Comparing CNN Architectures for Visual Impairment Assistive Technology Applications

Allen Shelton

Abstract

Visual Impairment is a problem among people of all demographics, and much research has been done to offer people with visual impairment options for safely navigating their environment. The project I am proposing will build upon work that I have already done in this area of research, where I developed a prototype for a deep learning-enabled guide in which a camera recognizes objects using a Convolutional Neural Network, and then a text-to-speech algorithm verbalizes what objects are identified. My project will focus on the CNN portion of that work, and trying to improve the prediction accuracy. The main things that will be explored to improve the accuracy will be the collection of more data, the use of different architectures for transfer learning, and utilizing data augmentation during training.

1 Problem Statement

Advancements in artificial intelligence and deep learning have had a significant impact on the healthcare industry. These advances have specifically had a notable impact on people with visual impairments. An estimated 295 million people globally have moderate to severe visual impairment, and 43.3 million people are legally blind [1]. People that fall into these categories of visual impairment frequently experience a reduced quality of life because they lack the ability to navigate their environment without assistance. Much in the way of assistive services are available, including guide dogs and walking sticks, but these solutions lack the ability to provide enough context for visually impaired people to know what's around them. That is why object detection and recognition solutions using deep learning have been heavily explored: to give people with visual impairment the capabilities to understand what's in their surroundings.

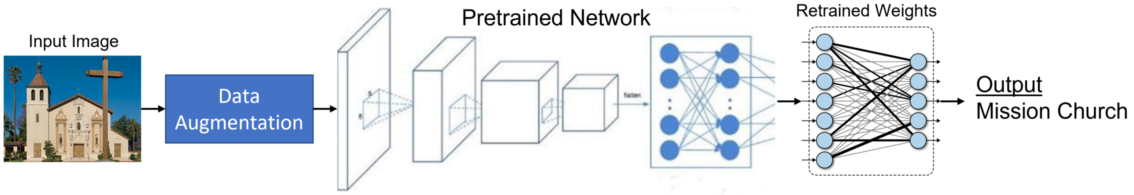


Figure 1: Block Diagram of Proposed Solution

2 Introduction

My work for this project will improve upon work that I've already done in using image recognition to develop assistive technology for blind and visually impaired people [2]. My original work focused on visual understanding in the context of a college campus. This involved training a Convolutional Neural Network (CNN) on images of common items found in a university as well as the university buildings themselves. Then, I used a webcam to feed successive frames into the trained CNN so that it could classify the frames in real time. Then, at regular intervals, a text-to-speech algorithm would verbalize what the CNN sees so that visually impaired people will know what's in their surroundings.

I believe the results that came from that work could be improved. The prediction accuracy was good, but more can be done to make the accuracy even higher. I plan to try three main methods to achieve this. The first is collecting more data. The data I used to train this neural network was compiled both from images taken from the Internet as well as images taken on my phone. In total I only had a few hundred images for training. I believe by collecting more data, the CNN can learn a better representation of the categories of images and achieve better prediction accuracy.

The second method is to test different neural network architectures. My original work used transfer learning, where an existing architecture with pretrained weights is used as the starting point for another task. The existing architecture I used was AlexNet [3], a very important CNN that popularized the use of deep learning. Even though AlexNet was a very good CNN at the time it was published, it's been over ten years since then, and many other architectures have been introduced with great performance as well. My goal is to compare the performance of AlexNet with other neural networks, namely, VGG Network [4], GoogleNet [5], and ResNet [6], to see if these newer architectures can outperform AlexNet.

The third method proposed is data augmentation. Data augmentation is a very low-cost and effective preprocessing step in training CNNs in the absence of a large dataset. By applying data augmentation during training, data augmentation is creating more data to train, as well as helping to avoid overfitting. The addition of data augmentation will be explored for my dataset to see if it can further improve the classification accuracy.

3 Related Work

This project touches on a few different concepts, all of which are fairly well-researched

Visual Impairment Solutions. Much work has been done in offering solutions to individuals with visual impairments to improve their quality of life. The authors of [8] have developed a reader for visually impaired people that uses a CNN as well as Long Short Term Memory (LSTM) to both verbalize written text and describe images in reading materials. In [9], an Extreme Learning Classifier is combined with CNNs to classify toilet signs for blind guidance. The authors of [14] developed smart glasses for blind navigation. These glasses combine deep learning to identify obstacles with stereo cameras to calculate distances between the obstacle and the user.

Transfer Learning. Transfer learning is frequently used in applications where access to vast amounts of data cannot be found. Transfer learning is applied to the field of biometrics in [18], where AlexNet is retrained to recognizes subjects based on pictures of their ears. Transfer learning has also been applied to food image recognition [10], which has benefits in health and marketing. The authors of [11] apply transfer learning to speed up the learning of action rule for the autonomous navigation of model cars.

Data Augmentation Data Augmentation is also used in application with a small amount of data to avoid overfitting. In [12], data augmentation is applied to inertial sensor data to effectively classify the behavior of cattle in the absence of a large-scale dataset to train the CNN. The authors of [13] use data augmentation to extend vital and lifestyle data for the detection of anomalies in the condition of elderly people. In [15], the authors use an intraoral image dataset to train a CNN with and without data augmentation, and their results show that using data augmentation gave better mean Average Precision (mAP) than without it.

4 Solution/Implementation

A rough diagram of my solution is shown in Figure 1. The diagram shows how the training process will work. Our dataset will be used to train the CNN architecture after first being passed through an image augmentation step. The pretrained network weights from AlexNet, VGG, ResNet, and GoogleNet will be frozen, and only the last few layers will have their weights fine-tuned. This is the benefit of transfer learning, that we have the ability to re-purposed an existing network for a new problem. The modified final layers weights will give the output, which will be a classification into one of the class the network was trained to identify. I will go over each aspect of the solution in a little more detail.

4.1 The Dataset

The dataset used for my original work was compiled both from images found from the Internet and images taken from my smartphone. The idea for the categories of images was to help visual impaired people gain scene understanding in the context of a college campus, specifically the Santa Clara University campus. I have broken up the images I've collected into 2 groups, where each group contains image pertaining to different categories. These groups and their associated image classes are shown in Table 1.

