The Use of Machine Learning for Magneto-Optical Polarization Rotation Measurements

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

by

Marcia Ann Suria

Advisor: Prof. Irina Novikova

Prof. Daniel Vasiliu

Prof. Keith Griffioen

Williamsburg, Virginia
December 2023

Contents

Acknowledgments

I would like to extend my greatest gratitude to Professor Irina Novikova for her valuable guidance and mentorship throughout this thesis project. The progress of this research would not be possible without her constant support and feedback. I am grateful for the role she has played in my academic journey and for the encouragement she provided at crucial stages of this research. I would also like to thank Professor Daniel Vasiliu for his support and insightful feedback for this project.

List of Figures

List of Tables

Abstract

We propose a novel optical magnetometer incorporating artificial intelligence for sensitive magnetic field measurement. The core of this magnetometer is a Faraday rotation apparatus, consisting of a glass rod surrounded by a solenoid, linked to a camera via a multimode fiber. We employed a Convolutional Neural Network (CNN)-based image classification algorithm to distinguish various "speckle" pattern outputs of the fiber, produced by the change in polarization of a probe beam. By decreasing the polarization rotation, we can effectively detect even smaller magnetic field strengths. Our algorithm successfully detected a change in polarization as small as $1/256^\circ$ or 0.0039°, corresponding to a magnetic field of 0.03mT, with 100% accuracy. Additionally, our algorithm was successfully configured for multiclass classification, efficiently distinguishing between three distinct categories of speckle pattern outputs. It proficiently differentiates polarization angles as closely related as 0°, $1/64^\circ$, and $1/128^\circ$. With our current setup, our device supports mobile/remote configuration and is also handy in settings where electronic devices are not suitable.

Chapter 1

Introduction

1.1 Background

Magnetometers are instruments used to measure the strength and direction of magnetic fields. They have a wide range of applications, from mapping the Earth's magnetic field to detecting magnetic anomalies in infrastructures. As technology advances, the need for increasingly sensitive and accurate magnetometers continues to grow, making them an active area of research and development. Optical magnetometers, in particular, have shown great potential for achieving high sensitivity and compact design. They work by probing atoms in a magnetic field with light, then using the atoms' or light's response to detect the field. In this project, we continue the work of Sofia Brown '22 on developing a novel optical magnetometer by combining an optical fiber and machine learning [1]. Our research aims to further optimize the previously developed training algorithm for a more sensitive magnetic field measurement.

1.2 Theory

We focus on a specific type of optical magnetometer known as the Faraday rotation magnetometer. This device utilizes the change in polarization of light as it

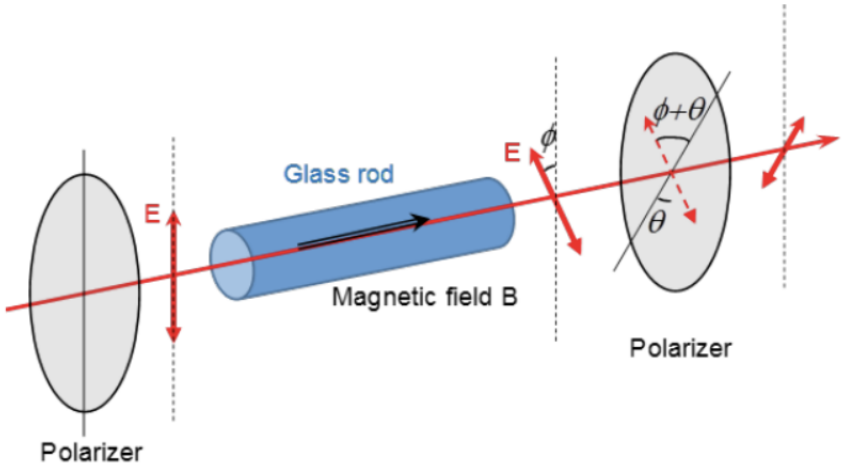

Figure 1.1: [1] A visualization of Faraday Effect, where light (red arrow) propagating through a material exposed to a magnetic field, B, undergoes change in polarization of angle ϕ.

interacts with atoms in a material exposed to a magnetic field. As shown in Figure 1.1, an external magnetic field B will rotate the light's polarization by an angle ϕ. By this Faraday Effect, the angle of polarization rotation is linearly proportional to the magnetic field according to

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

where C_V is the Verdet constant, which varies on the material, and l is the length of the material through which the light propagates. We can then measure magnetic field by measuring the changes in polarization.

The traditional setup includes a polarizer, which rotates the light's polarization by the angle ϕ, and a photodiode (placed after the polarizer), to measure the output. In our experiment, however, we use a novel approach of utilizing a multimode optical fiber followed by a CCD camera to detect polarization changes. When light is transmitted through the fiber, the different modes interfere with each other, creating a "speckle" pattern of varying intensity profile. As the change in polarization gets

smaller, the image features also change by a small amount, which is undetectable visually. This prompted the employment of an image classification algorithm to obtain the laser polarization from these speckle patterns. In the previous work, the training algorithm successfully detected a change in polarization as small as 0.6×10^{-4} degrees, corresponding to a magnetic field of 0.5 μT. We hope to push this limit further and achieve sensitivity up to the nanotesla range. With enhanced spatial resolution and physical compactness, our device would be suitable for settings that require small size and mobile/remote configuration.

