 1, 1.5, and 2 mW	29

4.6	Suboptimal Training Convergence When Distinguishing between 0° and 1°	30
4.7	Learning Curves for Training on Pairs of Power Values	31
4.8	Training on Two Different Powers between 0.5 and 2 mW.	32
4.9	Learning Curves for Training on Trios of Power Values	33
4.10	Training and Testing Combinations: Training on 3 Power Levels.	34
4.11	Training on a Random Assortment of Powers Generated by AOM	35
4.12	Learning Curves for Distinguishing between Images at Two Different Intensities with a Fixed Polarization	38
4.13	0.8 mW	39
4.14	0.2mW	39
4.15	0.8 mW/0.2 mW	39
4.16	Example Analysis of Saturated and Incorrectly Normalized Speckle Patterns	39
4.17	Example Analysis of Unsaturated and Correctly Normalized Speckle Patterns	40
4.18	Training on a Single Power Level: Unsaturated Images.	41
4.19	Learning Curves for Training on 0.2, 0.3, 0.5, and 1 mW, Unsaturated Images	42
4.20	Training and Testing Combinations: Training on 3 Power Levels, Unsaturated	43
4.21	Learning Curves for Training on Trios of Power Values, Unsaturated Images	44
4.22	Learning Curves for Distinguishing between Unsaturated Images at Two Different Intensities with a Fixed Polarization	45

5.1	Learning Curves for Improved Angular Sensitivity Using Unsaturated Images	48
6.1	Learning Curves for Benchmark Faraday Magnetometer Tests	51
6.2	Sample Masked Images	52
6.3	Learning Curves for Determining Angular Sensitivity Using Small Training Sets	57
6.4	Learning Curves for Determining Angular Sensitivity Using Larger Training Sets	58

List of Tables

2.1	Example of a Confusion Matrix	17
4.1	Modulated Power Testing Results.	35
4.2	Image Classification Accuracies for Two Values of Power and Fixed Polarization of 0° , Testing on 0°	37
4.3	Image Classification Accuracies for Two Values of Power and Fixed Polarization of 0° , testing on 1°	37
4.4	Image Classification Accuracies for Two Values of Power and Fixed Polarization of 0° , testing on 0° with Unsaturated Images	45
5.1	Improved Angular Sensitivity with Unsaturated Images	47
6.1	Faraday Rotation Benchmark Tests	50
6.2	Faraday Magnetometer Fractional Angles Tests	54
6.3	Faraday Magnetometer Fractional Angles Tests	56

Abstract

We combine a camera, an optical fiber, and artificial intelligence into a single optical magnetometer for magnetic field measurement. This novel combination provides enhanced spatial resolution, a mobile configuration, and efficient, unbiased data processing capabilities. The magnetometer is based on an undergraduate laboratory Faraday rotation apparatus (a glass rod surrounded by a solenoid), linked to a camera via a multimode optical fiber. To identify varying magnetic field strengths, an image classification algorithm analyzes the fiber output “speckle” patterns that result from different magnetically-induced changes in probe beam polarization. Initially, as we constructed and strengthened the algorithm, we simulated these polarization changes using a waveplate, and we investigated the algorithm’s response to external factors such as natural fluctuations in probe beam, and therefore image, intensity. Later, we replaced the waveplate with the glass rod and solenoid. Ultimately, we created a sensor with angular sensitivity as small as 0.6×10^{-4} degrees, corresponding to magnetic fields on the order of $0.5 \mu\text{T}$. The device is applicable in structural defects detection, including settings that require small size and mobility and that might not be electronics-friendly.

Chapter 1

Introduction

1.1 Background

Magnetic field sensors have a wide variety of applications, making them an ongoing and prolific topic of inquiry. The uses of magnetometers range from highly applied, such as geological surveying and heartbeat monitoring [1], to highly fundamental, such as in measurement of physical constants. As technology advances, the need for increasingly sensitive magnetometers continues to propel research on these devices. We propose a highly sensitive, optical magnetometer that boasts enhanced imaging capabilities and a mobile configuration. With these qualities, our device's applications will center on defect detection in metal components, which can be useful in areas like infrastructural health analysis, vehicle design, and passenger safety.

1.2 Motivation and Theory

Currently, in the field of magnetometry, there exists a tension between achieving high sensitivity and maintaining a compact, mobile device configuration. This problem is often accompanied by a quest for spatial resolution of the field, which can be extra, desirable or even essential, information. Over the course of more than half a century, various types of optical magnetometers have emerged and been enhanced

to meet these challenges. Optical magnetometers can reach unprecedented levels of sensitivity by probing atoms in a magnetic field with light. They then use the atoms' and/or light's response to detect the field.

We will focus on one traditional, elegantly simple type of optical magnetometer called a Faraday rotation magnetometer. This specific device relies on the change in polarization that light undergoes when it interacts with atoms in a material that is exposed to a magnetic field. This phenomenon is called Faraday rotation. Specifically, in a transparent material like glass, a magnetic field, directed along the direction of light propagation will rotate the light's polarization by an angle ϕ , as shown in Figure 1.1 [2]. This angle of polarization rotation is proportional to the magnetic field B according to

$$\phi = C_V B l, \tag{1.1}$$

where C_V is the Verdet constant, which depends on the material, and l is the length of the material along which the light propagates.

In the traditional setup, linearly polarized light shines through the sensor material, often a crystal, and the light's polarization is rotated by the angle, ϕ , proportional to B . This angle can be measured, for example, by a photodiode placed after a polarizer at the output. Note, however, that for a reasonably sized magnetic field, the change in polarization is quite small. Indeed, in the setup shown in Figure 1.1, a magnetic field of 11.1 mT leads to a polarization rotation on the order of 10^{-4} rad, or about 0.005° [2].

A significant drawback of this traditional setup is that using a photodiode as the sensing device does not provide any *spatial* resolution of the magnetic field. One solution is to replace the photodiode with a CCD camera to image the field [3]. The downside here is that adding a camera makes for a bulky sensor that would not be

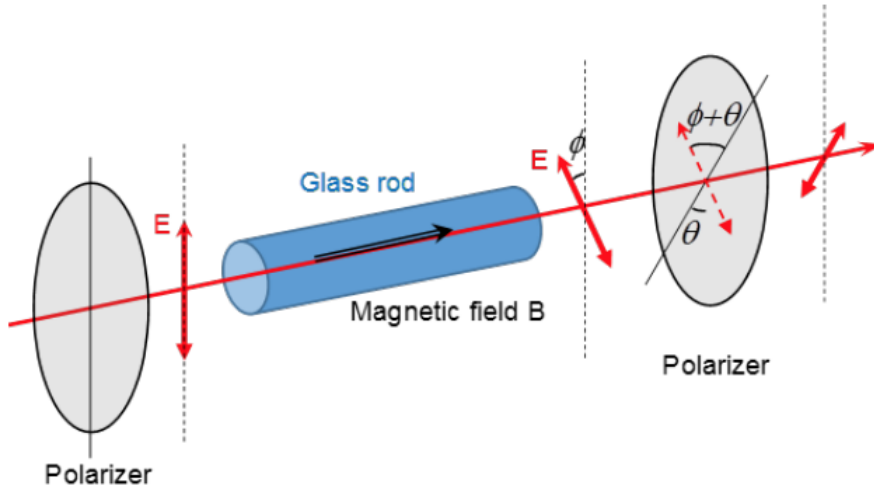


Figure 1.1: [2] A visualization of Faraday rotation, where light (red arrow) of a polarization θ traveling through a transparent medium subject to an external magnetic field, B , will rotate by an angle, ϕ , proportional to B . Note that ϕ is much smaller than shown here.

compatible with, say, chip-scale manufacturing [4].

To solve this problem, we instead separated the camera physically from the Faraday rotation magnetometer using a multimode optical fiber. Fibers are inexpensive and will allow a portable, flexible device configuration. When light travels down a multimode fiber, its different modes scatter differently, creating a kind of “speckle” pattern intensity profile. These patterns appear random but note that they can be theoretically predicted using the coupling conditions of the fiber. Figure 1.2 shows a comparison between the intensity profiles of a traditional Gaussian beam travelling through air and of a Gaussian beam after travelling down a multimode fiber.

Thus, when the laser’s polarization changes, its intensity does not change, but the speckle patterns do change, because of interference inside the fiber. Figure 1.3 shows a comparison of the speckle patterns for a 0° and 45° polarization. Therefore, we can use the differing speckle patterns to identify the beam polarizations corresponding to different magnetic fields. Using a fiber eliminates the need for the polarizer that the

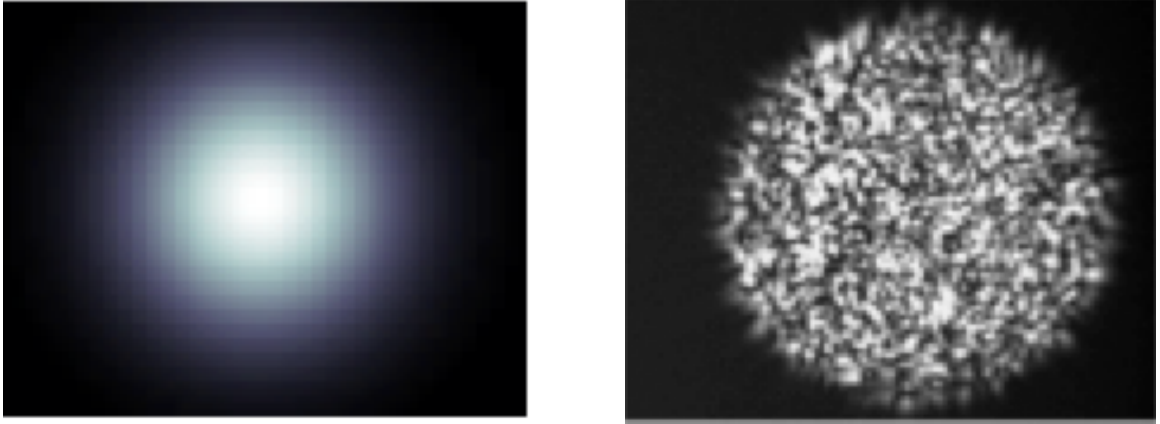


Figure 1.2: On the left is an example of a Gaussian laser beam intensity profile with no modifications. On the right is the scrambled “speckle” intensity profile resulting from sending the Gaussian beam travels down a multimode fiber.

conventional setup requires. Furthermore, it allows remote magnetic field detection, in which the detector can be at significant distance from the sensor, safe from interfering with or being affected by the measured fields.

The drawback, though, is that the speckle patterns are complicated images. Also, when the change in polarization is small, as it is when magnetically-induced, the image features only change by a small amount, which is almost entirely undetectable visually (See Chapter 3).

This problem invites the use of artificial intelligence (AI). Ultimately, we trained an image classification algorithm to recover the laser polarization from these speckle patterns. This AI approach increases data processing power for rapid field measurement and analysis.

To summarize, we built an optical magnetometer employing AI for magnetic field measurement. This approach combines the structural and informational advantages of a fiber + camera setup with the efficient, unbiased data processing power of AI for enhanced spatial resolution and physical compactness. We hope to achieve sensitivity

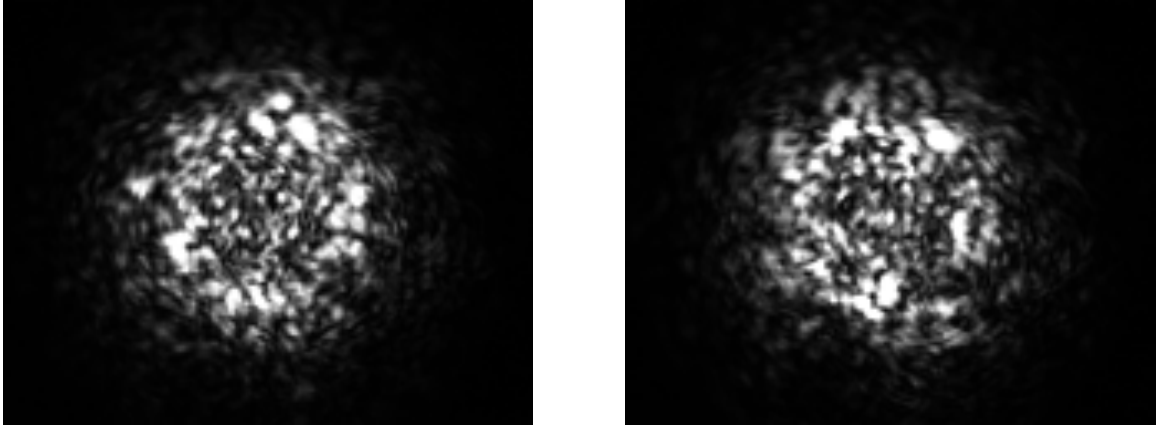


Figure 1.3: A comparison of the multimode fiber output “speckle” patterns for 0° (left) and 45° (right) polarization angles.

on the nanoTesla range for applications in structural defects detection of, say, air and space vehicles, buildings and bridges, or pipelines and storage tanks. Our device also offers the possibility of replacing the Faraday rotation apparatus (glass rod) with other media for enhanced sensitivity. One option is a crystal defect like a Nitrogen-Vacancy (NV) center [5], [6], which would allow for further compactness and possibly miniaturization to the chip scale.

The outline of this paper is as follows: In Chapter 2, we will explain the experimental design, walking through each part of the device, from the physical setup to the image classification algorithm used in this work. Interspersed throughout is theory on image classification to guide an understanding of our methods. Chapter 3 explains some results that are foundational to the current work, but that were obtained during preliminary investigations done in Spring 2021. This work consisted of constructing the image classification algorithm and optimizing it to our experimental conditions. Chapters 4 and 5 describe early efforts to improve algorithm sensitivity to external factors such as image intensity, and to establish a benchmark for angular sensitivity upon which to improve. These two chapters describe a setup wherein we used a half

waveplate to generate changes in laser polarization, rather than an actual magnetic field. Finally, in 6, we implement the Faraday rotation magnetometer. Here, we try various methods to create the highest possible angular sensitivity so as to measure the smallest magnetic fields.

Chapter 2

Experimental Design: Optical Setup, Data Acquisition, and Image Classification Algorithm

In this chapter, we will describe the experimental design, starting with the physical apparatus setup and then outlining the AI algorithm along with some useful theory on image classification.

2.1 Experimental Setup

We will now describe the experimental setup, shown in Figure 2.1. We conducted many experiments using a preliminary setup, with results discussed in Chapters 3, 4, and 5. We then transitioned to the final Faraday magnetometer setup used in Chapter 6. We describe both setups in the next two sections to facilitate a broad understanding of our fiber+camera+AI approach before discussing the grittier details of building, testing, and improving the algorithm.

2.1.1 Preliminary Waveplate Setup

As a foundational step toward constructing the actual magnetometer, we used a half waveplate (Figure 2.1 b)) to polarize the laser beam that propagated through the

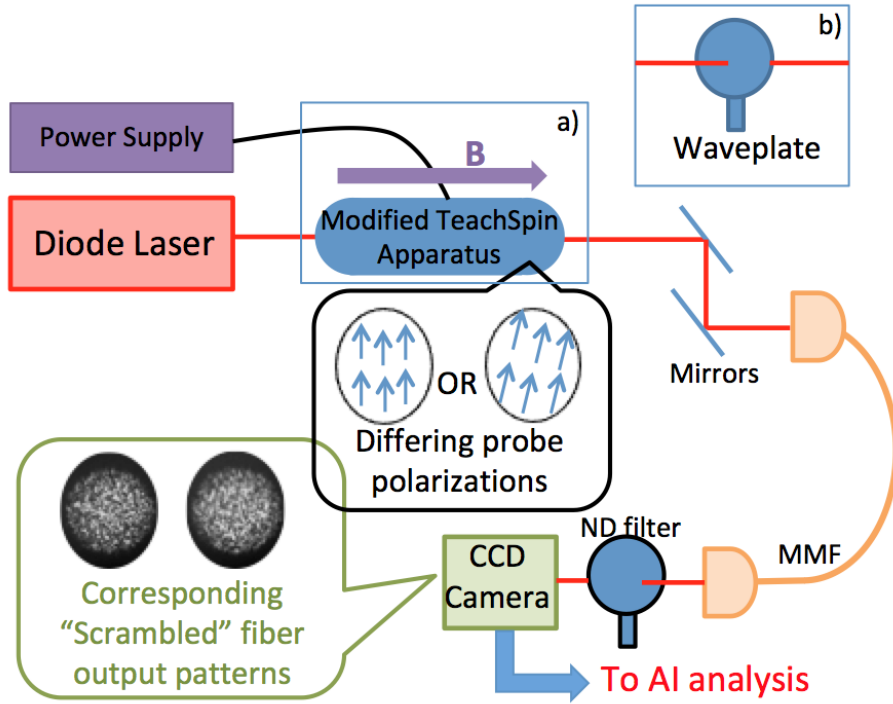


Figure 2.1: The experimental setup for the optical fiber-linked magnetometer using AI for magnetic field measurement. Either a) (the Faraday apparatus) or b) (the half waveplate) was used to change the laser polarization.

multimode fiber. A waveplate consists of a material, usually a crystal like quartz or mica, that breaks linearly polarized light into two components, one parallel and one perpendicular to the device’s own optical axis. These two components will travel at slightly different speeds through the crystal due to a polarization-dependent refractive index “birefringence”. Thus, the device introduces a phase difference between these two components, so that when they are recombined, the light’s overall polarization has rotated by a certain angle. The experimentalist can rotate the device’s optical axis to achieve the desired polarization rotation. A half waveplate introduces a 180° phase shift, and thus is generally used to rotate linearly polarized light’s polarization by such a desired amount. (By contrast, a quarter waveplate introduces a 90° phase shift, thus converting linearly polarized light into circularly polarized light). In this

work, we used a half waveplate only.