Group 1	Group 2
Bed	Benson Center
Bench	Bronco Statue
Car	Casa Italiana RLC
Couch	Daly Science
Door	Dowd Art Building
Elevator	Graham RLC
Exit Sign	Kenna Hall
Fire Extinguisher	University Library
Folder/Binder	Mission Church
Fork	O'Connor Hall
Knife	
Palm Tree	
Pen/Pencil	
Stairs	

Table 1: Image Classes for Dataset

The image classes in Group 1 pertain to general objects that a student would commonly see around SCU or in any of the dorm rooms. The classes in Group 2 are specific buildings and landmarks that are on SCU's campus. With both of these groups, students at SCU with visual impairment can gain scene understanding in both indoor and outdoor settings.

Increasing the number of images in the dataset can help to improve the prediction accuracy of the system. In my original work, my Group 1 dataset contained only 283 images, and my Group 2 dataset contained 171 images, with some image classes having less than 10 images. Because of this, the neural network from my previous work had a difficult time with recognizing some classes of images. Adding more images to these categories gives the CNN the ability to learn the relevant features of all image categories, leading to better prediction results. Not only do I want to increase the quantity of my data, but the quality of it as well, making sure that each class has close to the same number of images, and that a variety of different views of each image class are included in the data to make the system more robust [7].

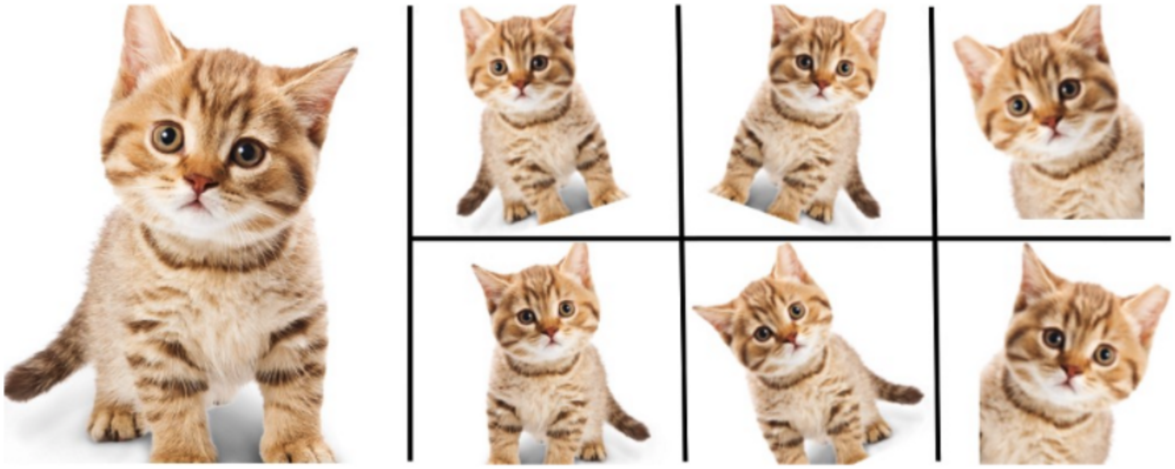


Figure 2: Image Augmentation Example

4.2 Data Augmentation

Data Augmentation is a very useful technique with a limited data set. In Data Augmentation, images can be transformed in a number of different ways before being passed into the neural network for training. This technique basically creates more data to train the network, and helps to avoid overfitting so that it generalizes well to test images. These transformations can be rotations, translation, reflections, dilations, changes in illumination, among others. An example is shown in Figure 2. We have a limited amount of data in a limited set of conditions, but in the real world, these objects exist in even more conditions, and so image augmentations help the CNN feature extractions to be invariant to these changes in conditions [16]. Using image augmentations can increase the robustness of the system, leading to better prediction accuracy.

4.3 Transfer Learning

Even with the plan to gather more data for this project, many state-of-the-art neural networks used for image recognition are trained on hundreds of thousands or even millions of images [17]. Gathering that much labeled data is a very difficult task, and with the time and resources I have it would be impossible. So, instead of creating my own CNN architecture and training it from scratch, I will utilize transfer learning. With transfer learning, one can rely on past knowledge gained from another neural network that was trained for a similar problem. We can use that architecture with weights that are already pretrained, and we only need to modify and retrain the last few layers. All other weights will be frozen. Because of transfer learning, training a neural network from scratch is not necessary, and because the weights are pretrained, much less data is required to achieve high prediction accuracy [18].

In my original project, I used transfer learning with the AlexNet CNN. This was a revolutionary

CNN architecture for the field of deep learning, but the original paper is over 10 years old now. Research into deep learning has progressed rapidly over the past 10 years because of AlexNet, and today newer and better performing architecture have been presented. I would like to compare the performance of AlexNet to newer CNN architecture on my dataset to see if they can give better prediction accuracy, this is a common practive given different learning algorithms might work better for specific types of data [19]. The other architectures I plan to use are VGG, GoogleNet, and ResNet. I will briefly explain each of them.

AlexNet. AlexNet's architecture consists of 8 total layers: 5 convolutional layers and 3 fully connected layers (see Figure 3a). All layers have ReLu activations, and convolutional layers have pooling and dropout layer proceeding them. AlexNet has a total of 650,000 neurons and 63.2 million trainable parameters. AlexNet was the winner of the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [17], where it achieved a top-5 error rate of 15.3%.

VGG. VGG is a deep CNN architecture. It is characterized by its simplicity because besides a single fully connected layer and some pooling layers, all other layers are convolutional (see Figure 3b). VGG comes in two flavors: VGG-16 and VGG-19, where the numbers denote the number of convolutional layers. The large number of layers is why VGG is referred to as a "deep" architecture. VGG was the runner up in the 2014 ILSVRC, with a top-5 error of 7.32%.