Chapter 2

Experimental Technique

2.1 Experimental Setup

Our experimental setup consisted of a red laser beam, a Faraday rotation apparatus, a multimode fiber (MMF), a neutral-density (ND) filter, and a CCD camera (Figure 2.1). We used a modified version of the TeachSpin undergraduate Faraday rotation apparatus [2], by removing the laser, polarizer, and photodiode that came attached to it. This apparatus contains a 10 cm long glass rod surrounded by a 15 cm long solenoid that connects to a power supply. For preliminary work, we used a half waveplate, instead of the Faraday rotation apparatus, to polarize the laser beam transmitted through the multimode fiber. The ND filter is used to reduce the intensity of the laser beam to avoid overexposure. We also placed a cardboard box on top of the MMF, ND filter, and camera to shield from temperature changes and background light (Figure 2.2).

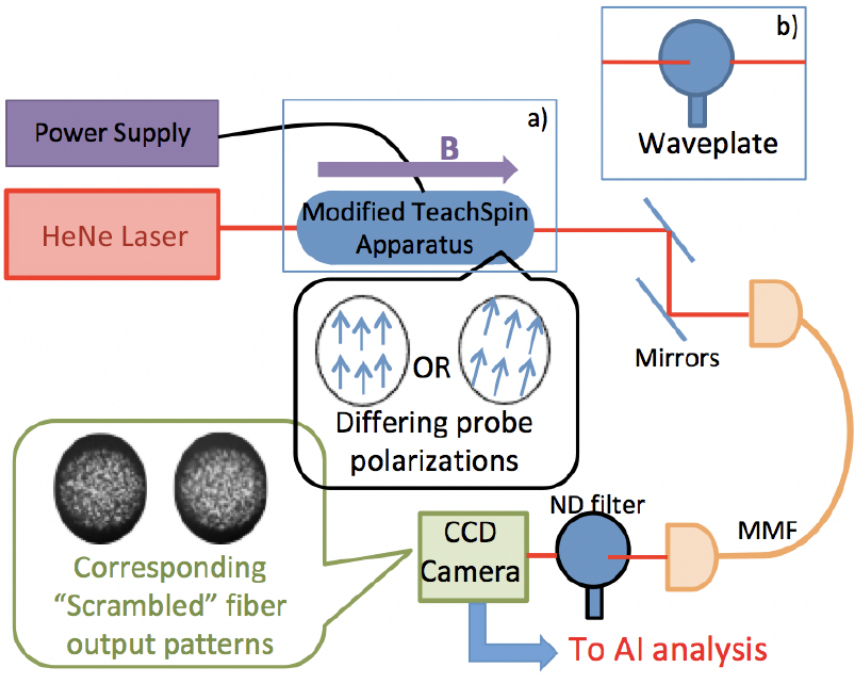

Figure 2.1: [1] A diagram of the experimental setup, consisting of the red HeNe laser, Faraday rotation apparatus, MMF, and CCD camera.

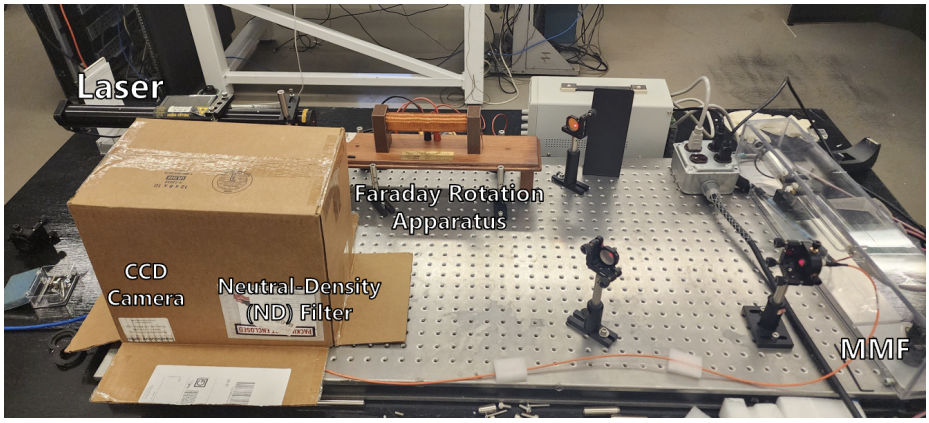

Figure 2.2: The experimental setup, consisting of the red HeNe laser, TeachSpin Faraday rotation apparatus, MMF, and CCD camera.