2.1.2 The Faraday Apparatus and Setup

The Faraday magnetometer setup is shown in Figure 2.1 a). For the Faraday rotation apparatus itself, we used a modified version of TeachSpin Inc.’s undergraduate laboratory Faraday rotation apparatus. The apparatus contains a 10 cm long, 5 mm diameter SF-59 glass rod surrounded by a 15 cm long solenoid that connects to a power supply. The value of C_V depends on wavelength, but for 650 nm light, $C_V = 23 \text{ rad}/(\text{T}\cdot\text{m})$. For the solenoid, the calibration, given by the manufacturer [2], between input current, I , and magnetic field, B is

$$B = (11.1 \text{ mT/A}) \cdot I. \quad (2.1)$$

However, when we measured current versus change in polarization in the lab, we obtained a calibration for I vs. ϕ , which was slightly different, causing our values of B obtained using Equation 1.1 to differ from those obtained using 2.1. Differences here likely result from the fact that C_V depends on wavelength. The wavelength of our laser was 780nm, when the value for C_V was reported at 650nm. Figure 2.2 shows our calibration of current vs. polarization for the apparatus. Placing a polarizer after the TeachSpin apparatus, we recorded the output intensity for different currents. Using the fact that intensity, i , is related to polarization, ϕ by

$$i = i_0 \sin^2(\phi), \quad (2.2)$$

where i_0 is the maximum intensity possible, we converted intensity values into polarization rotations. We fit the data according to an absolute value function because the relationship between polarization rotation and current is linear, but we included “negative” values of current as well. These negative current values indicate crossed

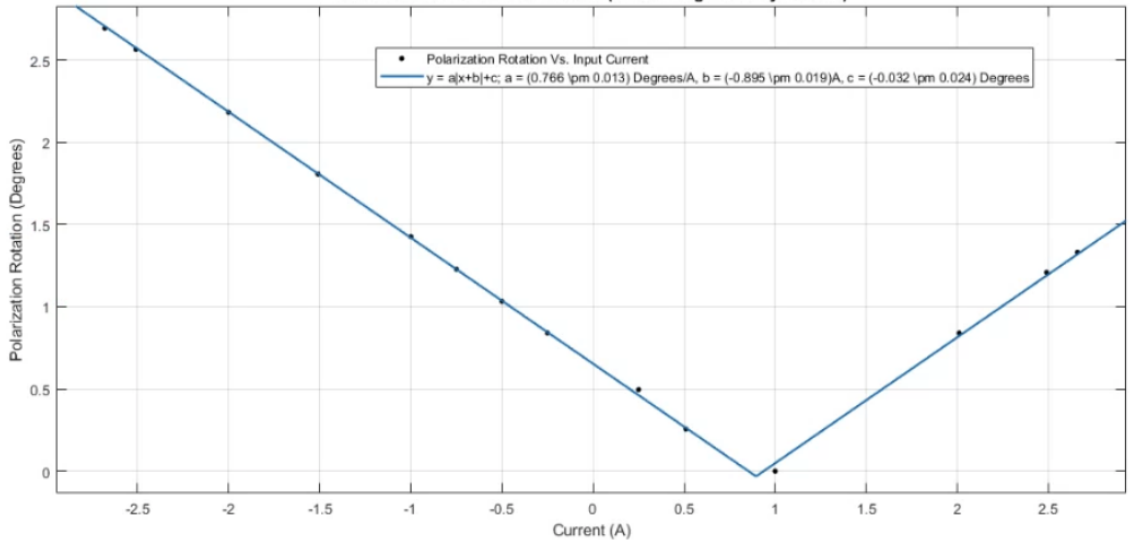


Figure 2.2: Calibration of the TeachSpin Faraday rotation Apparatus: Polarization Rotation, ϕ , vs. Input Current, I . All intensity values used to calculate ϕ were normalized by $i_0 = 2000 \mu\text{W}$, the maximum intensity transmitted through the polarizer.

wires in the power supply, which were used to obtain a wider range of polarization angles.

For further apparatus specifications, see the TeachSpin Faraday Rotation manual, [2].

For this work, we removed the laser, polarizer and photodiode that come attached to the TeachSpin apparatus. We then inserted the remaining part into our setup, as shown in Figure 2.1 a) pictorially and Figure 2.3 physically. Thus, our experimental setup is as follows: A 780.24 nm diode laser travels through a modified version of the TeachSpin Faraday rotation apparatus. After the light travels down the multimode fiber (MMF), a neutral density (ND) filter reduces the beam's intensity to a level appropriate for imaging. A CCD camera images the resultant beam profile. The images are then sent to the image classification algorithm. The MMF, ND filter, and CCD camera are all contained in a cardboard box to shield from temperature changes and ambient light.

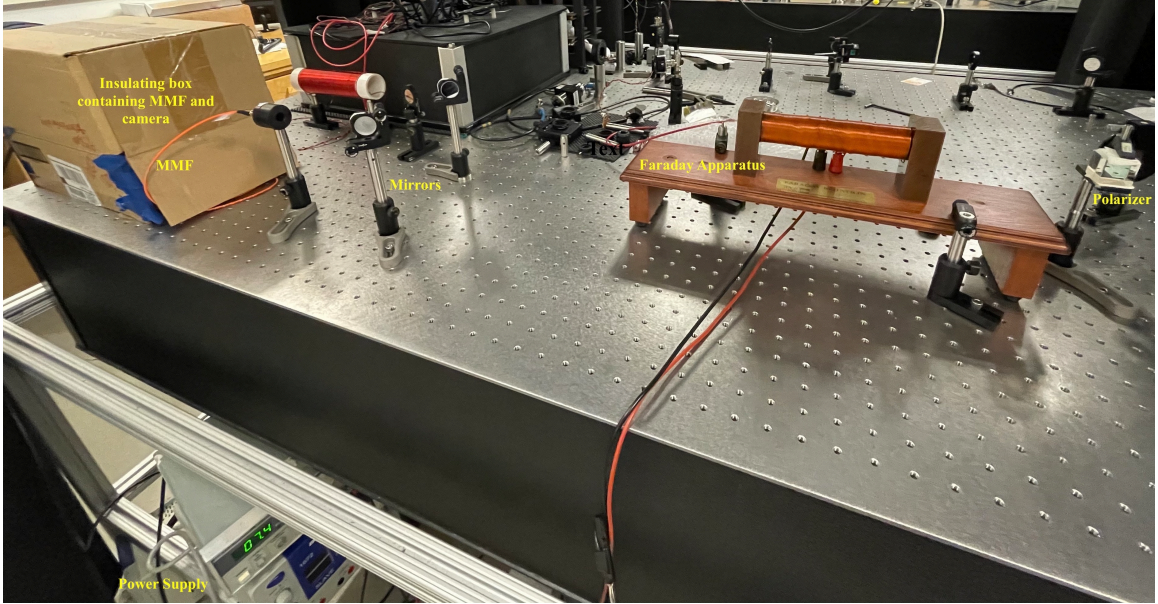


Figure 2.3: The experimental setup, consisting of TeachSpin’s Faraday rotation apparatus.

The data collection procedure for algorithm training and testing consisted of setting the power supply to different current values and collecting a given number of speckle pattern images for a given polarization.

The images taken were originally 1024×1280 matrices. To avoid long algorithm run times, the images were then re-scaled to 256×320 matrices using an interpolation procedure we developed in MATLAB in 2019 (See A.1). The images were normalized so their matrix entries fall between 0 and 1. They were then converted to png files, split into training and testing sets, and sent to the algorithm for learning and classification.

In the following section, we will outline some basic theory of image classification and then describe the algorithm constructed for this work.

2.2 The Image Classification Algorithm

Borhani et al.[7] and Wang et al. [8] have shown that deep learning can be used to classify the “speckle patterns” that result from the propagation of an input image through a multimode fiber. We therefore propose to use a similar approach to identify the different speckle patterns that result from different magnetically-induced probe beam polarizations traveling through a multimode fiber.

2.2.1 Image Classification Theory

The goal of image classification is to train a computer to sort images into two or more categories based on previous experience, or “learning.” We used a convolutional neural network (CNN) as our deep learning platform. A CNN distinguishes images by assigning importance, or weight, to the images’ distinguishing features. The network’s structure mimics that of a human brain, with different layers connected by neurons. Below we will outline a typical image classification process, which was followed in this work. We will start with the step directly following data acquisition and describe the process up to algorithm testing. Helpful background information on convolutional networks can be found in the overview paper by Albawi et al.[9]. Other more informal, but still informative sources include webpages by Deshpande [10] and Saha [11].

Data Pre-Processing

After acquisition, the image data is divided into training data and testing data. The images fall into two or more categories, where the correct category is known for each datum. Typically an equal number of images from each category is present in each of the training and testing sets. The training set is then fed into the CNN, while the testing set is set aside. In this work, we also impose circular masks on our data to block out any non-beam regions of our images. We explain this process further in

Section 6.3.

CNN Structure

Our CNN followed the traditional structure of a number of alternating convolutional and pooling layers, followed by a fully connected layer. The relevant specifics of these layers are outlined below.

Convolution. Traditionally, the first layer of this network is the convolutional layer. During convolution, a filter is superimposed onto the input image. A filter is simply a matrix of weights that is smaller than the original image. It may also be referred to as a neuron. The filter is then convolved with the region of the image that it covers. That is, element-wise multiplication is performed between the filter and the image region. The multiplications are then summed into a single value that represents that region of the original image. The filter then slides over all regions of the image to form a feature map that describes the image as a whole.

Maximum Pooling Layer. A pooling layer further reduces the size of the feature map in order to reduce the computational power required. A maximum pooling layer groups the values of the feature map and takes the maximum value in each group as the representative value. This way, the maximum pooling layer extracts dominant features in the image. It also serves as a noise suppressant.

Fully Connected Layer/Classification. After one feature map is formed, convolutional and pooling layers may be repeated any number of times to extract lower level features from the images. Once enough convolutions and poolings have occurred, a fully connected layer is installed. The fully connected layer analyzes the previous layer and determines to which image category the feature maps produced during convolution most likely belong.

Training and Testing

Training.

In order for convolution to produce accurate results, the algorithm must run through the above neural network structure multiple times, evaluating the “learning” process after each pass. Initially, the filter weights obtained during convolution are assigned randomly. However, during a process of “backpropagation,” the algorithm evaluates the accuracy of the fully connected layer output. During each convolutional cycle, the algorithm generates a loss function that determines which filter weights are contributing most to errors in classification accuracy. The loss function used in our algorithm was the Cross Entropy Loss. This function describes the difference between the algorithm’s prediction and the correct prediction. Based on the features the algorithm detects during convolution and comparison to the original images, it assigns a probability that those features belong to a given category. If it assigns a high probability to a category that is actually incorrect, the loss will increase. As the algorithm generates this loss function, the lossy weights are updated in the direction that minimizes classification error. This process repeats for as many iterations, or “epochs,” as desired until the algorithm has adequately learned to distinguish between the image categories.

There are a few training parameters that govern the training process and that may be optimized for best learning. The parameters relevant to this work will be outlined in the following paragraphs.

Number of Training Iterations, or “Epochs.” The number of training iterations is simply the number of times the algorithm runs through the CNN during training. More epochs leads to better performance, creating a trade-off between training run time and accuracy.

Filter Size. The size of the convolutional filters may be adjusted to capture

more general or more specific image features. Traditionally, filter size is increased in consecutive layers as the network pieces together bigger and bigger patterns. Again, larger filters may lead to higher algorithm run times, but may also improve accuracy.

Number of Filters Per Layer. The number of filters used in each layer may also be adjusted. Too few filters in a layer leads to poor performance, while too many increases run time.

Learning Rate. The learning rate is perhaps the most important parameter to optimize when constructing a CNN. During backpropagation, the algorithm takes steps backwards and forwards along the loss function it generates to find the point of minimal error. The size of these steps is called the learning rate. The learning rate is a positive value between 0 and 1. A learning rate of 0.01 means that the lossy weights are updated by 1% of the weight error during each training iteration. A larger learning rate allows the algorithm to run faster, but it risks converging on a suboptimal set of filter weights. A smaller learning rate leads to slower run times, but better algorithm performance. However, a learning rate that is too small may lead to the algorithm getting stuck in a local minimum in the loss function. Thus, proper configuration of the learning rate is crucial to algorithm performance [13].

Testing.

The final step in the image classification process is to present the algorithm with images similar to those it was trained on, but that it has never seen before. These images are passed through the CNN, and the ability for the algorithm to correctly identify these new images is evaluated.

Measurements of Success

Algorithm efficacy should be evaluated during both the training and testing processes.

Training Plots. In the training process, the algorithm will generate a learning curve, which is a plot of classification accuracy vs. training epoch and loss vs. training epoch. Increasing accuracy and decreasing loss over time will indicate successful training. Figure 2.4 shows an example of a suboptimal learning curve and an optimal learning curve. Ideally, the accuracy will increase steeply to indicate fast learning.

Usually, accuracy and loss should roughly mirror each other. However, they are truly different quantities, and therefore both are needed to obtain an accurate understanding of the model. Accuracy is defined as a binary true/false designation made at the end of each training cycle when evaluating the algorithm’s prediction. So, it is a discrete variable. Loss, though, is a continuous function that describes how far away the algorithm’s prediction is from the “correct answer.” A common instance where accuracy and loss will not fully match up is when the accuracy has plateaued at a high value, but loss is continuing to decrease. This phenomenon occurs when the algorithm has become confident, but the margin it has created between the two categories is narrow. Therefore, correct predictions may happen by chance. Allowing the loss to decrease to match the high accuracy will lead to increased algorithm robustness and thus better performance in testing [12].

Output Performance Accuracy. If training is successful, the algorithm is able to identify correctly the images included in the training set. During testing, we must evaluate the algorithm’s ability to identify images it has never seen before. The method of evaluation used in the work was a confusion matrix, a table showing, as a percentage, how often the algorithm correctly identified the images in the given category, and how often it confused one category for another. An ideal confusion matrix will have 1s on the diagonal and 0s elsewhere, indicating that the algorithm matched all images to their correct categories. Figure 2.1 shows a hypothetical confusion matrix. In this case, the algorithm never confused Class 1 for Class 2, but it

	Predicted Class 1	Predicted Class 2
Actual Class 1	1	0
Actual Class 2	0.44	0.56

Table 2.1: An example of a confusion matrix for an algorithm distinguishing between a hypothetical “Class 1” and “Class 2”. The row shows the algorithm’s prediction of the class of the image in testing and the column shows the actual class of the image. Thus, the third row, second column shows the percentage of times the algorithm predicted class 1 when the image was actually class 2.

confused Class 1 for Class 2 44% of the time. In other words, the algorithm correctly identified images in Class 1 100% of the time, and it correctly identified images in Class 2 56% of the time.

In this work, the overall output accuracy was computed by taking the average of the diagonal elements.

2.2.2 The Algorithm

We constructed the image classification algorithm used in this project in Spring 2021 as preliminary work. The flow chart in Figure 2.5 gives a visual representation of the algorithm used currently. The final algorithm consisted of 3 convolutional layers, each followed by a normalization layer and a maximum pooling layer. The first convolutional layer consisted of 5 filters of size 1x1 pixel, the second of 15 filters of size 6x6 pixels, and the third of 40 filters of size 12x12 pixels. After the three sets of convolutional, normalization, and pooling layers was a fully connected layer and then a classification layer.

The training parameters used were as follows:

- Filter weight optimization method: Stochastic Gradient Descent. This method finds the minimum of the loss function, calculating the derivative of the function in steps. Technically, the algorithm must follow this process for every single im-

age feature, but this stochastic method does so only on a collection of randomly chosen features in order to reduce computational time [14].

- Initial learning rate: 10^{-5}
- Number of epochs: 100 (subject to change depending on test conducted)
- Shuffle training data every epoch to ensure an unbiased training

(See A.2 for full code).

During each training, MATLAB generated a learning curve, which we used to evaluate training progress. After testing, the algorithm output a confusion matrix for the two image categories. The overall performance accuracy was evaluated by taking the average of the diagonal elements in this matrix. Thus, the algorithm performance was characterized by the average percentage of times the algorithm identified an image correctly.