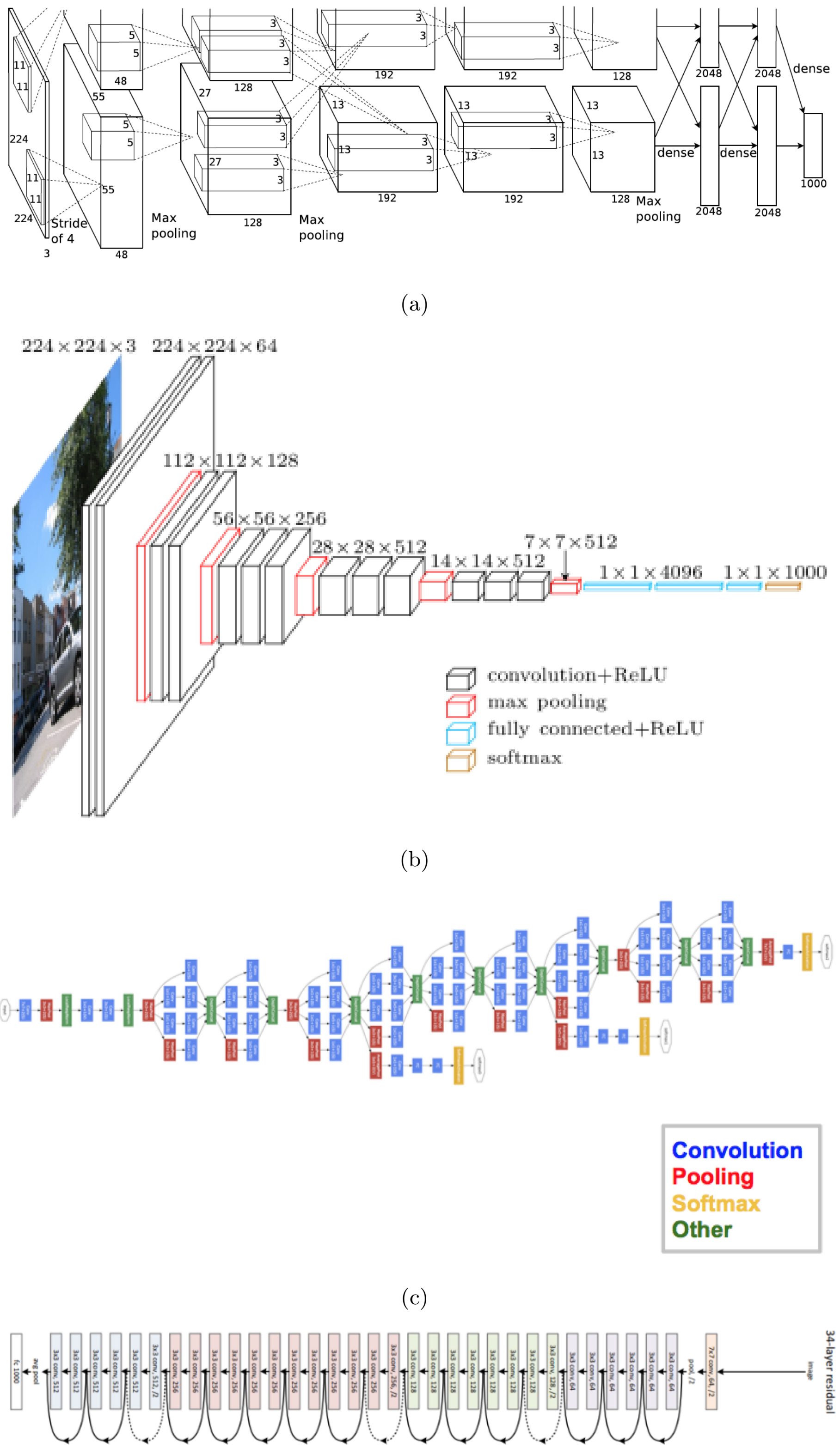
GoogleNet. GoogleNet is a CNN based on the Inception architecture, which allows the network to choose multiple filter sizes for each Inception block (see Figure 3c). GoogleNet utilizes other methods such as 1x1 convolutions, global average pooling, and auxillary classifiers to help with vanishing gradients. GoogleNet was the winner of the 2014 ILSVRC, with a top-5 error rate of 6.67%.

ResNet. ResNet, or Residual Network, seeks to solve the problem of vanishing or exploding gradients that was common in other CNN architectures of its time. It does this by using a technique called skip connections, where activation of one layers connect to other layers by skipping some in between (see Figure 3d). These skip connections form what's called a Residual block, and many of them are stacked together to create ResNet. So if any layer in the network is limiting its performance, it will be skipped by regularization. ResNet has 34 layers with skip connections throughout. ResNet has 152 layers, and also won the 2015 ILSVRC with a top-5 error rate of 5.71%.

5 Results

5.1 Baseline Results

First, we'll discuss the baseline results. The original datasets had 283 images in Group 1 and 171 images in Group 2. The baseline results actually use data augmentation during training, but in my



(a)

(b)

(c)

(d)

Figure 3: CNN Architectures for (a) AlexNet, (b) VGG, (c) GoogleNet, and (d) Resnet