We first created our own calibration plot of current vs. change in polarization for our apparatus, shown in Figure 2.3. By placing a polarizer after the Faraday

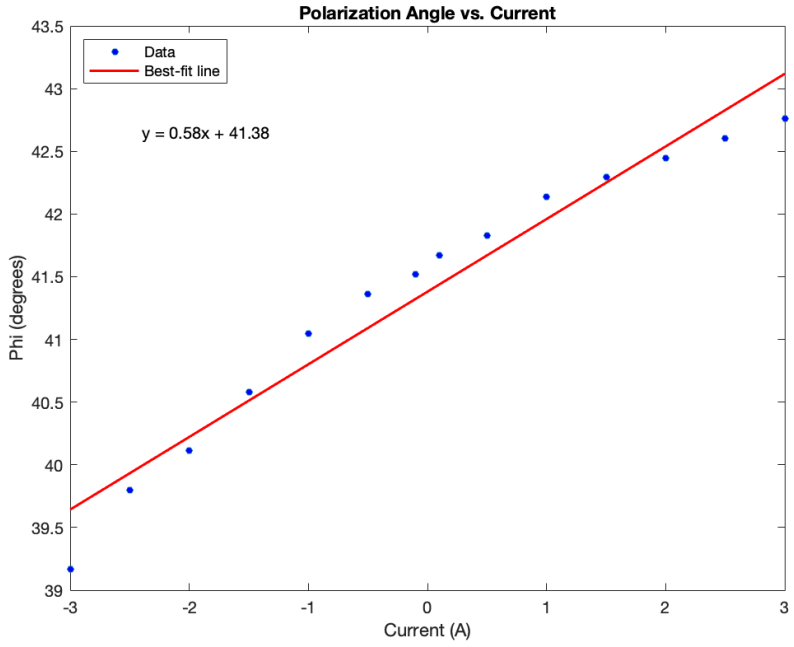

Figure 2.3: Calibration of the Faraday rotation apparatus: Polarization Angle ϕ (degrees) vs. Input Current I (A) plot.

rotation apparatus, we recorded the output power for different currents supplied to the solenoid. Then, using the fact that power, P, is related to the polarization angle, ϕ, by

$$P = P_0 sin^2(\phi) \tag{2.1}$$

where P_0 is the maximum power, we converted the power values to polarization rotation angle.

2.2 Data Collection

The data collection process consisted of setting the power supply to different current values and capturing a number of speckle pattern images for each polarization angle. In our initial approach to image acquisition, we utilized the video recording feature of MATLAB's Image Acquisition Tool, with the intent of extracting individual frames

for further analysis. However, this method presented significant drawbacks. Firstly, we realized that the video format employed inherently compresses the data, leading to a loss of critical details and a reduction in the resolution of the extracted frames. Moreover, opting for an uncompressed video format results in excessively large file sizes, making storage and processing unhandy. Additionally, there are limited support for easily extracting frames from uncompressed video formats, posing a technical hurdle. This combination of factors prompted us to explore alternative methods for image acquisition that could better meet our requirements for high-resolution and detailed imagery.

Our revised approach involves direct acquisition of images into a three-dimensional matrix, where each 'slice' of this matrix represents an individual frame. We then rescale and normalize the images to predefined dimensions, and save each 2D slice as a separate image file in PNG format. This approach not only preserves the high resolution and detail of each image but also allows for greater flexibility and precision in handling the data. We identified the optimal image dimension as rescaling from 1944 x 2592 to 256 x 320 matrices, effectively reducing computational time without substantial loss of image detail. The images were normalized to ensure the matrix entries lie between 0 and 1. Our approach also involves setting specific camera parameters — notably, the Black Level and Device Link Throughput Limit parameters, to establish ideal capture conditions. We used the default camera setting of "Mono 8" and turned off all "Auto" settings for the Exposure and Gain parameters.

These images are then split into training and testing sets and given to the training algorithm, which will output an accuracy vs. iteration and a loss vs. iteration plot. This will be explained further in Section 2.3. Throughout the semester, I have been working on optimizing the algorithm to detect decreasingly smaller polarization angles.

2.3 Image Classification Algorithm

The image classification platform we used is the built-in Convolutional Neural Network (CNN) in MATLAB, designed to classify images into two or more categories based on previous "learning". A CNN works by assigning importance, in the form of weights, to the distinct features present in images. These weights are initially set at random and are refined as the network 'learns' from a set of training data.

2.3.1 CNN Architecture

The CNN structure is composed of three key layers: alternating convolutional and pooling layers, followed by a fully connected layer, with each layer's specifics detailed below.

The first step of **convolution** involves the algorithm superimposing various filters (matrices of weights smaller than the original image) over the input image. These filters slide across the image and perform element-wise multiplication to produce feature maps. The network then employs **pooling** layers to reduce the spatial dimensions of these feature maps, therefore reducing computational load. A maximum pooling layer is usually utilized to help the network focus on the most prominent features of the image. The process of forming feature maps through convolutional and pooling layers may be repeated multiple times to extract increasingly complex and abstract features from the input images. Once the network has completed sufficient repetitions, a **fully connected** layer is introduced. It essentially acts as a classifier, analyzing the comprehensive set of features to determine the most likely category for each input image.

2.3.2 Training and Testing Process

The essence of CNN training lies in running the algorithm multiple times through its layered structure, continuously evaluating and fine-tuning its learning during each cycle. Initially, the network's convolutional filters are assigned random weights, which are refined throughout iterations (epochs) using a method called backpropagation. This is done by generating a loss function, specifically Cross Entropy Loss, which measures the difference between the network's predictions and the actual image categories. The loss function plays a pivotal role in guiding the network to adjust its weights, especially those contributing to errors in classification accuracy.