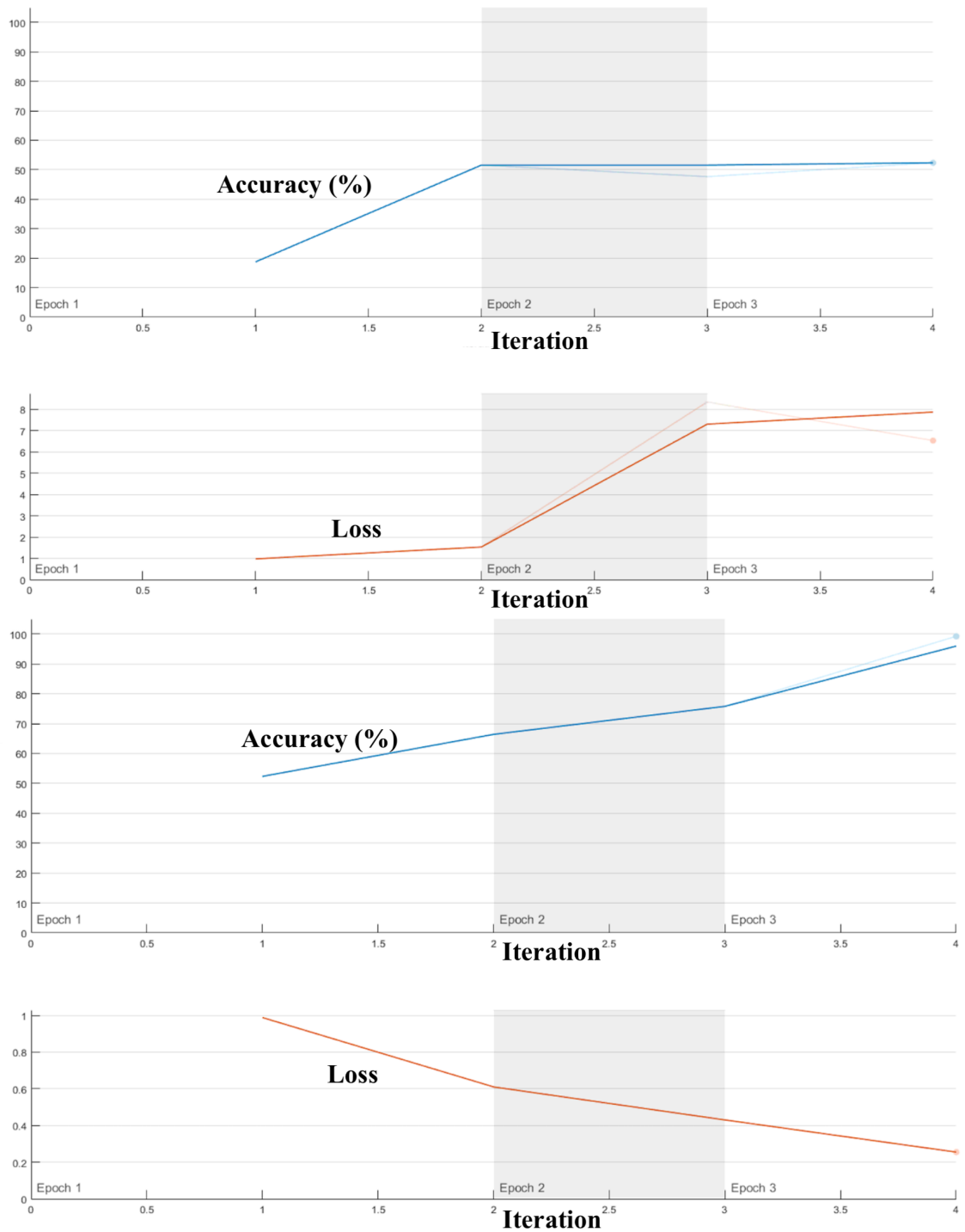


Figure 2.4: On the top is an example of a suboptimal learning curve, and the bottom is an example of an optimal learning curve. Plotted are training accuracy (blue/upper half of graph) and loss (red/bottom half) vs. training iteration or “epoch.”

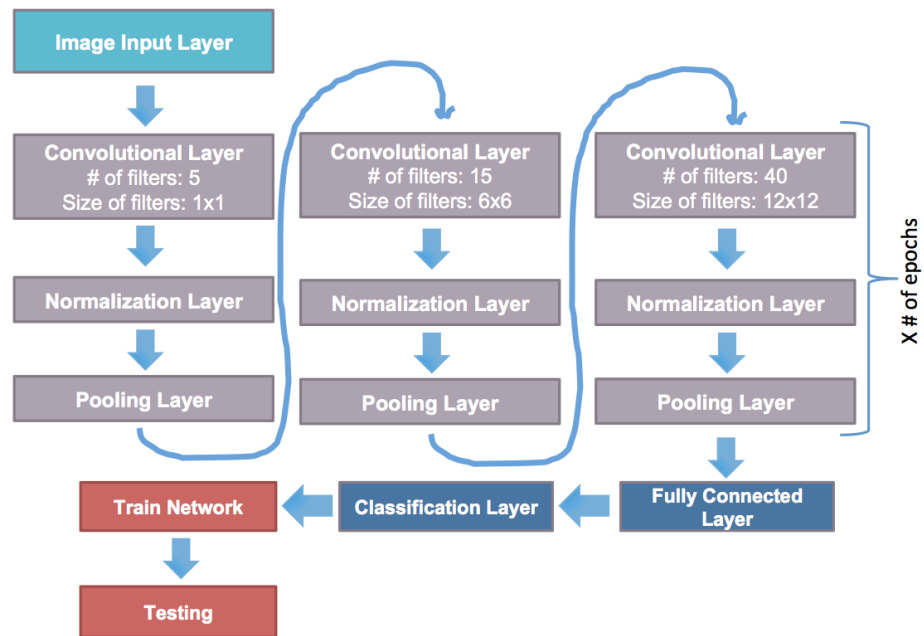


Figure 2.5: A flow chart representation of the image classification algorithm used in this work. Each colored box represents a layer in the neural network.

Chapter 3

Preliminary Work to Achieve Image Recognition

Before going into current work, we must outline some significant results of preliminary work conducted in Spring 2021. They are

1. Development of a robust image acquisition procedure that is used currently.
2. Achieving algorithm recognition of images corresponding to changes in polarization as small as 1° .

We will outline these achievements below.

Work done in Spring 2021 consisted of constructing and optimizing an image classification algorithm to identify different laser beam polarizations. The machine learning algorithm was constructed in MATLAB to distinguish between two image categories, one polarization angle and another. Figure 3.1 shows sample images of the speckle patterns taken at four different polarization angles: 0° , 6° , 10° , and 20° . Note the increasing similarities between image features as the difference in angle decreases. For a 6° difference in polarization, the features become difficult to distinguish visually.

Therefore, we expected the algorithm to identify large differences in polarization easily. Indeed, we did an initial check of training the algorithm to distinguish between a polarization angle difference of 45° , and it was 100% accurate. As real

magnetically-induced changes in polarization are fractions of degrees, the challenge would be to train the algorithm to distinguish between images that are much more similar. However, against our expectations, the initial algorithm was 100% accurate in distinguishing differences in polarization as small as 2° . This result seemed too good to be true. We concluded that the success of the identification was due to lack of sufficient variation in the image data - essentially, the algorithm was being trained on 100 or so identical images and then being tested on another identical image. This scenario does not model reality well. So, we developed a data collection procedure that introduces a realistic amount of variation in the speckle patterns.

First, we set the waveplate to a given polarization angle and recorded the desired number of images, usually 25. We captured an image every 5 seconds to allow for natural evolutions of the speckle patterns with time. This drifting of the patterns can be caused by factors such as temperature changes or mechanical stress/strain on the fiber from small vibrations or pressure perhaps. After recording the images of a given polarization angle, we rotated the waveplate to the other desired angle and captured another 25 images, one image every 5 seconds. We rotated the waveplate back to the original angle and repeated this process 4 times. Thus, our final training set consisted of 125 images at each polarization. Unlike in the case of acquiring 125 images of a given polarization all at once, we found that rotating the waveplate back and forth during data acquisition introduced just enough lack of exact reproducibility to provide adequate variation in our images.

With this new data collection procedure, our algorithm performance dropped to around 80% when distinguishing a 2° polarization difference. Therefore, we concluded that the method of rotating the waveplate back and forth was effective at introducing variation into our images.

Following this drop in accuracy, we optimized the algorithm training parameters

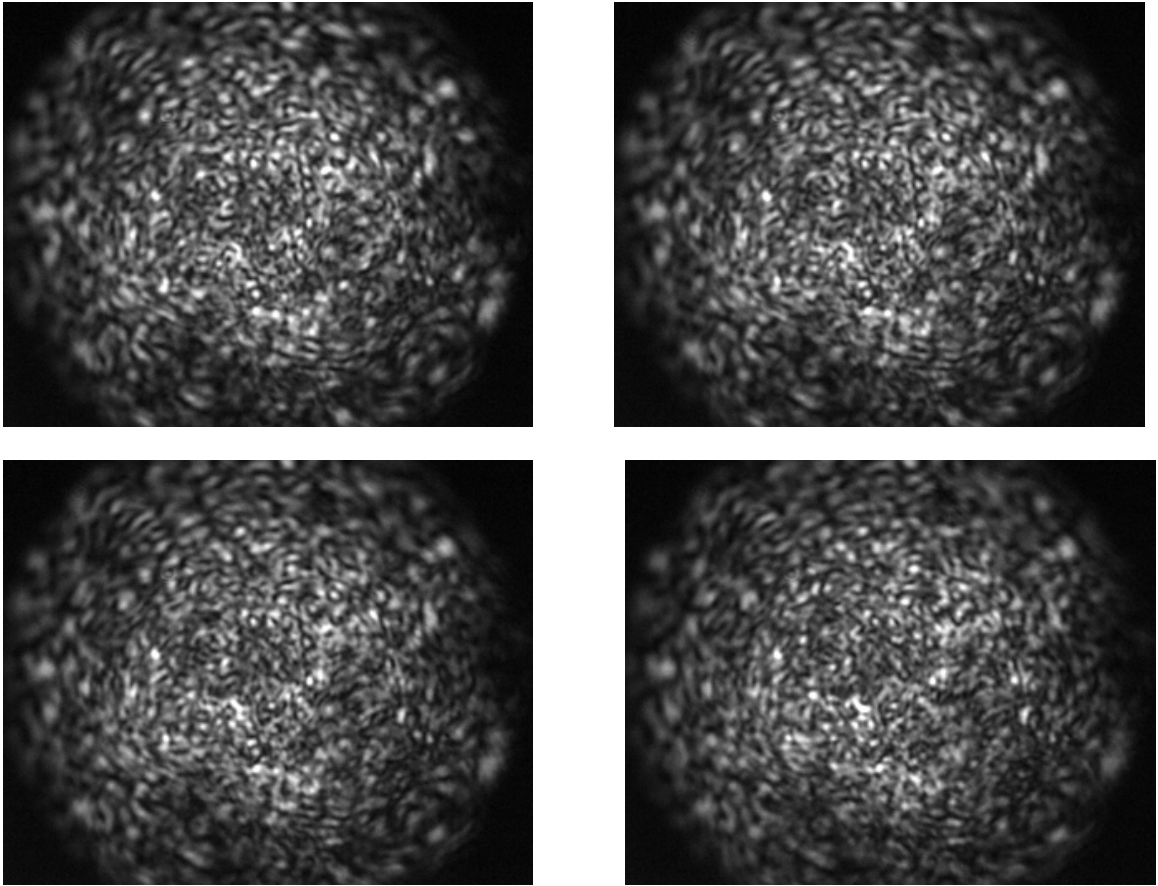


Figure 3.1: Shown above are sample images of speckle patterns at four different polarizations, 0° (top left), 6° (top right), 10° (bottom left), and 20° (bottom right), used in algorithm training.

until it was able to distinguish a 1° difference in polarization angle with 96% accuracy.

Chapter 4

Current Work to Reduce Algorithm Sensitivity to Intensity

The goal for the current work is twofold: we now hope to identify changes in polarization that are smaller than 1° in order to achieve magnetic field measurement on the nanoT scale. We also want to render the algorithm insensitive to external factors that may affect the algorithm’s “interpretation” of the image data, such as changes in image intensity.

4.1 Basis Algorithm Tests

Early in the semester, we set up the experiment as outlined in Section 2.1. As a basis, we performed preliminary tests to ensure comparability to results obtained in the spring. We used the “back and forth waveplate rotation” method to obtain image data at 40° , 20° , 10° , 2° , 1° , 0.5° , and 0° . For each angle, the training set contained 125 images, and the testing set contained the same amount. We then trained the algorithm to distinguish between each angle and 0° .

We found that the algorithm was highly accurate in recognizing polarization angle differences of 40° , 20° , 10° , 2° , and 1° . We found that a 0.5° difference in polarization angle, however, reduced the accuracy to 78%. Interestingly, when the number of

epochs was doubled to 200, the algorithm was able to distinguish a 0.5° difference with 90% accuracy.

4.2 Exploring Algorithm Sensitivity to Laser Power

Realistically, the laser's power may fluctuate, leading to changes in intensity of different features in the speckle patterns. We investigated the algorithm's response to these changes so that we could ultimately prevent them from hindering algorithm performance. A typical laser power fluctuates by a few percent, however, we wanted the algorithm to be insensitive to even greater changes for maximum robustness. In the following sections, we will outline a number of tests that we conducted to evaluate, and ultimately minimize, the effect of changing image intensity on algorithm performance. Since we had shown the algorithm to have a high performance accuracy at differences in polarization as small as 1° , we conducted all intensity tests with 0° and 1° as our two image categories.

4.2.1 Maximum Difference in Power of 4 mW

Using the back and forth waveplate rotation method, we collected 5 sets each of 0° polarization and 1° polarization. The first set contained 125 images at 1 mW laser power, the second 125 at 2 mW, and so on until 5 mW. We used a variable ND filter to adjust the power incident into the multimode fiber. We then trained the algorithm to distinguish between 0° and 1° at 1 mW and between 0° and 1° at 5 mW, both for 100 epochs. The algorithm was later tested on images of these same polarizations, but taken at different powers between 1 and 5 mW. The goal was to see how big of a difference in power the algorithm could tolerate before failing to identify the different polarizations. Figure 4.1 is a logical table showing the algorithm performance accuracies for all possible combinations of training and testing on different powers.

		Training Power (mW)	
		1	5
Testing Power (mW)	1	1	0.536
	2	0.500	0.520
	3	0.500	0.520
	4	0.500	0.652
	5	0.500	1

Figure 4.1: Image Classification Accuracies for distinguishing between a fiber input beam polarization 0° vs. 1° polarization. Accuracies range from 0 (0% accurate) to 1 (100% accurate). The first row is the training image powers and the first column in the testing image powers. Thus, shown in all other cells are classification accuracies for all possible combinations of training on 1 mW or 5 mW and testing on 2, 3, or 4 mW. Training and testing on the same powers are shaded out.

The algorithm is 50-65% accurate in all cases. A 50% accuracy means the algorithm is essentially guessing, and a 60% accuracy is not much better. We had expected that the algorithm would have been more successful, at least for small differences in power, like 1 mW.

There are two main possibilities for an image classification algorithm to return a 50% performance accuracy: 1) the algorithm did not learn during the training process; 2) the algorithm learned the training images well, but the algorithm interprets the testing images as very different from the ones familiar to it. We eliminated possibility 1) by examining the learning curves for training on 1 and 5 mW. As shown in Figs 4.2 and 4.3, both trainings were successful, with the performance accuracies increasing to near 100% and the loss decreasing to about 0.4.

We concluded that possibility 2) was likely true. A 4 mW difference in power was

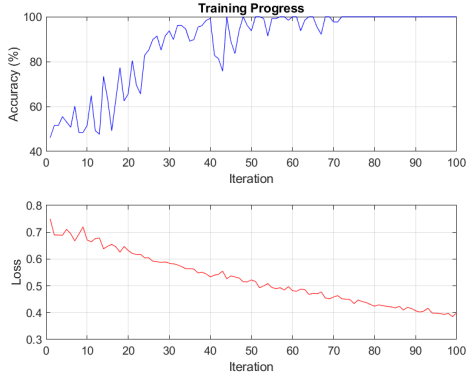


Figure 4.2: Learning Curve for 0° vs. 1° polarization, at 1 mW image intensity. The x axis is the number of training iterations, and the y axis is training accuracy (above) and training loss (below).

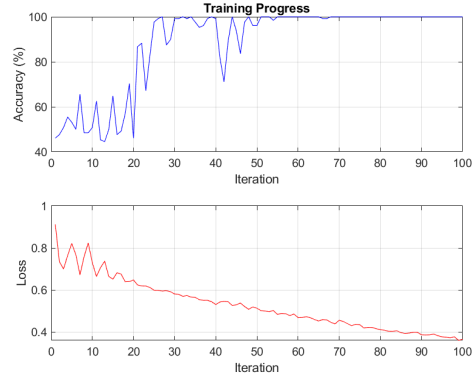


Figure 4.3: Learning Curve for 0° vs. 1° polarization, at 5 mW image intensity

too high for the algorithm to withstand. To investigate this hypothesis further, we conducted similar trainings, focusing on differences in power of 1.5 mW and smaller. These tests are outlined in the next section.