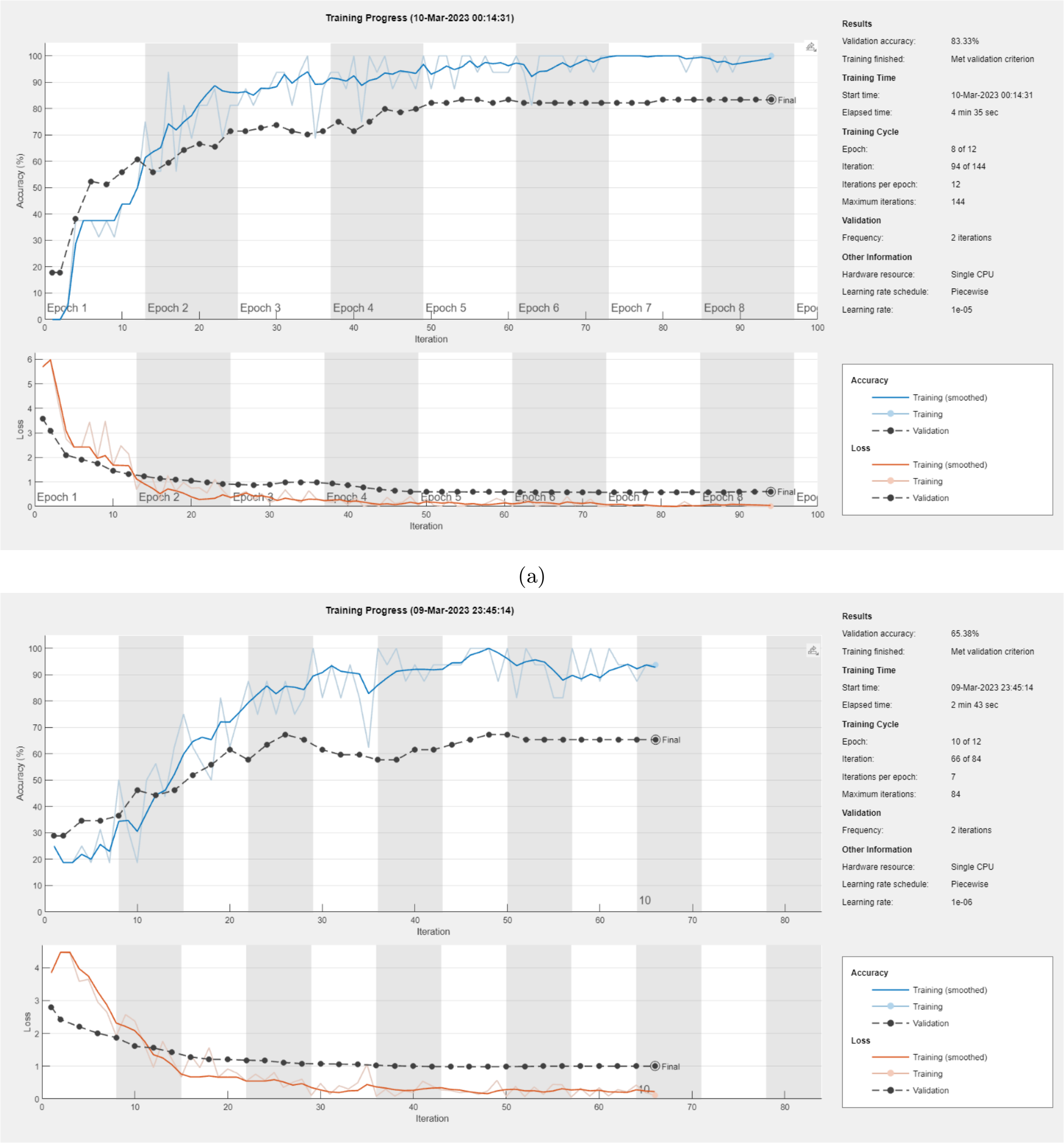
final results, I'll be testing each CNN with and without data augmentation to clearly show the effects that data augmentation has on performance. The data augmentation used for the baseline results are random rotations of the images up to 20 degrees in either direction, as well as random horizontal and vertical translations of the images up to 3 pixels.

The accuracy and loss plots for both the training and validation data are shown in Figure 4a for the Group 1 dataset and Figure 4b for Group 2. The baseline results for Group 1 are quite good. The training accuracy is able to get really close to 100% accuracy by the end of training. However, the validation accuracy is not able to get that high, and ends with an accuracy of 83.33%. The Group 2 accuracy is not as good. You can see that the training accuracy fluctuate a lot, and it has a harder time getting to a high value. Additionally, there is a larger difference between the final validation accuracy and the training accuracy, indicating the CNN doesn't do a great job a generalizing to the test images. Similar trends can be seen in the loss plots as well.

The baseline confusion matrices are shown in Figure 5. From the Group 1 results, one observation that can be made pertains to image classes 5 and 6, corresponding to Couch and Door. You can see several image that were predicted as those classes even though they belonged to different classes. You can also see there were a significant number of images that belonged to class 1, which is Bed, yet were classified as being in other classes. In Group 2, you can see even more mispredictions, especially for class 7, which is Kenna.

The Recall, Precision, and F1 Scores are shown in Figure 6. Note that the F1 Score is simply the harmonic mean of Recall and Precision. For Group 1, you can see that many image classes performed well, such as Bike, Fire Extinguisher, and Palm Tree, while others struggled, like Bed, Bench, or Couch. There is no data for the Folder/Binder and Knife classes because there were so few images for those categories in the dataset that results could not be given there. For Group 2, most of the image classes struggled, with the worst performing ones being Dowd, Graham, Kenna, and O'Connor.

Finally, ROC Curves for the baseline results are shown in Figure 7. Usually, ROC curves are useful for binary classification problems. However, there are a couple of different ways to create ROC for multi-class classification problems. I am using the OvA (One vs. All) approach. In this approach, one class label is treated as the positive class, and all other classes are treated as the negative class, reducing the problem to binary classification. This is done for each image class, and the ROC curves are generated for each case. Curves that are red are those whose area under the curve is less than 0.95, indicating suboptimal performance. An ideal ROC curve has area under the curve of 1, with the optimal operating point being in the top left corner. From these plots, you can see for Group 1 that it is the Exit Sign and Stairs image classes that perform the worst, and for Group 2, half of the image class are performing suboptimally.



Results

Validation accuracy:	83.33%
Training finished:	Met validation criterion

Training Time

Start time:	10-Mar-2023 00:14:31
Elapsed time:	4 min 35 sec

Training Cycle

Epoch:	8 of 12
Iteration:	94 of 144
Iterations per epoch:	12
Maximum iterations:	144

Validation

Frequency:	2 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Piecewise
Learning rate:	1e-05

(a)

Results

Validation accuracy:	65.38%
Training finished:	Met validation criterion

Training Time

Start time:	09-Mar-2023 23:45:14
Elapsed time:	2 min 43 sec

Training Cycle

Epoch:	10 of 12
Iteration:	66 of 84
Iterations per epoch:	7
Maximum iterations:	84