Several key parameters could be optimized during training to enhance the CNN's learning efficiency. The number of **epochs**, or training iterations, balances between sufficient learning and practical runtime. The **size of the convolutional filters** and the **number of filters per layer** are adjusted to capture the necessary level of detail in image features, with larger and more filters generally improving accuracy at the cost of increased computational time. The **learning rate**, a crucial parameter, determines the magnitude of weight adjustments during backpropagation. A properly tuned learning rate ensures that the network neither converges too rapidly on suboptimal solutions nor gets trapped in local minima, allowing for optimal performance during the training phase.

The last step of the image classification process involves exposing the algorithm to new images that it hasn't encountered before, yet are similar to the training images. The algorithm's accuracy in correctly classifying these new images is then evaluated.

2.3.3 Accuracy Analysis

During training, the algorithm generates learning curves, plotting **classification accuracy vs. training epochs** and the **loss vs. training epochs**. A clear indication

of successful training is observed when there is a steady increase in accuracy and a corresponding decrease in loss as training progresses. While accuracy is a discrete measure, reflecting the correctness of the algorithm's predictions in a binary format after each training cycle, loss is a continuous function quantifying the deviation of the algorithm's predictions from the actual outcomes. An algorithm displaying both high accuracy and low loss is typically a sign of a robust training, making it likely to succeed in future testing situations.

For the testing process, a critical tool of assessment is the **confusion matrix**. This matrix quantifies the algorithm's performance by showing the percentage of correct identifications and misclassifications for each category. An optimal confusion matrix would display high values (ideally 1s) along the diagonal, indicating accurate categorization, and low values (ideally 0s) off the diagonal, signifying few misclassifications. The overall output accuracy of the algorithm can be quantified by averaging the values along the diagonal of the confusion matrix.

2.3.4 Multiclass Classification

We also configured our algorithm for multiclass image classification (using CNN as well), to classify images into more than two different categories. In a multiclass classification, the CNN architecture, particularly in the fully connected and output layers are altered to have three nodes, each corresponding to a classification category. Consequently, the performance of the algorithm is evaluated through a 3x3 confusion matrix, with each row representing the actual class of the images, and each column representing the predicted class by the algorithm.

Chapter 3

Results

3.1 Binary Image Classification

For preliminary work, I analyzed image differences for relatively large polarization angles (between 4 to 40 degrees) using a half waveplate. The different features of the images started to be visually indistinguishable as the polarization degree gets smaller than 10° (shown in Figure 3.1). For a polarization angle of 2° and smaller, I replaced the waveplate with a Faraday rotation setup.

Table 3.1 shows the accuracy results for polarization angle differences of 0° vs. 40° to 0° vs. 4°. I initially used a sample size of 100, splitting 50 of them to the training and the other 50 to the testing set. This returned an accuracy of 100% up until 0° vs. 10°. The algorithm started having troubles for 0° vs. 4° with the same parameters as the previous ones. It only returned an accuracy of 50% and the training plot was too fluctuating (see Figure 3.2). So I decreased the learning rate (step size at each iteration while moving toward a minimum loss function) and increased the sample size to 300, splitting 240 to the training and 60 to the testing set. This significantly improved the training plot and returned an accuracy of 100%. As seen in Figure 3.2, the algorithm also took a longer time to reach 100% accuracy for smaller angles.