4.2.2 Maximum Difference in Power of 1.5 mW.

We examined the effect of smaller differences in power on algorithm performance. We expected that these smaller differences would lead to testing images that were more recognizable to the algorithm. Using again the back and forth waveplate rotation method, we collected 4 sets each of 0° polarization and 1° polarization. The first set contained 125 images at 0.5 mW laser power, the second 125 at 1 mW, the third 125 at 1.5 mW, and the fourth 125 at 2 mW. As before, we trained the algorithm to distinguish between 0° and 1° at a certain power for 100 epochs, and tested the algorithm on images of these same polarizations, but taken at a different power. Figure 4.4 shows the logical table representing the algorithm performance accuracies

		Training Power (mW)			
		0.5	1	1.5	2
Testing Power (mW)	0.5	1	0.53	0.42	0.50
	1	0.80	1	0.50	0.50
	1.5	0.56	0.61	1	0.50
	2	0.84	0.94	0.50	1

Figure 4.4: Image Classification Accuracies for 0° vs. 1° , training and testing on images taken at 0.5, 1, 1.5, and 2 mW. The first gives the training images' power, and the first column the testing images' powers. Thus, shown in the other cells are classification accuracies for all possible training and testing combinations.

for all possible combinations of training and testing on different powers.

The algorithm was about 60% accurate or worse, except for three exceptions:

- Train on 0.5 mW, test on 1 mW: 80% accurate
- Train on 0.5 mW, test on 2 mW: 84.4% accurate
- Train on 1 mW, test on 2 mW: 94.4% accurate.

We noted also that switching the training and testing sets did not necessarily guarantee the same performance accuracy. For example, training on 1 mW and testing on 2 mW was 94.4% accurate, but the reverse scenario was 50% accurate. Again, we eliminated possibility 1) of poor training by examining the learning curves corresponding to these results. As shown in Fig 4.5, the trainings for all four values of power were successful, with the performance accuracies increasing to near 100% and the loss decreasing to 0.5 or below.

Therefore, we hypothesized that the algorithm is so sensitive to changes in power that even a 0.5 mW difference in image intensity rendered the testing images unrec-

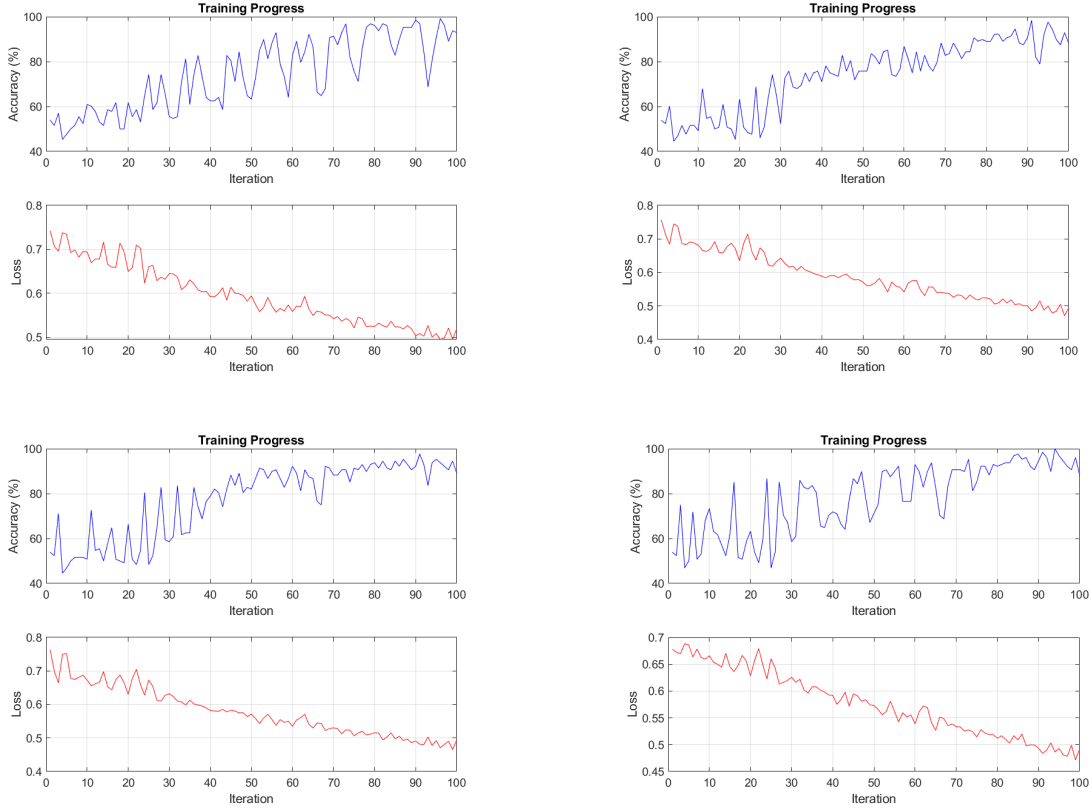


Figure 4.5: Learning Curves when training to distinguish between 0° and 1° polarization, where each plot shows trainings conducted on images taken at four different fiber input powers, 0.5, 1, 1.5, and 2 mW. The top left image shows the training conducted on 0.5 mW images, the top right 1 mW, the bottom left 1.5 mW, and the bottom right 2 mW.

ognizable. As of now, we are unable to explain the three exceptions, though it is interesting that 2/3 of them occur when training on 0.5 mW, the dimmest setting.

4.2.3 Training on Mixed Power Datasets

Because our algorithm seemed very sensitive to small changes in image intensity, we tried to improve performance by training on a set of images taken at *multiple* laser powers. We used data acquired in the previous section (0.5, 1, 1.5, 2 mW) to conduct these tests. We started by including 2 different values of power in the training set.



Figure 4.6: Learning curve for 0° and 1° , where the training set consisted of images taken at 0.5 mW and 2 mW.

The training set contained 120 images of 0° , divided into 60 images taken at one power and 60 at another, and 120 images of 1° , with 120 image divided evenly among those same two powers. We trained on all possible combinations of two of four power levels. We then tested on a single power of the four, where the testing set contained 125 images of each angle, as before.

When first training on a mixed power dataset, the learning curves were suboptimal. For example, when training on a set containing images at 0.5 and 2 mW, the plot did not converge to a high accuracy within 100 epochs, see Figure 4.6. This poor learning led to a performance accuracy of 66.4% when testing on images at 0.5 mW. So, we doubled the number of epochs for subsequent trainings and saw significant improvement in the learning curves as shown in Figure 4.7.

Figure 4.8 shows the results of these trainings with 200 epochs. Training on

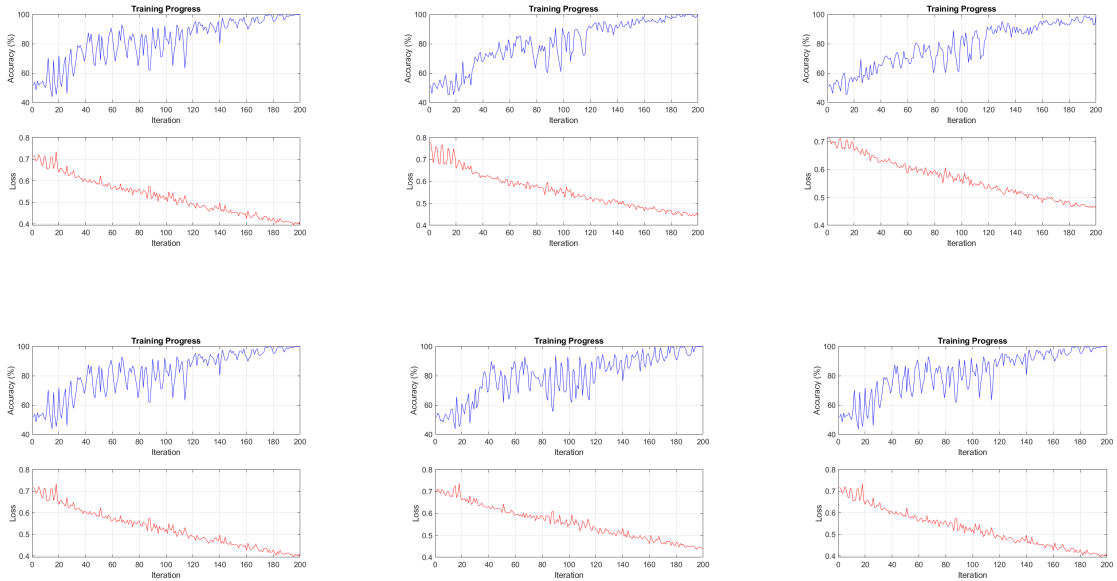


Figure 4.7: Learning Curves for training to distinguish between 0° and 1° , where training sets consisted of combinations of 2 power values among 0.5, 1, 1.5, and 2 mW. 200 training iterations were used to allow convergence to high accuracy and low loss. The top right image shows training on images taken at 0.5 and 1 mW, the top middle at 0.5 mW and 1.5 mW, the top right at 0.5 mW and 2 mW, the bottom left at 1 mW and 1.5 mW, the bottom middle at 1 mW and 2 mW, and the bottom right at 1.5 mW and 2 mW.

mixed power data sets improved the performance accuracies considerably compared to the single power training sets. Promisingly, whenever the algorithm was tested on a power included in the set it was trained on, it performed excellently. However, if the testing level of power was *not* included in the training set, half the time the algorithm performed well and the other half it performed poorly. The main pattern we observed was that if testing power was between the minimum and maximum power levels included in the training set, then the performance was high. We will investigate this pattern further in the upcoming sections.

It seemed that the algorithm needed to be trained on varying levels of power in order to become insensitive to power variation. We now trained the algorithm on sets

		Training Power (mW)					
		0.5+1	0.5+1.5	0.5+2	1+1.5	1+2	1.5+2
Testing Power (mW)	0.5	1	0.996	0.952	0.640	0.572	0.492
	1	1	1	0.996	0.996	0.972	0.5
	1.5	0.948	0.998	0.884	0.988	0.988	0.964
	2	0.5	1	1	0.68	1	1

Figure 4.8: Image Classification Accuracies for 0° vs. 1° , training on images taken at two different powers and testing on a single power. The first row indicates the power combination trained on, while the first column indicates the single power tested on. All possible combinations of two power levels were tried for training.

containing *three* different powers out of the four. This time, the training set for each angle, 0° and 1° , contained 180 images, 60 at each power. We trained on all possible combinations of three of four power levels. We then tested on a single power of the four. Again, the testing set contained 125 images of each angle, as before.

Training on three power levels improved algorithm performance further. The learning curves remained ideal (Fig 4.9), and all accuracies were above 96%, with just two exceptions (See Figure 4.10) As before, the algorithm was consistently accurate in identifying testing images when the image’s power matched one of the power levels in the training set. Again, when the testing level of power was not included in the training set, the algorithm performed well only if the testing level was between the minimum and maximum training levels.

4.2.4 Training on Modulated Power

We concluded that training on datasets with images taken at multiple powers best reduced algorithm sensitivity to power. Since a laser may fluctuate by small

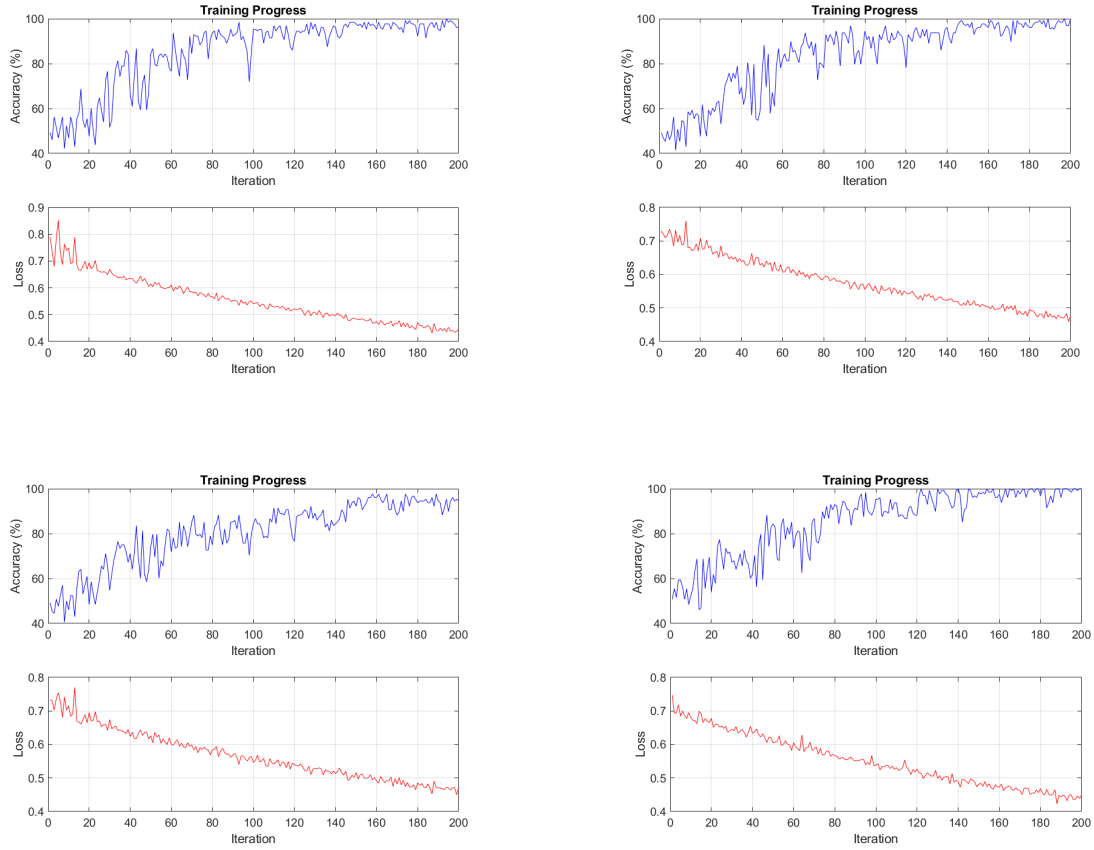


Figure 4.9: Learning Curves for training to distinguish between 0° and 1° , where training sets consisted of combinations of 3 power levels among 0.5, 1, 1.5, and 2 mW. The top left training was conducted on images taken at 0.5 mW, 1 mW, and 1.5 mW, the top right at 0.5 mW, 1 mW, and 2 mW, the bottom left at 0.5 mW, 1.5 mW, and 2 mW, and the bottom right at 1 mW, 1.5 mW, and 2 mW.

amounts within a given range, a more representative training set would consist of images taken at random powers within that range. To achieve this randomness, we used an acousto-optical modulator (AOM) placed before before the waveplate. We sent the first diffraction order down the multimode fiber and set the modulation frequency to 0.3 Hz. We used the variable ND filter after the waveplate to adjust the input power to vary between 0.6 and 0.9 mW (the range of power was limited by the AOM). Using the back and forth waveplate rotation method, we collected 125 images

		Training Power (mW)			
		1+0.5+0.2	1+0.3+0.2	1+0.5+0.3	0.5+0.3+0.2
Testing Power (mW)	0.2	1	1	0.556	0.996
	0.3	0.94	0.952	1	1
	0.5	1	0.916	1	1
	1	1	0.992	1	0.732

Figure 4.10: Image Classification Accuracies for 0° vs. 1° , training on images taken at three different powers and testing on a single power. All possible combinations of two power levels were tried for training.

of 0° polarization and 125 of 1° polarization. We captured 3 images every second so as to allow the images to show the random distribution of powers created by the AOM.

With this training set, we then acquired test images at 0.7, 0.8, and 0.9 mW. We turned the AOM off and used the ND filter to achieve these powers. Each testing set consisted of 125 images taken 0.3 seconds apart. (Note that we took a similar testing set with images taken 5 seconds apart, as before, but we found that the duration of pause did not affect the algorithm performance).

As shown in Figure 4.11, the algorithm easily learned to identify the training images at randomly distributed powers. However, the performance accuracies were much worse than expected, especially given the promising results described in the previous section. Table 4.1 shows that the algorithm was 50% accurate when being tested on 0.8 mW and 0.9 mW, and 81% accurate when tested on 0.7 mW.

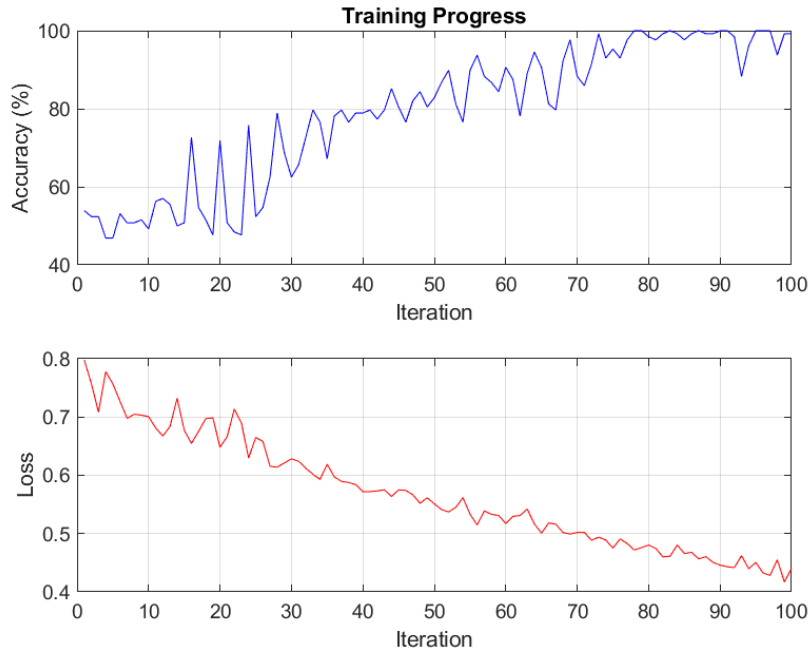


Figure 4.11: Learning curve for a training set consisting of a random distribution of image intensities between 0.7 and 0.9 mW. These training images were obtained using an AOM.