Validation

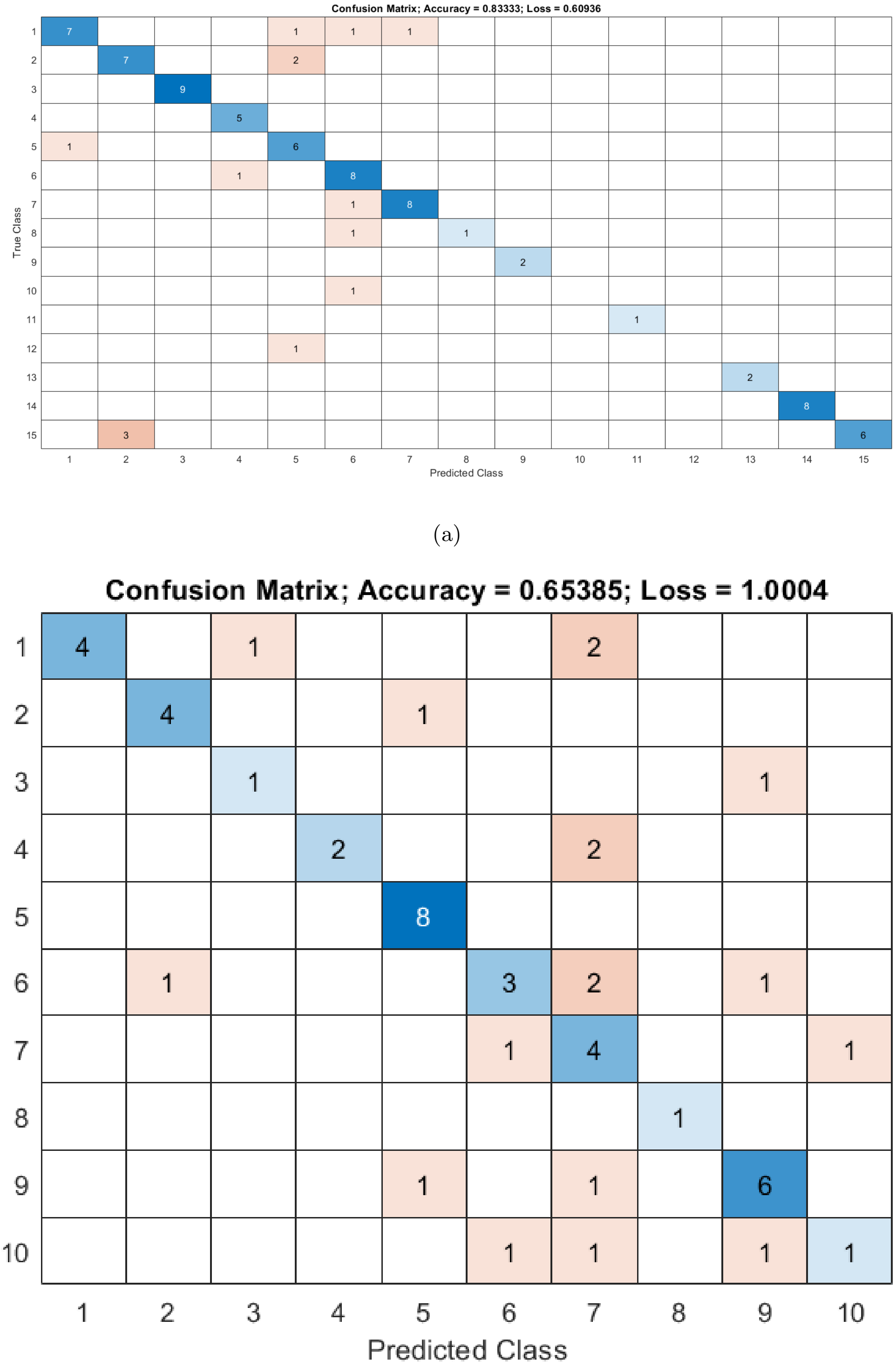
Frequency:	2 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Piecewise
Learning rate:	1e-06

(b)

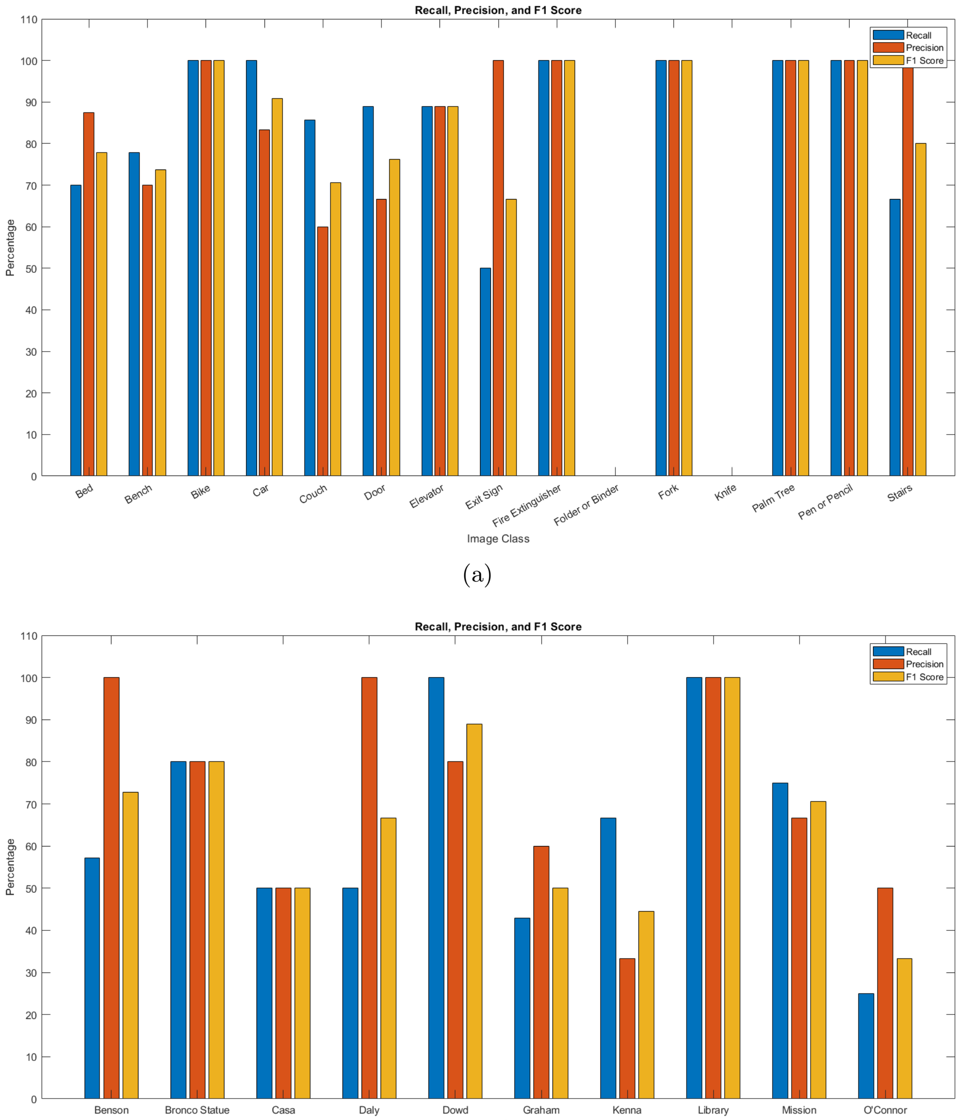
Figure 4: Baseline Accuracy and cross-entropy loss for (a) Group 1 Dataset and (b) Group 2 Dataset