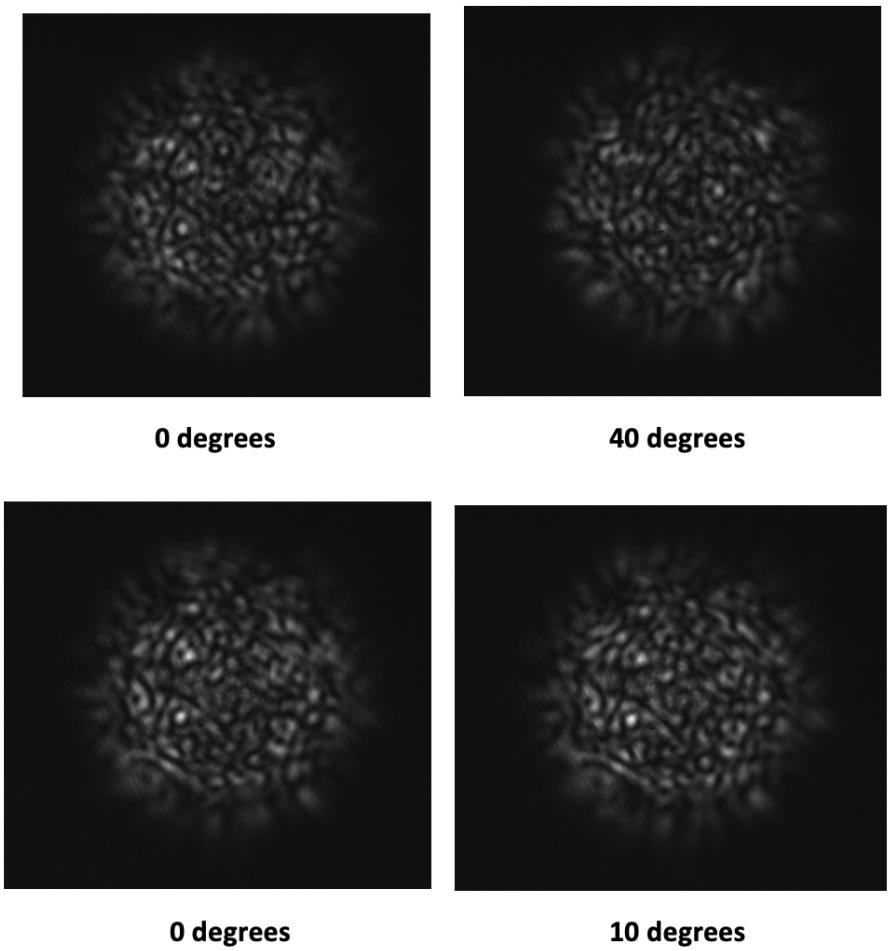

Figure 3.1: Image comparison of 0° vs. 10 degrees polarization

Polarization Angle (degrees)	Accuracy	Sample Size	Learning Rate
0° vs. 40°	1	100	0.0001
0° vs. 20°	1	100	0.0001
0° vs. 10°	1	100	0.0001
0° vs. 4°	1	300	0.00001

Table 3.1: Accuracy results for polarization angle differences of 0° vs. 40° to 0° vs. 4°

Using the Faraday rotation apparatus (instead of the waveplate), I repeated the same data collection procedure for even smaller angles. Table 3.2 shows the accuracy

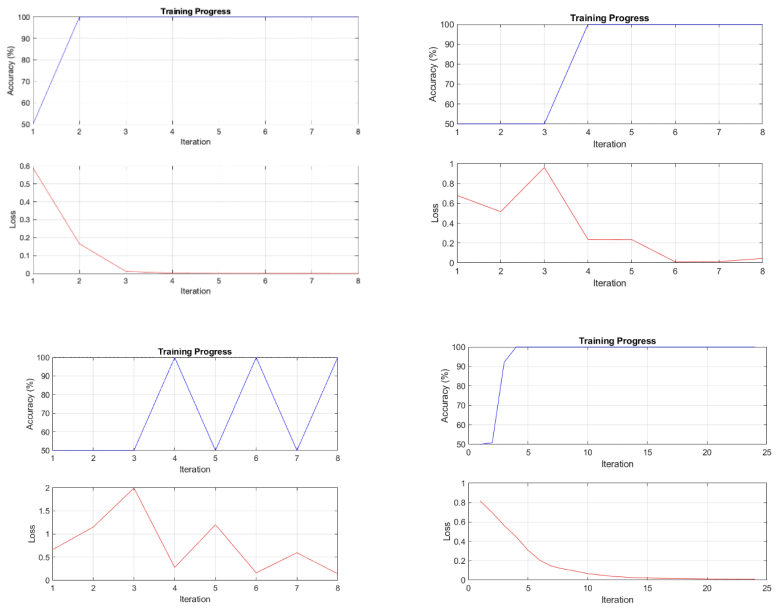

Figure 3.2: Comparison of learning curves for training between 0° vs. 40° and 0° vs. 4°. The top left is for 0° vs. 40°, the top right is 0° vs. 10°, the bottom left is 0° vs. 4° (not optimized), and the bottom right is the optimized plot for 0° vs. 4°.

results for polarization angle differences of 0° vs. 2° to 0° vs. 1/4°. For 0° vs. 2°, I initially used a sample size of 300, and the algorithm returned an accuracy of 1. However, as predicted, the training plot was not as smooth as the previous one for 0° vs. 4°. I tried increasing the sample size to 500 (400 for training and 100 for testing), and the plot improved. With this same sample size, the algorithm returned an accuracy of 100% for 0° vs. 1° (as shown in Table 3.2). However, for 0° vs. 1/2°, I had to further increase the sample size to 1000 (800 for training and 200 for testing), and found that this is the current most optimum number for small angles. Although, from this angle and smaller, it is apparent that the training plot is more fluctuating and not as smooth (Figure 3.3), even though the algorithm still returned an accuracy of 1. For 0° vs. 1/4°, I also had to increase the epoch from 8 to 12, as the algorithm fluctuated longer before reaching 100% accuracy.

Polarization Angle (degrees)	Current	Accuracy	Sample Size	Epochs
0° vs. 2°	3.45A	1	500	8
0° vs. 1°	1.72A	1	500	8
0° vs. 1/2°	0.86A	1	1000	8
0° vs. 1/4°	0.43A	1	1000	12

Table 3.2: Accuracy results for polarization angle differences of 0° vs. 2° to 0° vs. 1/4°