Testing Power (mW)	Accuracy
0.7	0.81
0.8	0.50
0.9	0.50

Table 4.1: Image Classification Accuracies for 0° vs. 1° , training on a random assortment of power levels between 0.7 mW and 0.9 mW, obtained using the AOM. We conducted 3 tests of this training, with images at 0.7 mW, 0.8 mW, and 0.9 mW. The testing images at each power were taken 0.3 seconds apart.

4.3 Training the Algorithm to Distinguish between Images Taken at Different Powers

To investigate more thoroughly the algorithm’s response to different image intensities, we reversed the training scenario with regards to polarization and power. Using the same data as in Section 4.2.3, we fixed the *polarization angle* and trained

the algorithm to distinguish between images at different intensities. Ultimately, we want the algorithm to view images at the same polarization but different intensity as identical because, in that case, the algorithm would be using polarization features, rather than intensity features, to distinguish between images.

We trained the algorithm to distinguish between all possible combinations of 2 power levels, among 0.5, 1, 1.5, and 2 mW. We ran the trainings for 100 epochs, to ensure consistency with previous protocols. Each training set contained 100 images. Notably, though, all the learning curves converged to near 100% before the 10th epoch (Fig 4.12). By contrast, when distinguishing between different polarizations, the algorithm needed 100 or more epochs to reach high accuracy. We also note that the loss in these cases decreased essentially to zero, whereas in the previous scenario, the loss never reached below 0.4. Therefore, we postulate that the algorithm picks up on differences in intensity very easily and that differing intensity features take precedent over the differing spatial/shape features that changing polarization causes. This hypothesis may explain the low accuracies we saw when training the algorithm to distinguish between different polarizations when the power varied: It would mean that the algorithm sees test images of the same polarization, but different intensity, as completely different from the training images.

Indeed, for a fixed polarization of 0° , the algorithm was 100% accurate in testing on images of two different power levels at the same polarization, with one exception. (See Table 4.2). There were 25 images for each power in the testing set. For the same training scenario, we also tested the algorithm on images of 1° polarization, as shown in Table 4.3. Notably, changing the polarization of the testing images did not affect the algorithm performance. These results suggest that, when differing intensities are present among the training and testing images, the algorithm relies heavily on these features, thus preventing it from easily distinguishing different polarizations.

Training Powers (mW)	Testing Accuracy
0.5 vs. 1	0.5
0.5 vs. 1.5	1
0.5 vs. 2	1
1 vs. 1.5	1
1 vs. 2	1
2 vs. 1.5	1

Table 4.2: Image classification accuracies for training the algorithm to distinguish between speckle patterns at different intensities. The first column displays the two power levels the algorithm was trained on, and the second displays the resulting accuracy in testing. All training and testing images were of a 0° polarization. There were 100 images per power level in the training set and 25 per power level in the testing set.

Training Powers (mW)	Testing Accuracy
0.5 vs. 1	0.5
0.5 vs. 1.5	1
0.5 vs. 2	1
1 vs. 1.5	1
1 vs. 2	1
2 vs. 1.5	0.94

Table 4.3: Image classification accuracies for training the algorithm to distinguish between speckle patterns at different intensities. All training images were of a 0° polarization, and all testing images were of 1° polarization.

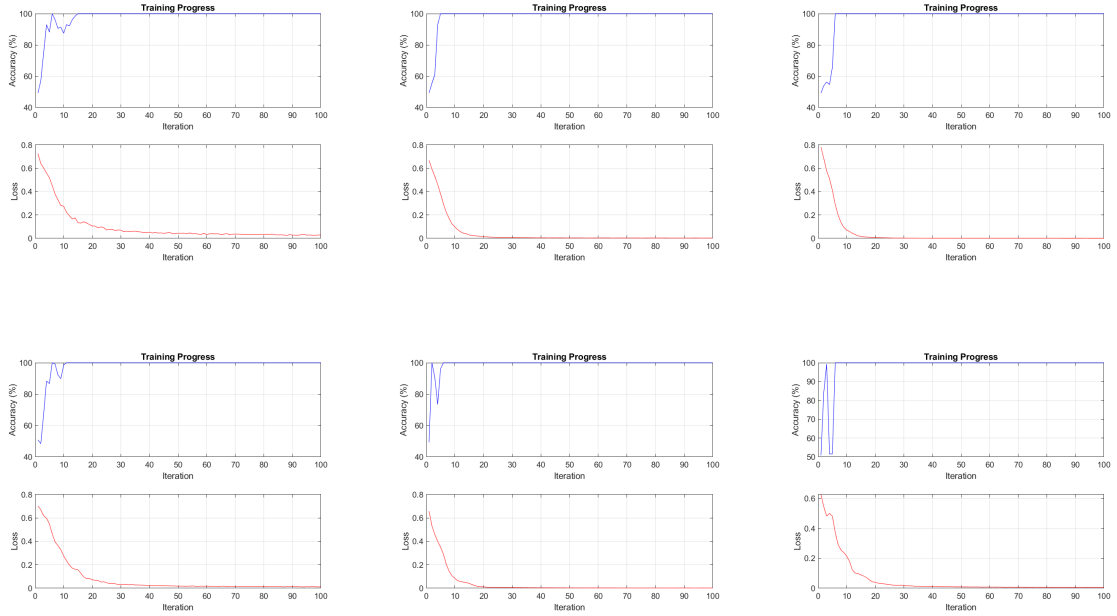


Figure 4.12: Learning Curves for training the algorithm to distinguish between images at two different intensities, where the polarization was fixed to 0° . The top left curve shows training on 0.5 mW vs. 1 mW, the top middle 0.5 mW vs. 1.5 mW, the top right 0.5 mW vs. 2 mW, the bottom left 1 mW vs. 1.5 mW, the bottom right 1 mW vs. 2 mW, and the bottom right 1.5 mW vs. 2 mW.

4.4 Fixing Image Saturation Issues

After the inconclusive investigations into the algorithm's sensitivity to power, outlined in sections 4.2 and 4.3, we examined the image data more closely and determined that many of the images used were saturated. Saturation could obscure important image features. During data processing, we normalize the images so that their pixel values fall between zero and one. This way, images taken at different power levels should not show any visual differences in intensity. The hypothesis was, then, that the algorithm would be insensitive to changes in power. However, the saturation was skewing the normalization process, so that differences in intensity were in fact visible. Figure 4.16 shows two example normalized images, one taken at 0.8 mW and

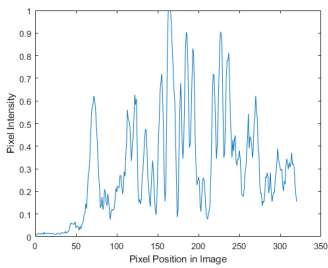
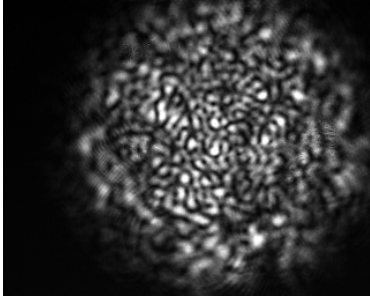


Figure 4.13: 0.8 mW

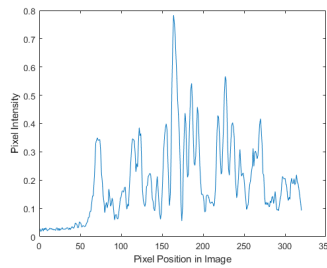
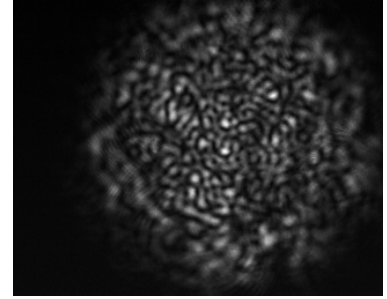


Figure 4.14: 0.2mW

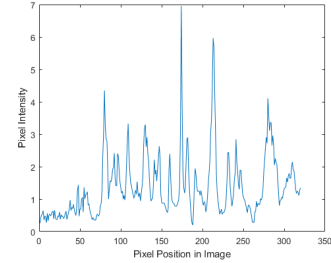


Figure 4.15: 0.8 mW/0.2 mW

Figure 4.16: The top two images are the incorrectly normalized speckle intensity profiles of the beam at 0.8 mW (left) and 0.2 mW (right). The bottom left image shows a cross section of the 0.8 mW image taken at the 125th row of pixels. Plotted are the normalized (between 0 and 1) intensity vs. pixel number, where 1 is the leftmost pixel and 356 the rightmost. The bottom middle image shows the same plot for the 0.2 mW image. The bottom right image is the cross section of the 0.8 mW image divided by the cross section of the 0.2 mW image vs. pixel position.

one at 0.2 mW, which, despite normalization, show visible differences in intensity. Examining a cross section of the 0.8 mW image reveals saturation at the 163rd pixel since the normalized pixel intensity is cut off at 1. Dividing this cross section by the same cross section in the 0.2 mW image shows an average pixel intensity above one, meaning that the normalization was unsuccessful. That is, the images taken at different powers do not have the same pixel intensity values.

The saturation was occurring because the camera exposure time was set to “autoexposure” in the data acquisition script. We therefore changed the exposure time manually to 5000. To check for saturation in the new images, we plotted a cross

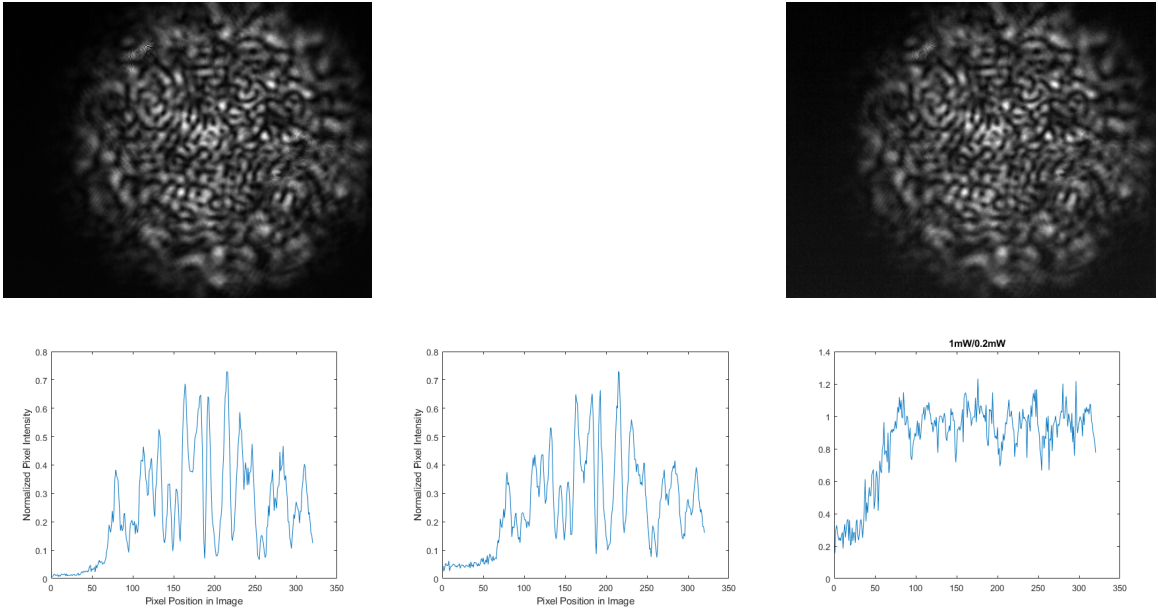


Figure 4.17: The top two images are the correctly normalized speckle intensity profiles of the beam at 1 mW (left) and 0.2 mW (right). The bottom left image shows a cross section of the 1 mW image taken at the 125th row of pixels. Plotted are the normalized (between 0 and 1) intensity vs. pixel number, where 1 is the leftmost pixel and 356 the rightmost. The bottom middle image shows the same plot for the 0.2 mW image. The bottom right image is the cross section of the 1 mW image divided by the cross section of the 0.2 mW image vs. pixel position.

section of an image at a given power divided by the same cross section of an image at a different power. The plot showed values hovering around 1 with fluctuations due to noise, as expected. There is also some noise on the first 50 or so pixels due to the image being dark at those points. Similarly, the resulting png images showed no visual differences in intensity - see Figure 4.17 for an example of images at 1 mW and 0.2 mW, and compare to Figure 4.16.

Now avoiding image saturation, we took a new set of images at 0° and 1° , at power levels of 1, 0.5, 0.3, and 0.2 mW. We performed the three main trainings that we had tried in the past, which will be outlined in the following sections.

		Training Power (mW)			
		0.2	0.3	0.5	1
Testing Power (mW)	0.2	1	0.50	0.55	0.53
	0.3	0.50	1	0.50	0.55
	0.5	0.50	0.59	1	0.66
	1	0.50	0.51	0.54	1

Figure 4.18: Image Classification Accuracies for 0° vs. 1° , training and testing on all possible combinations of 0.2 mW, 0.3 mW, 0.5 mW, and 1 mW.

4.4.1 Distinguishing between 0° and 1° , Train on One Power Level, Test on Another

We first followed the protocol given in section 4.2.2, where we trained the algorithm to distinguish between 0° and 1° , training on one power level and testing on a different power level. The powers used were 0.2 mW, 0.3 mW, 0.5 mW, and 1 mW. Despite our avoidance of saturation and successful normalization, we saw similar behavior as in the past. As shown in Figure 4.18, the algorithm was about 50% accurate for all training/testing combinations. The training plots also look ideal and similar to past plots in terms of convergence time and loss decrease. We were surprised to see no obvious improvement over trainings using saturated images, as shown in Figure 4.19.

4.4.2 Distinguishing between 0° and 1° , Train on Mixed Power Levels

We next followed the protocol given in section 4.2.3, where we trained the algorithm to distinguish between 0° and 1° , this time training on a set of images containing 3 different power levels. As before, the algorithm was highly accurate when tested on a power level included in the training set. However, as shown in Figure 4.20,

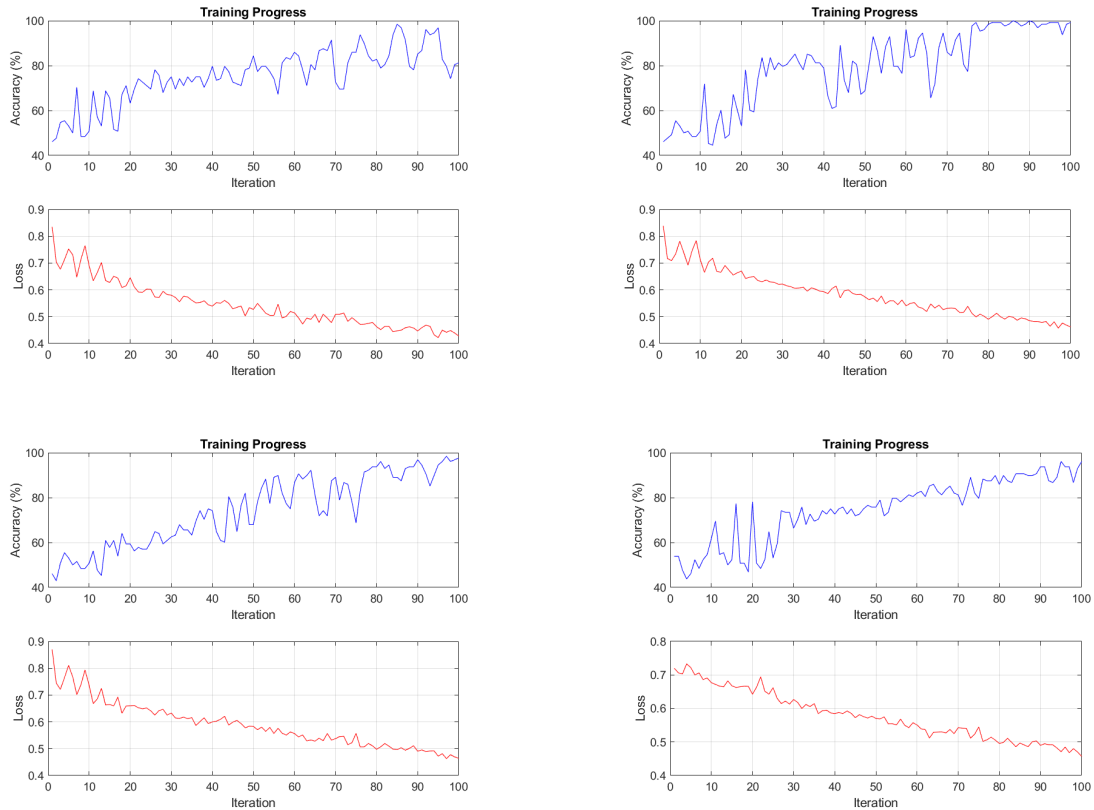


Figure 4.19: Learning Curves when training to distinguish between 0° and 1° polarization, where each plot shows trainings conducted on unsaturated images taken at four different fiber input powers, 0.2, 0.3, 0.5, and 1 mW. The top left image shows the training conducted on 0.2 mW images, the top right 0.3 mW, the bottom left 0.5 mW, and the bottom right 1 mW.

when testing on a power level not included, there were two cases where the algorithm was over 90% accurate. In one other case, though, it was 78% accurate, and in the other, it was 56% accurate. These results follow the same pattern we noticed before where the testing images must be taken at levels of power between the minimum and maximum training powers in order to achieve high performance. It seems in general that the algorithm can extrapolate *between* two training endpoints, but not outside of them.