(a)

(b)

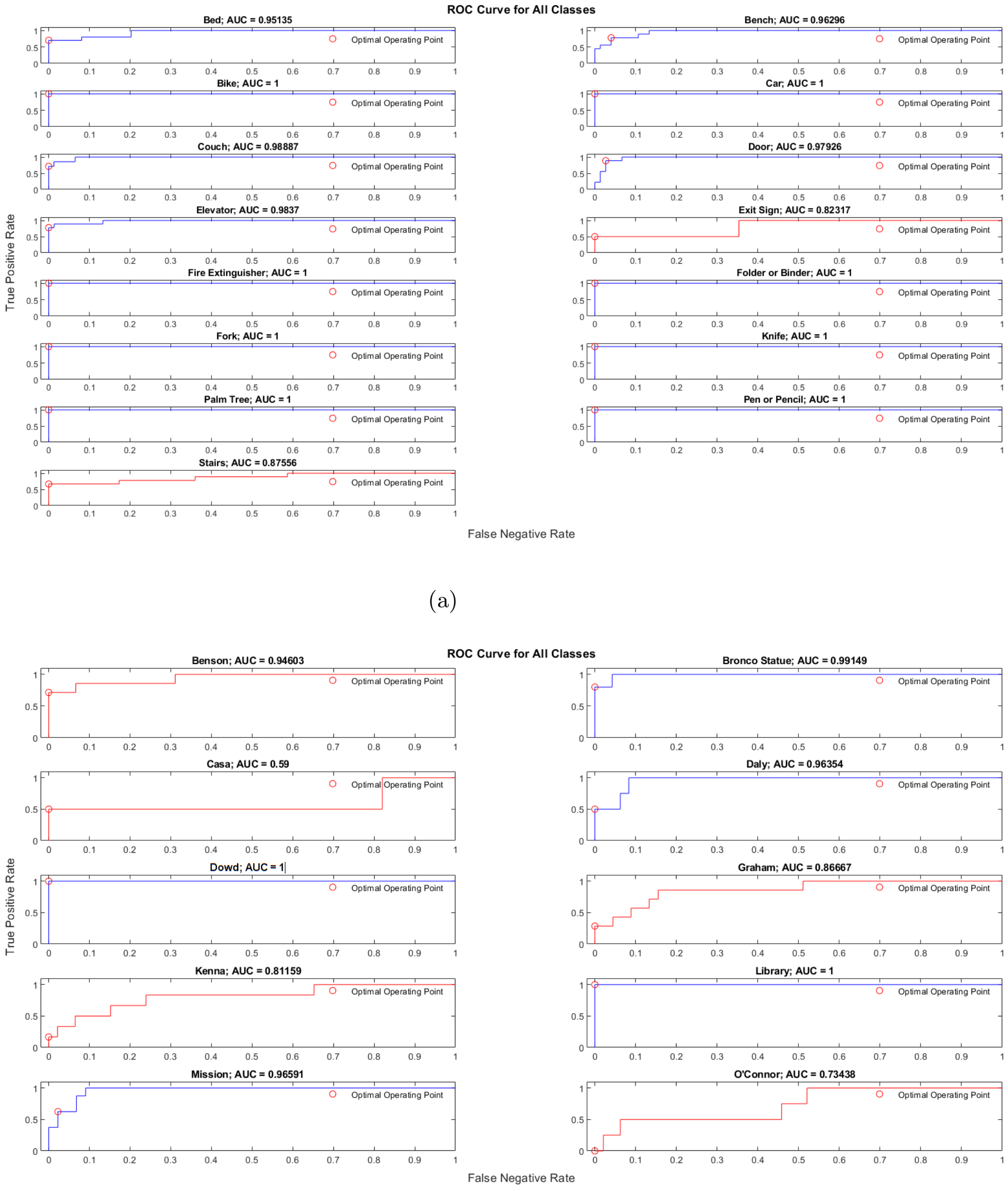
Figure 5: Confusion matrix for Validation data for (a) Group 1 Dataset and (b) Group 2 Dataset



(a)

(b)

Figure 6: Recall, Precision, and F1 Scores for (a) Group 1 Dataset and (b) Group 2 Dataset



(a)

(b)

Figure 7: Recall, Precision, and F1 Scores for (a) Group 1 Dataset and (b) Group 2 Dataset

5.2 Final Results

For generating the final results, I compiled more images for all image classes in both datasets. For the Group 2 dataset, I decided to swap out some of the Santa Clara University buildings, as it was fairly difficult to either find images online of the buildings online or take good quality pictures of them without different obstacle getting in the way. The updated image class are shown in Table 2. After collecting more data, Group 1 now has 399 images, and Group 2 now has 284 images.

Old Classes	New Classes
Benson	Benson Center
Bronco Statue	Bronco Statue
Casa	Dowd
Daly Science	Graham
Dowd	Library
Graham	**Lucas Hall**
Kenna	Mission Church
Library	O'Connor
Mission Church	**SCDI**
O'Connor	**Vari Hall**

Table 2: New Image Classes for Group 2 Dataset

For the sake of brevity, in showing my final results, I will be showing the graphical results only of the CNN architectures that performed the best for each dataset with data augmentation. This would be GoogleNet for Group 1 and ResNet50 for Group 2. Afterwards, I'll show a table comparing the accuracy and loss for all 4 CNNs, for both datasets, and both with and without data augmentation in a final table. The graphical results for all test cases are shown in my final presentation slides. GoogleNet trained for a total of 15 epochs with an initial learning rate of 0.00005. ResNet50 trained for 13 epochs with and initial learn rate of 0.0001. The data augmentation used is the same as in the baseline results.