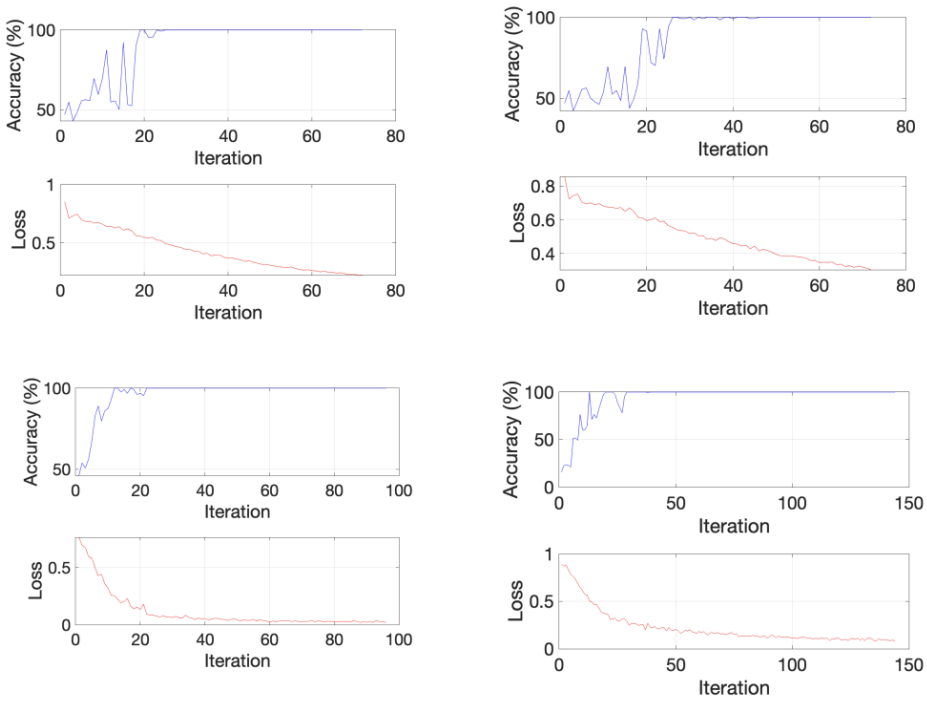

Figure 3.3: Comparison of learning curves for training of 0° vs. 2° to 0° vs. 1/4°. The top left is for 0° vs. 2°, the top right is 0° vs. 1°, the bottom left is 0° vs. 1/2°, and the bottom right is 0° vs. 1/4°.

Further decreasing the angle to 0° vs. 1/8°, we found that the algorithm started to struggle even more, both for the training and also testing process. We experimented with a lot of different filter sizes and number of filters for the convolution layer, particularly focusing on the third layer. The third convolution layer is crucial as it is a deeper layer responsible for extracting more complex and abstract image features. The optimum values we found are as follows:

- **Layer 1:**
  - Filter size: 1
  - Number of filters: 12 (previously 5)
- **Layer 2:**
  - Filter size: 8
  - Number of filters: 24 (previously 15)
- **Layer 3:**
  - Filter size: 20 (previously 12)
  - Number of filters: 60 (previously 40)

This adjustment improved the algorithm's accuracy in distinguishing these finer angles, even as small as 0° vs. 1/256°. Although it came with a slight trade-off: a longer runtime for the algorithm. With the initial filter configuration, the average training time was around 10-20 minutes, but with the new one, it became 40-60 minutes. With these changes, we also found that an epoch count of 8 (instead of 12) is enough to reach a 100% accuracy, except for 0° vs. 1/128° and 0° vs. 1/256° (which needs 12). Figure 3.4 shows a comparison of the training progress achieved with the previous and updated filter size configurations for 0° vs. 1/128°. Table 3.3 shows the accuracy

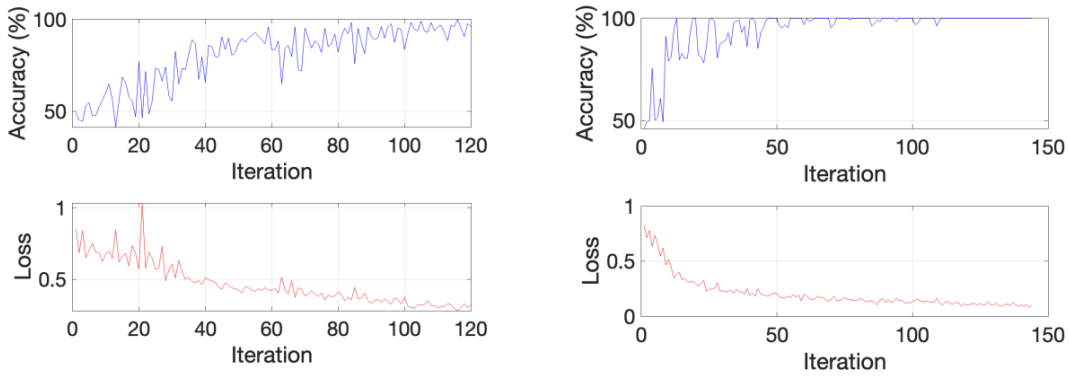

Figure 3.4: Comparison of training plots for 0° vs. 1/128° for different filter sizes. The plot on the right corresponds to the results using the current optimized filter sizes and number of filters.

results and corresponding magnetic fields for polarization angle differences of 0° vs. 1/8° to 0° vs. 1/256°, where our algorithm successfully achieves a 100% accuracy for all the angles. Note that the magnetic field values were calculated using Eq. 1.1, and we used a value of $C_V = 23$ rad/(T·m), which is the value for 650nm light. As shown in Figure 3.5, the training plots were also ideal for all of these angles, with more fluctuations for smaller angles. Hence, our algorithm successfully detects a change in polarization as small as 1/256°, or 0.0039°. Using Equation 1.1, this corresponds to a magnetic field of 0.03mT.