The corresponding training plots are shown in Figure 4.21.

		Training Power (mW)			
		1+0.5+0.2	1+0.3+0.2	1+0.5+0.3	0.5+0.3+0.2
Testing Power (mW)	0.2	1	1	0.556	0.996
	0.3	0.94	0.952	1	1
	0.5	1	0.916	1	1
	1	1	0.992	1	0.732

Figure 4.20: Image Classification Accuracies for 0° vs. 1° , training on images taken at three different powers among 0.2 mW, 0.3 mW, 0.5 mW, and 1 mW, and testing on one of those powers. All possible combinations of two power levels were tried for training.

4.4.3 Train the Algorithm to Distinguish between Two Intensities for a Fixed Polarization

Finally, we followed the protocol outlined in section 4.3. We fixed the polarization to 0° and trained the algorithm to distinguish between two different intensities. With one exception, the algorithm was 100% accurate in all cases (Table 4.4), just as before. Furthermore, the training plots converged to 100% accuracy by the tenth epoch, and the loss decreased to almost zero (Figure 4.22). The results of this third test especially indicate that the algorithm is picking up on differences in intensity, despite our efforts to eliminate these differences through normalization. Initially we thought that this behavior could be explained by the algorithm picking up on the noise pixels, from positions 1-50 shown in Figure 4.17. However, as will be discussed in Section 6.3, we found that blocking out regions of the images outside the actual beam did not affect performance accuracy. Another possibility is that this noise acts as a background that remains present in the images even after normalization. In this case, we would need to calculate this background and subtract it out.

At this time, therefore, we conclude the investigations into intensity by noting that

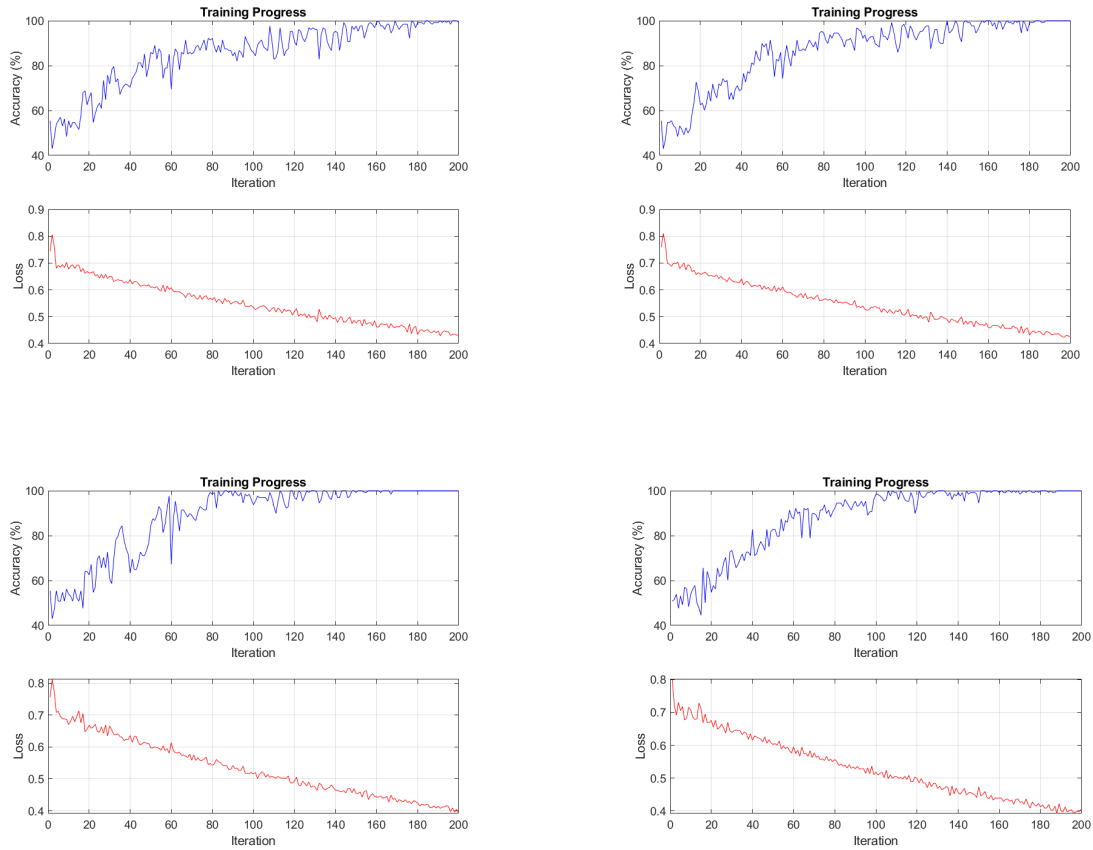


Figure 4.21: Learning Curves for training to distinguish between 0° and 1° , where training sets consisted of combinations of 3 power levels among 0.2, 0.3, 0.5, and 1 mW. The top left training was conducted on images taken at 1 mW, 0.3 mW, and 0.2 mW, the top right at 1 mW, 0.5 mW, and 0.2 mW, the bottom left at 1 mW, 0.5 mW, and 0.3 mW, and the bottom right at 0.5 mW, 0.3 mW, and 0.2 mW.

algorithm performance depends on image intensity more strongly than expected. We are content, then, to keep the laser intensity constant in further experiments, with the acknowledgement that the matter of intensity merits more investigation when time allows. The algorithm's ability to extrapolate between levels of training powers is encouraging, but we would eventually like to investigate why it cannot extrapolate outside the training levels. Practically, though, this drawback could be easily solved by knowing the endpoints of the device laser's intensity fluctuations.

Training Powers (mW)	Testing Accuracy
0.2 vs. 0.3	1
0.2 vs. 0.5	1
0.2 vs. 1	1
0.3 vs. 0.5	0.78
0.3 vs. 1	1
0.5 vs. 1	1

Table 4.4: Image classification accuracies for training the algorithm to distinguish between speckle patterns at different intensities. The first column displays the two power levels the algorithm was trained on, and the second displays the resulting accuracy in testing. All training and testing images were of a 0° polarization. There were 100 images per power level in the training set and 25 per power level in the testing set.

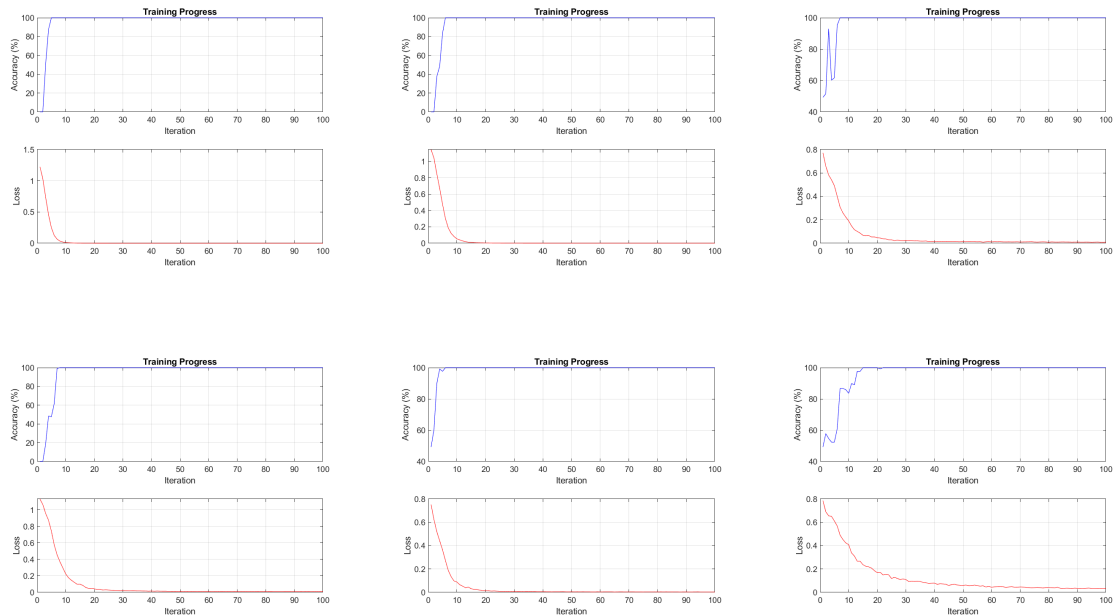


Figure 4.22: Learning Curves for training the algorithm to distinguish between images at two different intensities, where the polarization was fixed to 0° . The top left curve shows training on 0.2 mW vs. 1 mW, the top middle 0.3 mW vs. 1 mW, the top right 0.3 mW vs. 0.2 mW, the bottom left 0.5 mW vs. 1 mW, the bottom right 0.5 mW vs. 0.3 mW, and the bottom right 0.5 mW vs. 0.3 mW.

Chapter 5

Improving Angular Sensitivity with the Waveplate

Besides decreasing the algorithm’s sensitivity to changing environmental factors, the goal of this work is to enable the algorithm to detect the small changes in polarization that correspond to those induced by a magnetic field. By Spring 2021, the algorithm developed was able to distinguish between changes in polarization as small as 1° .

After fixing the image saturation issues mentioned in Section 4.4.3, we reassessed the algorithm’s angular sensitivity. Using the “back and forth waveplate rotation” method, we took images of multimode fiber output patterns corresponding to polarizations of 1° , $\frac{1}{2}^\circ$, $\frac{1}{4}^\circ$, $\frac{1}{8}^\circ$, and 0° . Each training set consisted of 100 images, and each testing set consisted of 25. As shown in Table 5.1, with the unsaturated images, the algorithm was able to distinguish between a change in polarization as small as $\frac{1}{4}^\circ$. This test was 90% accurate, while a change in polarization of $\frac{1}{2}^\circ$ was 86% accurate, and a change of 1° was 100% accurate, as before. A change of $\frac{1}{8}^\circ$ was 74% accurate, but this result is unreliable as the change in polarization is so small. The waveplate used had a precision down to $\frac{1}{25}^\circ$, so we would estimate an instrumental uncertainty of $\frac{1}{50}^\circ$. However, a $\frac{1}{8}^\circ$ rotation corresponded to $\frac{3}{25}^\circ + \frac{1}{8 \cdot 25}^\circ$ rotation of the waveplate knob. This extra $\frac{1}{8 \cdot 25}$ was beyond the waveplate’s precision.

Training Polarizations	Testing Accuracy
0° vs. 1°	1
0° vs. $\frac{1}{2}^\circ$	0.86
0° vs. $\frac{1}{4}^\circ$	0.90
0° vs. $\frac{1}{8}^\circ$	0.74

Table 5.1: Image Classification Accuracies for changes in polarization of 1° , $\frac{1}{2}^\circ$, $\frac{1}{4}^\circ$, and $\frac{1}{8}^\circ$ when using unsaturated images.

As shown in Figure 5.1, the learning curves were ideal in all cases except $\frac{1}{8}^\circ$, indicating that the algorithm can indeed learn to distinguish changes in polarization as small as $\frac{1}{4}^\circ$.

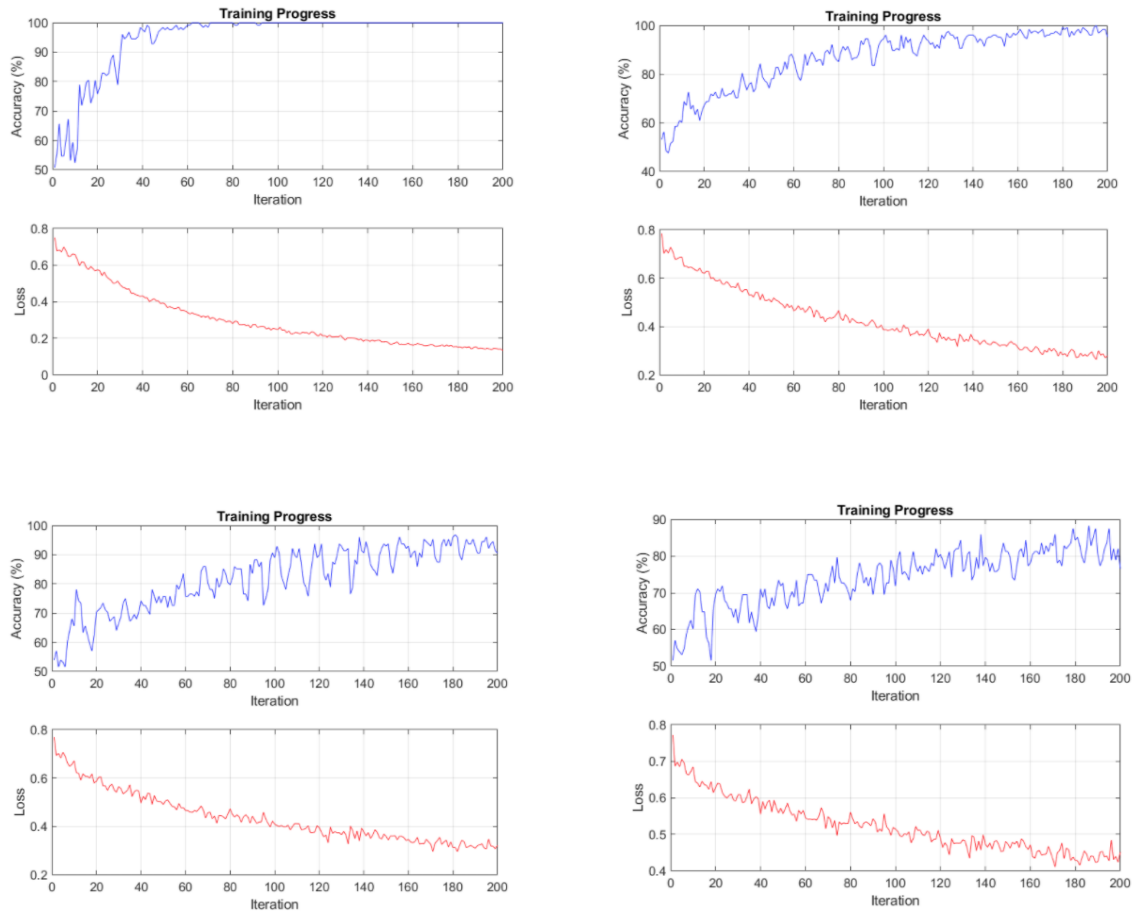


Figure 5.1: Learning Curves for improved angular sensitivity using unsaturated images. The top left plot shows training to distinguish between 0° and 1° , the top right shows 0° and $\frac{1}{2}^\circ$, the bottom left shows 0° and $\frac{1}{4}^\circ$, and the bottom right shows 0° and $\frac{1}{8}^\circ$.

Chapter 6

Implementing a Faraday Magnetometer

6.1 New Setup Replacing the Waveplate with a Solenoid

With the improved angular sensitivity, we transitioned to imposing a real magnetic field onto our setup instead of using a waveplate to rotate the laser polarization. Thus, we moved to the final experimental setup, originally discussed in Chapter 2 and shown in Figure 2.1.

6.2 Faraday Magnetometer Benchmark Angular Sensitivity Tests

After calibrating the TeachSpin apparatus to determine what current values corresponded to what angle of polarization rotation (see Figure 2.1), we performed similar tests to those done with the waveplate to ensure that we obtained comparable algorithm performance. We took image data at a 0° , 0.5° , 1° , 1.5° , and 2° . As before, the training sets for each angle consisted of 100 images and the testing sets consisted of 25. We took all the images for each angle in one straight set since there was no equivalent to the “back and forth waveplate rotation” method with the solenoid. A

Training Polarizations	Testing Accuracy
0 ° vs. 0.5 °	0.92
0 ° vs. 1 °	1
0 ° vs. 1.5 °	1
0 ° vs. 2 °	1
0.5 ° vs. 1 °	0.8
0.5 ° vs. 1.5 °	1
0.5 ° vs. 2 °	1
1 ° vs. 1.5 °	1
1 ° vs. 2 °	1
1.5 ° vs. 2 °	1

Table 6.1: Image Classification Accuracies for the benchmark Faraday magnetometer tests.

pause of 3 seconds separated each image capture. The trainings were conducted for 100 epochs.