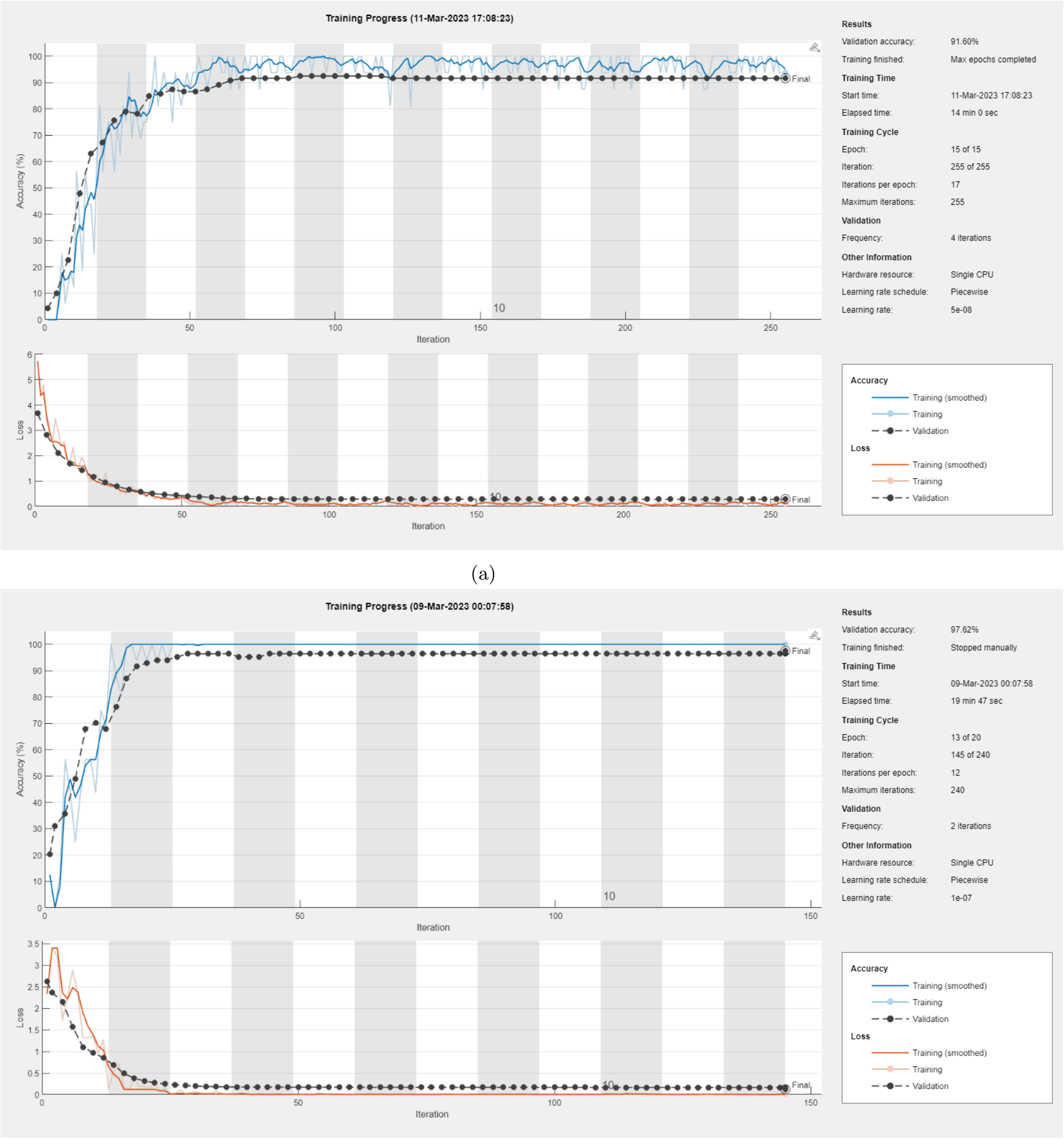
The final accuracy and loss plots for each dataset are shown in Figure 8. You can see there is a clear improvement in accuracy compared to the baseline results. For both groups, the final validation accuracy much more closely matches the final training accuracy, indicating less overfitting and better generalization to the validation data. The same can be said for the loss as well. For Group 1, you can see that it takes around 4 epochs for the validation accuracy to get to its highest value, and the training accuracy clearly oscillates throughout the training time. However, for Group 2, the validation accuracy only takes about 2 epochs to reach its highest value, and the training accuracy is much smoother, and consistently achieves 100% for most of the training time. This could be due to the slightly faster initial learn rate, or that the ResNet50 architecture is better able to handle vanishing/exploding gradients and converge to the minimum of the cost function at a faster rate.

The final confusion matrices are shown in Figure 9. You can see there are far fewer mispredictions for the validation data compared to the baseline results. You can see in Figure 9a that the Bed category (Class 1) performs somewhat poorly. There were images classified as Bed which belonged to other classes, as well as images whose true class was Bed that were classified as other classes. There were 2 cases where images of a Couch (Class 5) were classified as a Bed, and one case where and image of a Bed was classified as a Couch, indicating that GoogleNet had some trouble differentiating between those two classes. In Figure 9b, you can see that all but 2 classes were predicted perfectly. The only mispredictions were 2 cases where an image of Vari Hall (Class 10) was classified as being a image of SCDI (Class 9). Even though ResNet50 had some trouble separating those two classes, the overall results are excellent.

The Final Recall, Precision, and F1 Score Metrics are shown in Figure 10. This data reinforces what was seen in the confusion matrices. In Figure 10a, you can see that the Bed image class was the worst performing image class, with a few other classes performing less than optimally, but for the most part, all other image class have great results for both recall and precision. Looking at Figure 10b, with the exception of SCDI and Vari giving somewhat poor results, all other image classes have perfect recall and precision.

The final ROC Curves are shown in Figure 11. For both Group 1 and Group 2, you can see that all ROC curves for each class has a very high area under the curve, with none of them falling below 0.95. All optimal operating points, with the exception of the Vari Hall class in the Group 2 results, are either at the top left or very close to the top left of the graph, which is the best performance you can get. Also, in the Group 2 dataset, all curves except for Vari Hall have an area under the curve of 1.