Polarization Angle (degrees)	Current	Magnetic Field (mT)	Accuracy	Epochs
0° vs. 1/8°	0.21A	0.95	1	8
0° vs. 1/16°	0.10A	0.47	1	8
0° vs. 1/32°	0.05A	0.24	1	8
0° vs. 1/64°	0.02A	0.12	1	8
0° vs. 1/128°	0.01A	0.06	1	12
0° vs. 1/256°	6.7mA	0.03	1	12

Table 3.3: Accuracy results for polarization angle differences of 0° vs. 1/8° to 0° vs. 1/256°

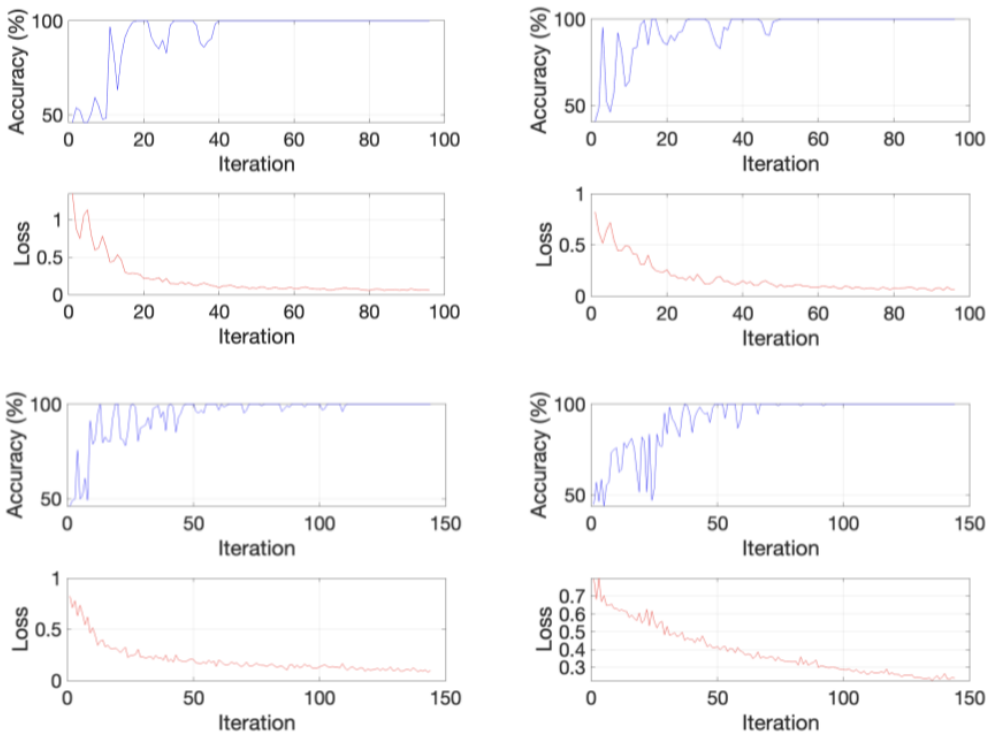

Figure 3.5: Comparison of learning curves for training of 0° vs. 1/32° to 0° vs. 1/256°. The top left is for 0° vs. 1/32°, the top right is 0° vs. 1/64°, the bottom left is 0° vs. 1/128°, and the bottom right is 0° vs. 1/256°.

3.2 Multiclass Image Classification

To enhance the efficiency of our image classification process, we configured our algorithm to accomodate multiclass image classification, specifically to classify three different polarization angles. The approach is similar to the binary classification, although this model now involves three different categories for the training and testing set. Our first training attempt involves classifying 0° vs. 1° vs. 2°, where we used a sample size of 500 (400 for training and 100 for testing) for each angle. Our algorithm returned a 100% accuracy using our initial convolution filter configuration and shows an ideal training plot (as shown in Fig. 3.6). We also visualized the confusion matrix,

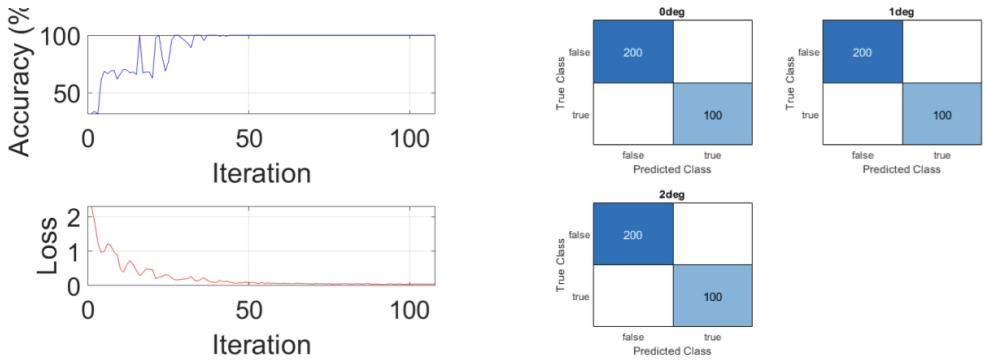

Figure 3.6: Multiclass training plots and confusion matrix visualization for 0° vs. 1° vs. 2°.

showing that each test images were successfully classified and labeled as the actual polarization angle. For example, in the 0deg class, the model predicted 'true' (true positive) for the actual 0deg images 100 times, and predicted the rest of the 200 images of 1deg and 2deg as 'false' (true negative). This suggests a perfect classification for the 0deg class.