All possible combinations of angles were tested, and the training plots were ideal in all cases - four examples are shown in Figure 6.1. As shown in Table 6.1, the performance accuracy was 100% in all cases except two. From here, we can conclude that so far the algorithm can generally distinguish between changes in polarization as small as half a degree.

6.3 Imposing Masks on the Image Data

Our earlier tests of image intensity brought up concerns that perhaps the algorithm was picking up on differences in the images that lay outside the actual beam, in the black background. To eliminate this possible effect, we henceforth imposed circular masks on our images to “block out” non-beam regions.

To create the masked images, we passed the image data through a MATLAB code that set all the pixels outside a given radius to the same value. Thus, all the images would be uniform outside the given circle. The circle radius and center location could

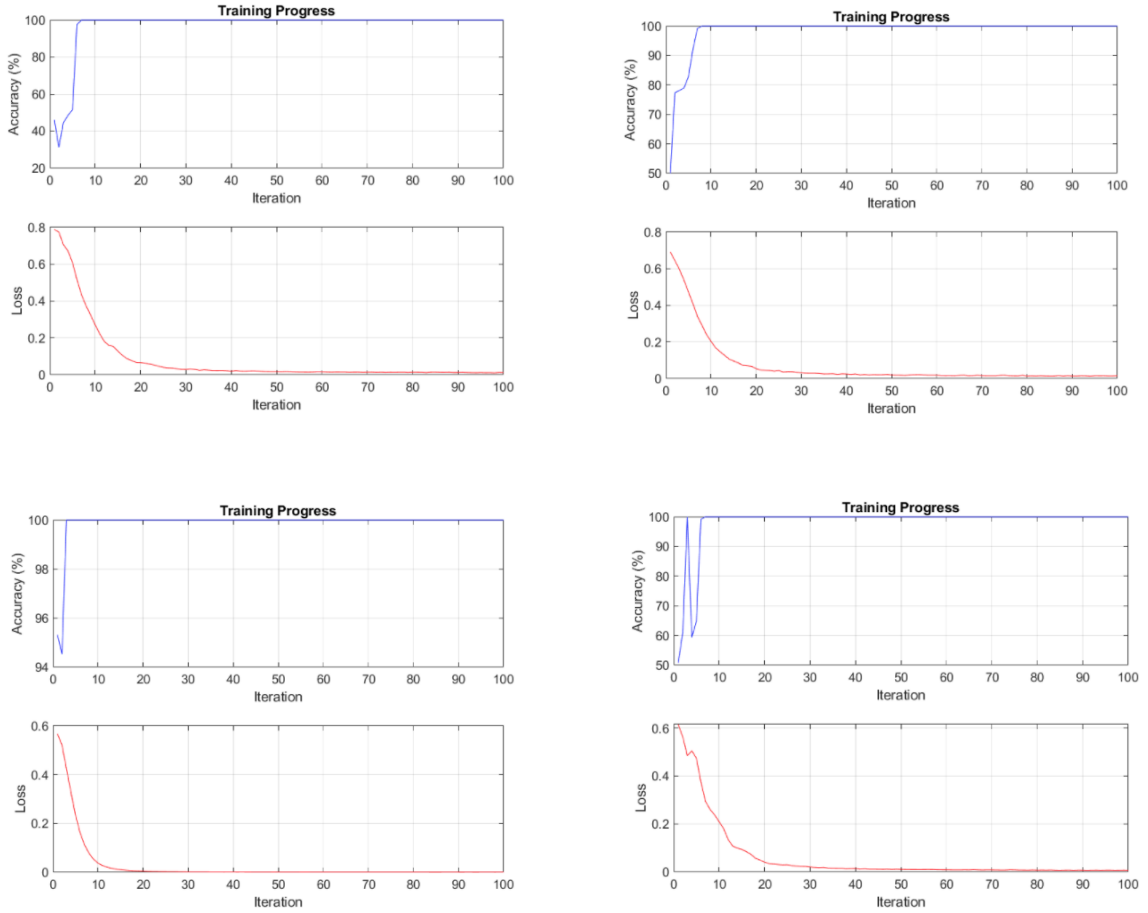


Figure 6.1: Learning Curves for determining angular sensitivity using the Faraday magnetometer. The top left plot shows training to distinguish between 1° and 0.5° , the top right shows 1° and 1.5° , the bottom left shows 1° and 2° , and the bottom right shows 0° and 0.5° .

be adjusted to match the data.

The data from Section 6.2 showed speckle patterns of radius about 125 pixels centered about at coordinates (175, 150). We imposed masks with these specifications on this data and repeated the trainings. For the training categories that were 100% accurate without the mask, imposing the mask resulted in no change. Interestingly, in the 0° vs. 0.5° and the 0.5° vs. 1° cases, which had below 100% accuracy without the mask, imposing a mask did not significantly improve performance. The former

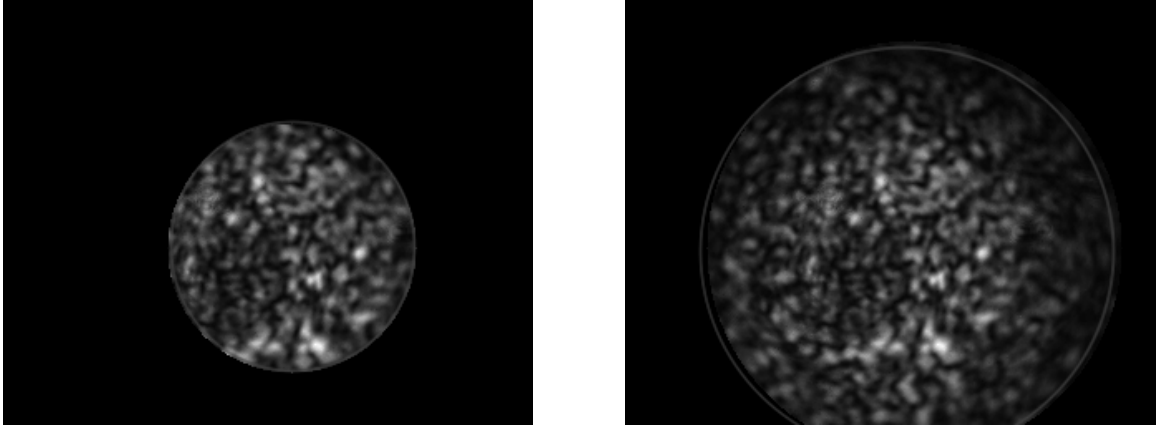


Figure 6.2: Sample Masked Images, where the mask has been outlined for viewing ease. Left: sample data with a mask of 75 pixel radius. Right: sample data with mask of 125 pixel radius.

case became 94% accurate and the latter 78% accurate, from 92% and 80% accurate, respectively. Both of these small changes are likely attributable to the uncertainty in algorithm performance. Thus, we concluded that the algorithm was *not* using non-beam regions to distinguish between different polarizations.

6.4 Exploring Performance Dependence on Mask Size and Location

We hypothesized that reducing the size of the masks could improve accuracy by allowing the algorithm to focus on a smaller area of the complicated speckle patterns, while also decreasing computational power.

We first investigated how far the mask radius could be decreased before performance suffered. We used the data described in the previous two sections. We conducted this study on images of 0° and 1° because the algorithm was 100% accurate at distinguishing these two angles without a mask and with a mask the size of the beam. We tested mask radii of 125, 100, 75, 50, 25, 12, 6, and 3 pixels. The masks were centered on the beam at an x value of 175 and y value of 150. The trainings

lasted 100 epochs. Surprisingly, we found that the algorithm was 100% accurate in distinguishing these two angles at *all* the radii listed tested. This indicates that the algorithm can distinguish between detailed image features at the pixel level.

With these promising results, we tested whether decreasing the mask size would improve the performance when distinguishing between the pairs of angles that yielded less than 100% accuracies without a mask and with a mask of 125 pixels, 0° vs. 0.5° and 0.5° vs. 1° .

For a 50 pixel radius mask centered on the beam, the algorithm was 92% accurate at distinguishing between 0° and 0.5° , and for a 25 pixel radius mask centered on the beam, it was 88% accurate for those same angles. When distinguishing between 0.5° and 1° with a 50 pixel radius mask, centered, it was 74% accurate. Without a mask, the algorithm was 92% and 80% accurate at distinguishing these two pairs of angles. Thus, decreasing the mask size did not improve performance that was sub par without a mask. We may try even smaller mask radii in case, but we do not expect improvement.

We also experimented with changing the location of the center of the mask in our images. As before, we started investigations with an angle pair that yielded high accuracies without a mask. We chose 1° vs. 1.5° . Using a radius of 20 pixels, so as to avoid hitting the edge of the beam, we tested center locations at (x, y) coordinates of (150, 110), (220, 200), and (210, 100). All cases were 100% accurate. Because of such effective training, we were able to reduce the number of training epochs to just 40.

From all these tests involving image masks, we concluded that imposing a mask to “block” out non-beam regions of the image is a best practice that we will incorporate into our data processing procedure from now on. Changing the size or location of the mask does not seem to have any benefit.

Training Polarizations	Number of Epochs	Testing Accuracy (%)
0° vs. $\frac{1}{2}^\circ$	100	1
0° vs. $\frac{1}{4}^\circ$	100	66
0° vs. $\frac{1}{4}^\circ$	200	74
0° vs. $\frac{1}{8}^\circ$	100	84
0° vs. $\frac{1}{8}^\circ$	200	72

Table 6.2: Image Classification Accuracies for Fractional Angles Using the Faraday Magnetometer.

6.5 Testing Smaller Angles

We now tested the angular sensitivity of our Faraday magnetometer. We took image data at 0° , $\frac{1}{8}^\circ$, $\frac{1}{4}^\circ$, and $\frac{1}{2}^\circ$, and tested all possible combinations. As shown in Table 6.2, the algorithm was 100% accurate in distinguishing a $\frac{1}{2}^\circ$ change in polarization, as before. However, for a $\frac{1}{4}^\circ$ change, it was only 66% accurate with 100 epochs, and 74% accurate with 200 epochs. Strangely, with 100 epochs, it was 84% accurate in distinguishing a $\frac{1}{8}^\circ$ change, but this accuracy *decreased* to 72% when doubling the number of epochs. These results contradict our two basic expectations that a smaller change in polarization will be less accurate and a larger number of epochs will improve performance. We also observe from the learning curves in Figure 6.3 that despite the fact the accuracy increases to 100% in all cases, the loss does not reach or just barely reaches zero. This suboptimal loss could indicate, that though the algorithm has high confidence, it is not sufficiently robust in testing (see Section 2.2.1). The next section will discuss efforts to understand and improve these inconsistencies.

6.6 Improving Angular Sensitivity of the Faraday Rotation Magnetometer

In this last section, we will discuss methods we tried to improve the angular sensitivity of the Faraday magnetometer and to resolve the questions that concluded

the last section. To address these inconsistent results, we first tried increasing the size of the testing and training sets to 500 images in each. We also ensured that images were placed into the training and testing sets *randomly*. With these changes, we observed drastic improvements.

Note that we attribute the vast majority of this improvement to the larger data set size rather than the random image distribution into the testing and training sets. With previous, smaller datasets, we tried swapping the images included in the training and testing sets and found no change performance, so we concluded that randomly distributing the images would not greatly affect performance. We merely implemented the random distribution as a best practice.

We achieved to 100% accuracy in detecting a change in polarization as small as $0.6 \times 10^{-4} \text{ }^\circ$, as shown in Table 6.3. Note that in this Table, magnetic field values were calculated using equation 1.1 after ϕ was calculated from the current vs. polarization calibration shown in Figure 2.2. We used a value of $C_V = 23 \text{ rad}/(\text{T}\cdot\text{m})$, which is the value for 650 nm light. Our light was of a higher wavelength (780 nm), so our reported values of B might be slightly high. We were unable to find a reference that reported a value of C_V for SF-59 glass at 780nm.

The learning curves shown in Figure 6.4 were ideal. This change in polarization corresponds to a magnetic field of about $0.5 \mu\text{T}$. When measuring an angle this small, we ran up against the precision of our mA power supply. So far, there is no indication that the algorithm will struggle to distinguish smaller changes in polarization, which can be obtained with an extra sensitive power supply.

We thus conclude that a training set of 100 images was too small for this type of training when the change in polarization was so small.

We are currently continuing efforts to push the algorithm's sensitivity lower. So far, the only obstacle is the precision of our power supply, which is sensitive up to

Training Polarizations ($\times 10^{-4}$ degrees)	Magnetic Field (μT)	Testing Accuracy (%)
0 vs. 156.3	119	1
0 vs. 78.1	59	1
0 vs. 39.0	30	1
0 vs. 19.5	15	0.998
0 vs. 6.8	5	1
0 vs. 5.0	4	0.999
0 vs. 2.4	2	1
0 vs. 0.6	0.5	1

Table 6.3: Image Classification Accuracies for Smaller Fractional Angles and Corresponding Magnetic Fields.

0.1 mA. There are no indications yet that the algorithm will fail at smaller magnetic fields.

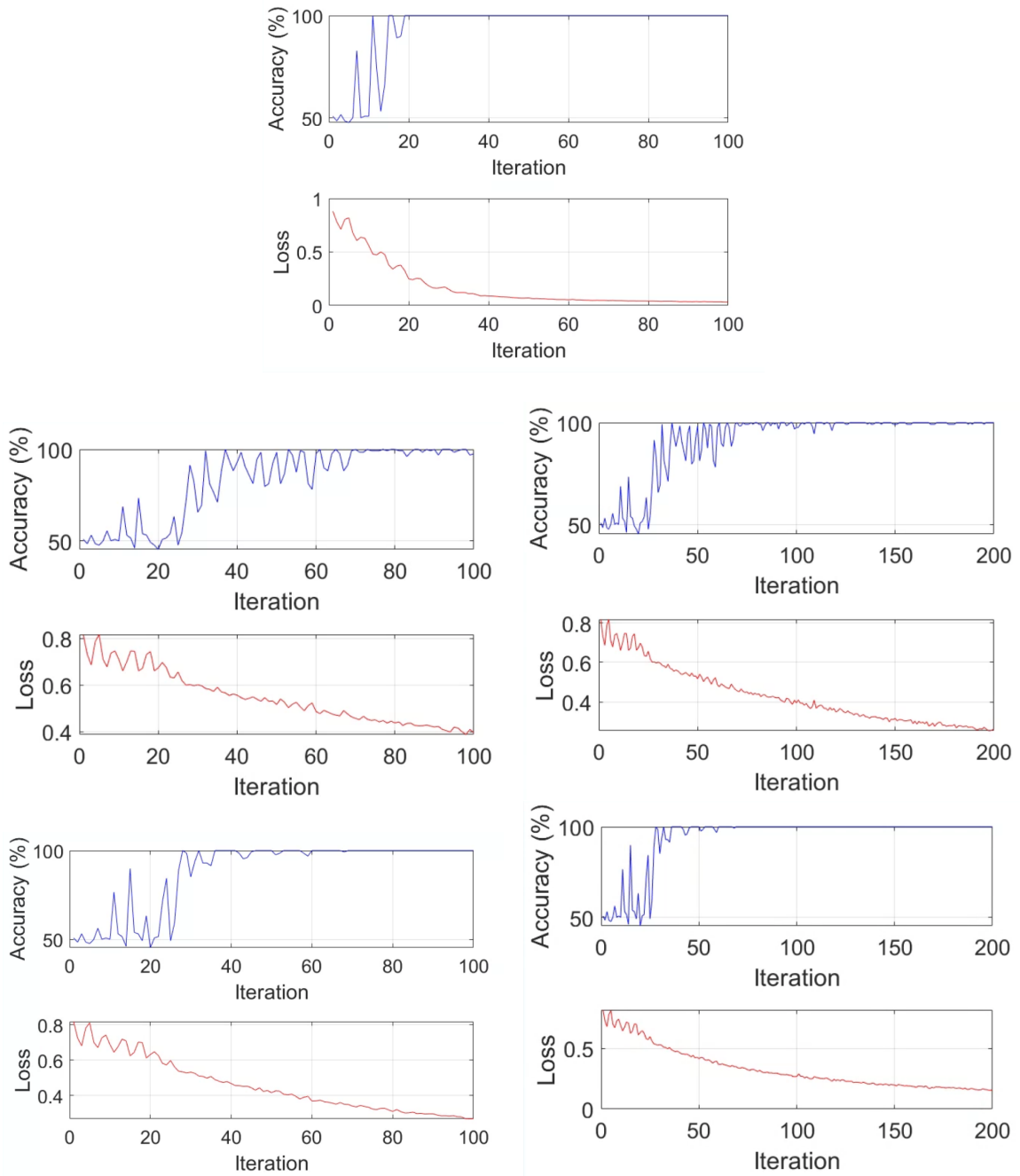


Figure 6.3: Learning Curves for determining angular sensitivity using the Faraday magnetometer. The topmost plot shows training to distinguish between 0° and $\frac{1}{2}^\circ$, the second row shows 0° and $\frac{1}{4}^\circ$ for 100 and 200 iterations (left and right, respectively), and the bottom row shows 0° and $\frac{1}{8}^\circ$ for 100 and 200 iterations (left and right, respectively).