With the newly updated convolution filter configurations, we also classified polarization angles of 0° vs. 1/8° vs. 1/32° and 0° vs. 1/64° vs. 1/128°. For these smaller angles, we used a sample size of 1000 (800 for training and 200 for testing). The resulting training plots and confusion matrix are shown in Fig. 3.7. We can see that for each polarization angle, the algorithm successfully classified the actual 200 images to the correct category. This results in an output accuracy of 1 for both attempts.

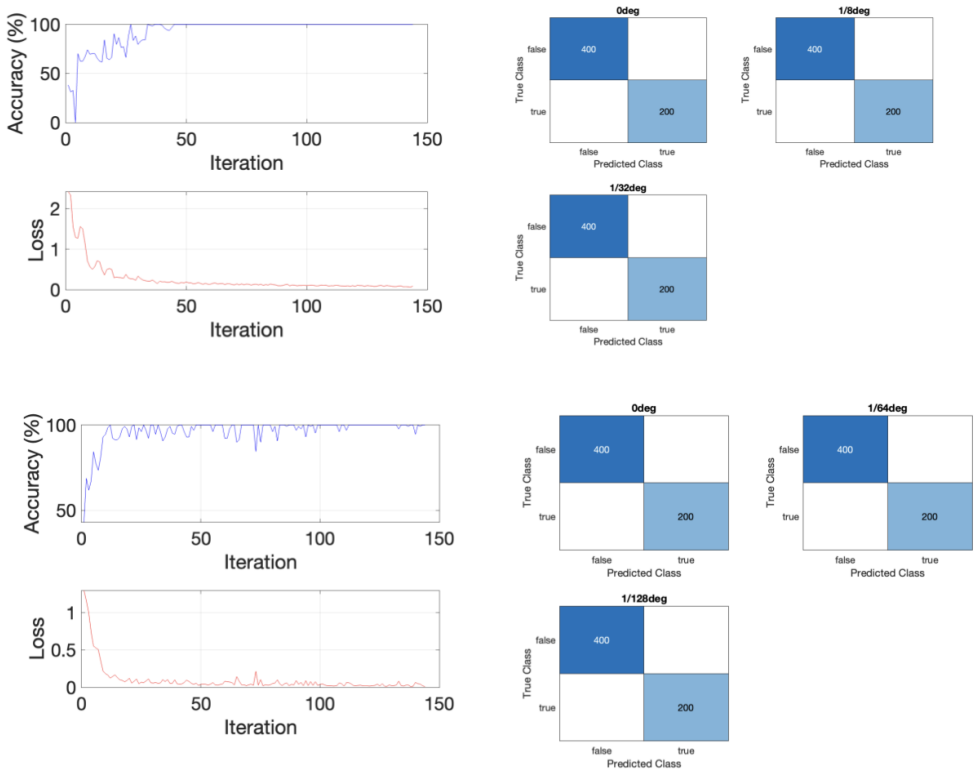

Figure 3.7: Multiclass training plots and confusion matrix visualization for 0° vs. 1/8° vs. 1/32° (top) and 0° vs. 1/64° vs. 1/128° (bottom).

Chapter 4

Conclusion

Our training algorithm successfully detected a change in polarization as small as $1/256^\circ$ or 0.0039°, corresponding to a magnetic field of 0.03mT, with 100% accuracy. Throughout the semester, we also developed an optimum data collection process, which includes:

- Collecting a large sample size of 100 (splitting 800 for training and 200 for testing set)
- Rescaling the input images from 1944 x 2592 to 256 x 320 matrices to reduce computational time
- Normalizing the images so their matrix entries fall between 0 and 1
- Using specific learning parameters (setting the learning rate to 0.0001 and the convolutional filter sizes to the specified values)

We also successfully configured our algorithm to perform multiclass image classification, with it returning a 100% accuracy for angles as small as 0° vs. $1/64^\circ$ vs. $1/128^\circ$. Future work includes further optimizing the algorithm to detect even smaller changes in polarization angle for a more sensitive magnetic field measurement. This could be

done by optimizing the learning parameters, improving the quality of the data, or using more complex machine learning models.

Bibliography

[1] Brown, Sofia. "Optical Fiber-Linked Magnetometer Employing Artificial Intelligence for Magnetic Field Measurement". (2022). Undergraduate Honors Theses. William & Mary. Paper 1801. https://scholarworks.wm.edu/honorstheses/1801

[2] https://www.teachspin.com/faraday-rotation

[3] The MathWorks Inc. Deep Learning Toolbox. (2022). Massachusetts: The MathWorks Inc. https://www.mathworks.com/help/deeplearning/

[4] Deshpande, A. "A Beginners Guide to Understanding Convolutional Neural Networks." (2016). https://adeshpande3.github.io/A-Beginner27s-Guide-To-Understanding-Convolutional-Neural-Networks/

[5] 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/

[6] 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

[7] Mamun, I. "A Simple CNN: Multi-Image Classifier". Towards Data Science. (2019). https://towardsdatascience.com/a-simple-cnn-multi-image-classifier-31c463324fa

[8] Javed, A. "Multiclass Image Classification using CNN". (2021). https://www.kaggle.com/code/ahmadjaved097/multiclass-image-classification-using-cnn