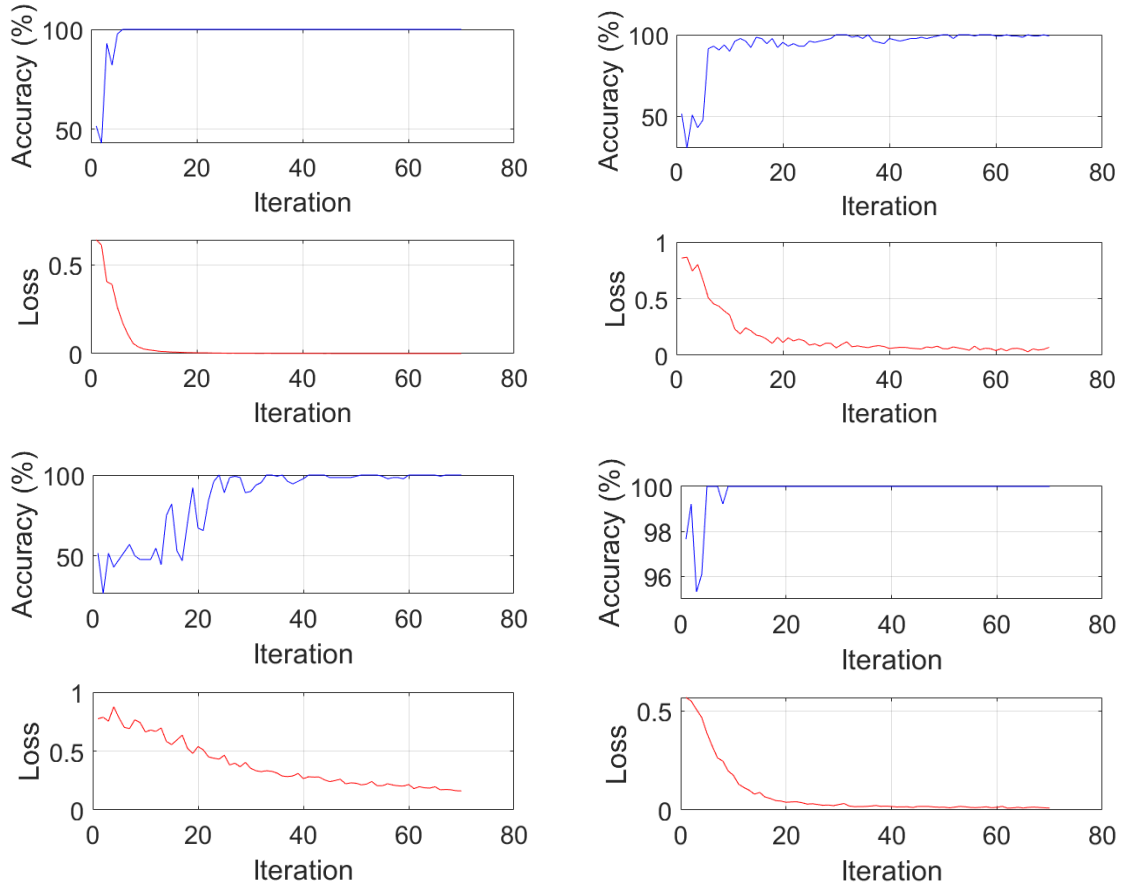


Figure 6.4: Learning curves for determining further angular sensitivity using the Faraday magnetometer. The top left plot shows training to distinguish between 0° and $156.3 \times 10^{-4} \text{ }^\circ$, the top right shows 0° and $19.5 \times 10^{-4} \text{ }^\circ$, the bottom left shows 0° and $5.0 \times 10^{-4} \text{ }^\circ$, and the bottom right shows 0° and $0.6 \times 10^{-4} \text{ }^\circ$.

Chapter 7

Conclusion

We constructed an optical fiber-linked magnetometer for magnetic field measurement. Using this unique approach, we detected a change in magnetically-induced laser polarization as small as 0.6×10^{-4} degrees, or 1.0×10^{-5} rad, with 100% accuracy, which corresponds to a magnetic field of $0.5 \mu\text{T}$.

For this work, we constructed a convolutional neural network capable of distinguishing the small changes in polarization that occur in the presence of a magnetic field. We tested the algorithm first using a waveplate to obtain a preliminary angular sensitivity of $\frac{1}{4}^\circ$. We tested the algorithm's response to changing image intensity. We concluded that the algorithm is very sensitive to changes in image intensity even when the images are all unsaturated and then normalized identically. We implemented the procedure of imposing masks on the images, which confirmed that the algorithm was not training on non-beam regions of the images.

When replacing the waveplate with TeachSpin's Faraday rotation apparatus, we found that we could improve the device's angular sensitivity beyond $\frac{1}{8}^\circ$ by training on a data set five times the size of the original sets we had used.

To summarize, in addition to optimizing the image classification algorithm itself, we also developed a variety of data collection and processing procedures that lead to optimal performance. They are:

- Collecting all the image data for a single training episode as close together as possible in order to avoid large scale drifts in the speckle patterns. We acknowledge that for a practical, commercial device, this protocol would be difficult. We will work on ways to mitigate the effects of pattern changes on performance.
- Rescaling the images from 1280×1024 pixels² to 320×256 pixels². to reduce computational power and time.
- Normalize images so their pixel values all fall between 0 and 1.
- Ensure images are unsaturated to avoid skewing the normalization.
- Impose on all images a mask the size of the beam, centered on the beam, as a best practice.
- Keep laser intensity constant until further investigations into the effect of intensity on performance are done. Currently, it seems that the algorithm can identify images at new levels of power if that level is within endpoints set by the training data.
- Allow the performance loss to decrease to zero even if the accuracy has plateaued at 100% to ensure algorithm robustness.
- Use large training and testing sets - we used 500 images in each.

In the future, to improve the magnetic field sensitivity, we may replace the crystal with some much more magneto-sensitive material, such as a Rb atomic vapor, which has a Verdet constant, C_V , of 1.4×10^3 [15], two orders of magnitude higher than that of the glass rod used thus far. Another option is to use a crystal vacancy, such

as an Nitrogen-Vacancy (NV) center, as the sensor. The NV center would allow for further compactness, with the possibility of miniaturization in a chip-scale device.

We expect our magnetometer to perform well in applications like defects detection of metal components, say, in aerospace vehicles or infrastructure. The device's compact, flexible configuration makes it attractive for settings that require small scale and mobility. The optical fiber makes the device useful also in settings that are not electronics friendly because it allows for separation between the actual sensor and the detection mechanism. using a crystal, we could place the sensor in small, hard-to-reach places where a magnetic field needs to be measured, while the detection mechanism is far away in a more accessible location. Thus, the user may detect very small changes in high magnetic fields. Finally, the AI component allows for high speed analysis in time-sensitive situations.

Bibliography

- [1] Kitching, J., Knappe, S., and Donley, E.A. “Atomic Sensors - A Review”. In: *IEEE Sensors Journal* 11 (2011), pp. 1749-1758. DOI: 10.1109/JSEN.2011.2157679.
- [2] <https://www.teachspin.com/faraday-rotation>
- [3] Mikhailov, E. E., Novikova, I., Havey, M.D., Narducci, F. A. “Magnetic Field Imaging with Atomic Rb Vapor”. In: *Optics Letters* 34 (2009), pp. 3529-3531. DOI: 10.1364/OL.34.003529.
- [4] Kitching, J. “Chip-Scale Atomic Devices”. In: *Applied Physics Reviews* 5 (2018), pp. 031302-1 - 031302-38. DOI:10.1063/1.5026238.
- [5] Maze,J., Stanwix, P., Hodges, J. et al., “Nanoscale Magnetic Sensing with an Individual Electronic Spin in Diamond”. In: *Nature* 455,(2008), pp. 644-647. DOI: 10.1038/nature07279.
- [6] Kuwahata, A. “Magnetometer with nitrogen-vacancy center in a bulk diamond for detecting magnetic nanoparticles in biomedical applications”. In: *Scientific Reports* 10 (2020). DOI: 10.1038/s41598-020-59064-6.
- [7] Borhani, N., Kakkava, E., Moser, C., and Psaltis, D. “Learning to See through Multimode Fibers”. In: *Optica* 5 (2018), pp. 960-996. DOI: 10.1364/OPTICA.5.000960.

- [8] Wang, P. and Di, J. “Deep Learning-Based Object Classification through Multimode Fiber via a CNN-Architecture SpeckleNet”. In: *Applied Optics* 57 (2018), pp. 8258-8263. DOI: 10.1364/AO.57.008258.
- [9] Albawi, S., Mohammed, T. A., Al-Zawi, S. “Understanding of a Convolutional Neural Network”. In: *IEEE International Conference on Engineering and Technology (ICET)* 2017, pp. 1-6. DOI: 10.1109/ICEngTechnol.2017.8308186.
- [10] Deshpande, A. “A Beginners Guide to Understanding Convolutional Neural Networks.” (2016) <https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/>.
- [11] Saha, S. “A Comprehensive Guide to Convolutional Neural Networks”. *Towards Data Science* (2018). <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>.
- [12] Huotari, Jussi. “Why Loss and Accuracy Metrics Conflict?”. (2018). <http://www.jussihuotari.com/2018/01/17/why-loss-and-accuracy-metrics-conflict/>.
- [13] Brownlee, J. “How to Configure the Learning Rate when Training Deep Learning Neural Networks.” *Machine Learning Mastery*. (2019). <https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/>.
- [14] Srinivasan, A. “Stochastic Gradient Descent — Clearly Explained!!!”. (2019). <https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31>.

- [15] Weller, L., et al. “An optical isolator using an atomic vapor in the hyperfine Paschen-Back regime.” In: *Optics Letters* 37 (2012), pp.3405-3407
<https://doi.org/10.1364/OL.37.003405>.

Appendix A

Image Processing Code and Image Classification Algorithm

A.1 Image Acquisition and Rescaling Code Sample

The following is the MATLAB code which acquired the speckle pattern images, scaled them down to 320x256 using interpolation, and normalized their pixel values between 0 and 1.

```
/* **** */
/* Sofia Brown */

function [data] = acquire_images_Matrix2RescaledFile(vid_obj,exposure_time,frames_per_trigger,
frame_num)
%% setting defaults
if nargin < 1
    vid_obj = ' mWspinnakerimaq';
end

if nargin < 3
    frames_per_trigger = 1;
end

%% Acquiring Data from Blackfly
% set file name params
date = '20211117'; %yyymmdd %
%type_ind = '10';% '00','10','-10' (vortex setting)
%pos_ind = '_t' ;% '_L', '_R', '_C' (vortex horizontal position) %
pol_angle = '1'; %Polarization angle in degrees
frame_num = 25 ;%Input number of images you want to be taken per cycle
data_set = '2' ; %The number of the data set being taken
power = '_Half mW_';
% Run aquisition
```

```

vid = videoinput(vid_obj, 1, 'Mono10Packed');

%Set camera settings
src = getselectedsource(vid);
src.ExposureAuto = 'Off';
src.ExposureTime=5000;
vid.FramesPerTrigger = frames_per_trigger;

for i = 1:frame_num
start(vid);
data(:,:,i) = getdata(vid,1);
stop(vid);
pause(5) %Pause number of seconds between images in a cycle
msg=horzcat('frame ', num2str(i));
disp(msg)
end

disp('saving data')
%file_name = horzcat(date,'_LG_',type_ind,pos_ind);
filename = horzcat(date, '_', pol_angle, 'deg', power, data_set);
save(horzcat('Z:\Sofia\Fall2021\RangeofPowers_Unsaturated_11_17_21\UnnormalizedData\'',filename),
'data');

%
filepath = 'Z:\Sofia\Fall2021\RangeofPowers_Unsaturated_11_17_21\UnnormalizedData\'';
savepath = 'Z:\Sofia\Fall2021\RangeofPowers_Unsaturated_11_17_21\'';
fileext = '.png';
Nsmallx = 320;
Nspecially = 256;
start_num = 1;
finish_num = 25;

Matrix2Rescaled2file(filepath,filename, savepath, fileext, Nsmallx, Nspecially, start_num,
finish_num);

end

end

%% Matrix2Rescaled file cuts a 3D matrix into i 2D matrices, rescales them to Nsmallx x
Nspecially pixels (which can increase or decrease the size of the original matrix through
interpolation) Then, the rescaled images are saved in the desired format

%% Break 3D matrix
%filepath is a string that is the name of the file and its directory
%savepath is the path where the new files are saved
function [] = Matrix2Rescaled2file(filepath,inputfile, savepath, fileext, Nsmallx, Nspecially,
start_num, finish_num)
datin=load(horzcat(filepath, inputfile));
%Change datain._____ to match variable name
dat = datin.data;
[~, ~, zdim] = size(dat);

```

```

for i = start_num : finish_num
    slice = dat(:,:,i);
    slice = double(slice);

%% Rescaling

[Nsupery, Nsuperx] = size(slice);

xcsuper = 1: Nsuperx;
xcsuper = xcsuper - 0.5;
ycsuper = 1:Nsupery;
ycsuper = ycsuper-0.5;

% sx, sy: axis scale factors
sx = Nsuperx/Nsmallx;
sy = Nsupery/Nsmally;

xcsmall=1:Nsmallx;
xcsmall = (xcsmall-0.5)*sx;
ycsmall = 1:Nsmally;
ycsmall = (ycsmall-0.5)*sy;

%% Interpolation

[Xcsuper, Ycsuper] = meshgrid(xcsuper, ycsuper);
[Xcsmall, Ycsmall] = meshgrid(xcsmall, ycsmall);

NormInterpIntensity = interp2(Xcsuper, Ycsuper, slice, Xcsmall, Ycsmall, 'makima');
NormInterpIntensity = NormInterpIntensity.*(sx).*(sy);

%% Saving

%Normalize the interpolated matrix so that i mWrite to png/jpg doesn't return just black and
white (because slice is double) - shift data so that minimum is zero and maximum is 1

NormInterpIntensity = double(NormInterpIntensity - min(NormInterpIntensity(:))) /
double(max(NormInterpIntensity(:)));

filename = horzcat(inputfile, '_', int2str(i));
i mWrite(NormInterpIntensity,horzcat(savepath,'Pngs\Half mW\0deg\'', filename, fileext),'png');

save(horzcat(savepath, filename, '.mat'), 'NormInterpIntensity');

end

end

```

A.2 Image Classification Algorithm

The following is the MATLAB code which represents the machine learning algorithm used to classify the speckle pattern images.

```
/**
/*  Sofia Brown  */
%% Optimized parameters (learning rate, epochs, and filter sizes) as of 3.19.21
%Set up training data
rootFolder = 'FolderContainingTrainingsImages';
categories = {'1deg','0deg'};
rng(0);
imds = imageDatastore(fullfile(rootFolder, categories), 'LabelSource', 'foldernames',
'FileExtensions','.png');

%Define Layers
layers = [
    imageInputLayer([256 320 1])

    convolution2dLayer(1,5,'Padding',2)
    batchNormalizationLayer
    reluLayer

    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(6,15,'Padding','same')
    batchNormalizationLayer
    reluLayer

    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(12,40,'Padding','same')
    batchNormalizationLayer
    reluLayer

    fullyConnectedLayer(2)
    softmaxLayer
    classificationLayer];

%Set training options
options = trainingOptions('sgdm', ...
    'InitialLearnRate',0.00001, ...
    'MaxEpochs' , 100, ...
    'Shuffle','every-epoch', ...
    'Verbose',false, ...
    'Plots','training-progress');

%Train
[net, info] = trainNetwork(imds, layers, options);
```

```

%% Plot and save the accuracy and loss data
figure();
subplot(2,1,1);
plot(info.TrainingAccuracy, 'b');
title('Training Progress');
grid on;
xlabel('Iteration');
ylabel('Accuracy (%)');
subplot(2,1,2);
plot(info.TrainingLoss, 'r');
grid on;
xlabel('Iteration');
ylabel('Loss');
fname = horzcat('Z:\Sofia\SavePlotLocation', '.png');
print(fname, '-dpng')

%Save the raw data as well, with the same name
save(horzcat('Z:\Sofia\SavePlotRawDataLocation'));
%% Set up test data
rootFolder = 'FolderContainingTestingImages';
imds_test = imageDatastore(fullfile(rootFolder, categories), ...
    'LabelSource', 'foldernames');

%Test one at a time
figure();
labels = classify(net, imds_test);

ii = randi(2);
im = imread(imds_test.Files{ii});
imshow(im);
if labels(ii) == imds_test.Labels(ii)
    colorText = 'g';
else
    colorText = 'r';
end
title(char(labels(ii)), 'Color', colorText);

%Test all the data at once and assess the accuracy by generating a confusion matrix
confMat = confusionmat(imds_test.Labels, labels);
confMat = confMat./sum(confMat,2);
mean(diag(confMat))

```