A comparison of the validation accuracy and loss for all test cases can be seen in Table 3. Here we can clearly see the contributions that more data, different CNN architectures, and data augmentation has on system performance. For both Table 3a and Table 3a, you can see that going from the initial results to the final results without data augmentation, there is a clear increase in accuracy and decrease in loss. This proves that providing more data allowed the neural network to better approximate the mapping of images to class labels. Furthermore, comparing the performance of all CNN architectures, you can see that VGG-16, GoogleNet, and ResNet50 outperform AlexNet for all comparable test cases, showing that newer, more advanced architectures contribute to improved performance. Finally, comparing the final results with and without data augmentation, it's clear that using data augmentation



Training Progress (11-Mar-2023 17:08:23)

Results

Validation accuracy: 91.60%

Training finished: Max epochs completed

Training Time

Start time: 11-Mar-2023 17:08:23

Elapsed time: 14 min 0 sec

Training Cycle

Epoch: 15 of 15

Iteration: 255 of 255

Iterations per epoch: 17

Maximum iterations: 255

Validation

Frequency: 4 iterations

Other Information

Hardware resource: Single CPU

Learning rate schedule: Piecewise

Learning rate: 5e-08

(a)

Training Progress (09-Mar-2023 00:07:58)

Results

Validation accuracy: 97.62%

Training finished: Stopped manually

Training Time

Start time: 09-Mar-2023 00:07:58

Elapsed time: 19 min 47 sec

Training Cycle

Epoch: 13 of 20

Iteration: 145 of 240

Iterations per epoch: 12

Maximum iterations: 240

Validation

Frequency: 2 iterations

Other Information

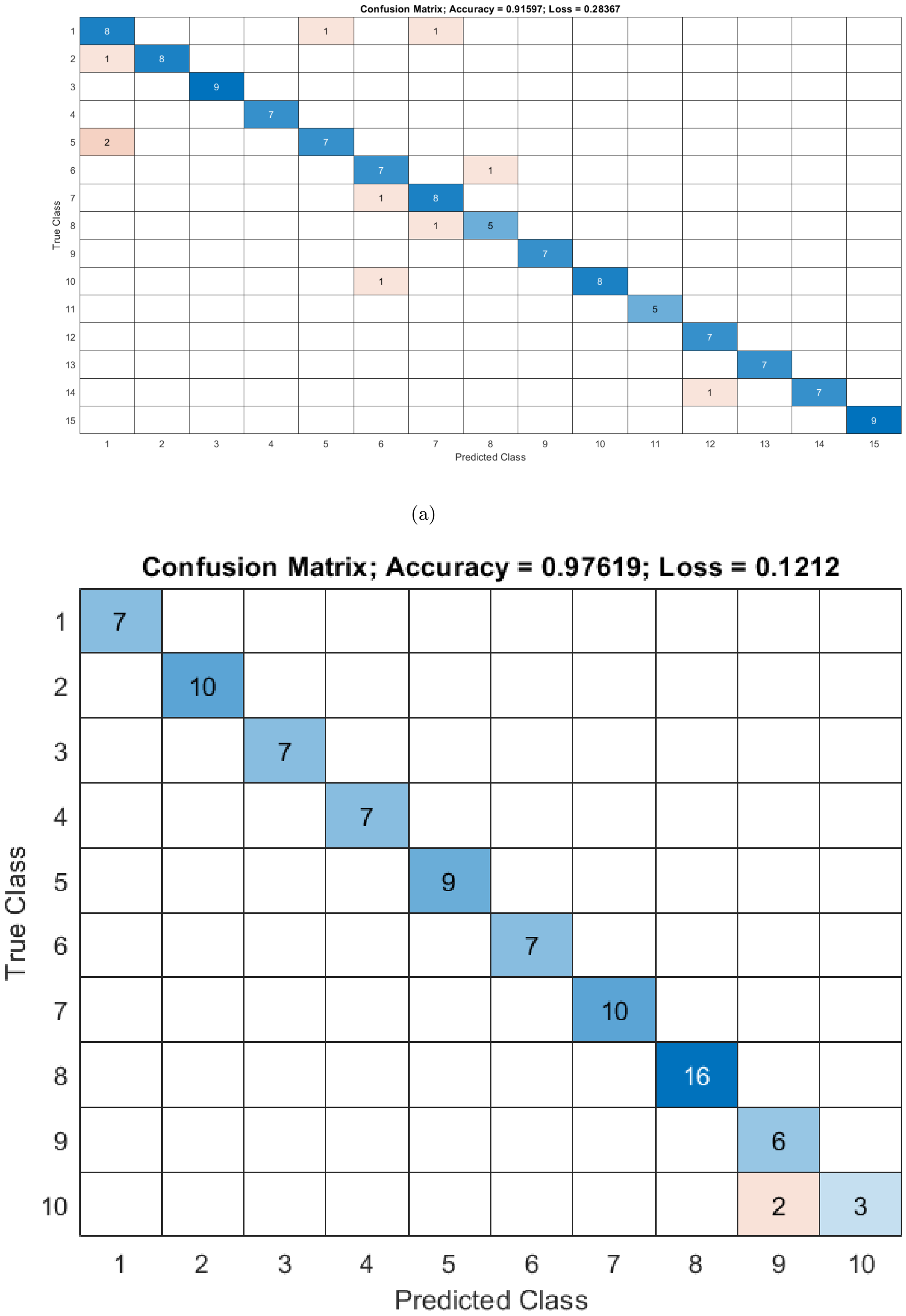
Hardware resource: Single CPU

Learning rate schedule: Piecewise

Learning rate: 1e-07

(b)

Figure 8: Final Accuracy and cross-entropy loss for (a) GoogleNet using Group 1 Dataset and (b) ResNet50 for Group 2 Dataset



Confusion Matrix; Accuracy = 0.91597; Loss = 0.28367

True Class

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	8				1		1								
2	1	8													
3			9												
4				7											
5	2				7										
6						7		1							
7						1	8								
8							1	5							
9									7						
10						1				8					
11											5				
12												7			
13													7		
14												1		7	
15															9

Predicted Class

(a)

Confusion Matrix; Accuracy = 0.97619; Loss = 0.1212

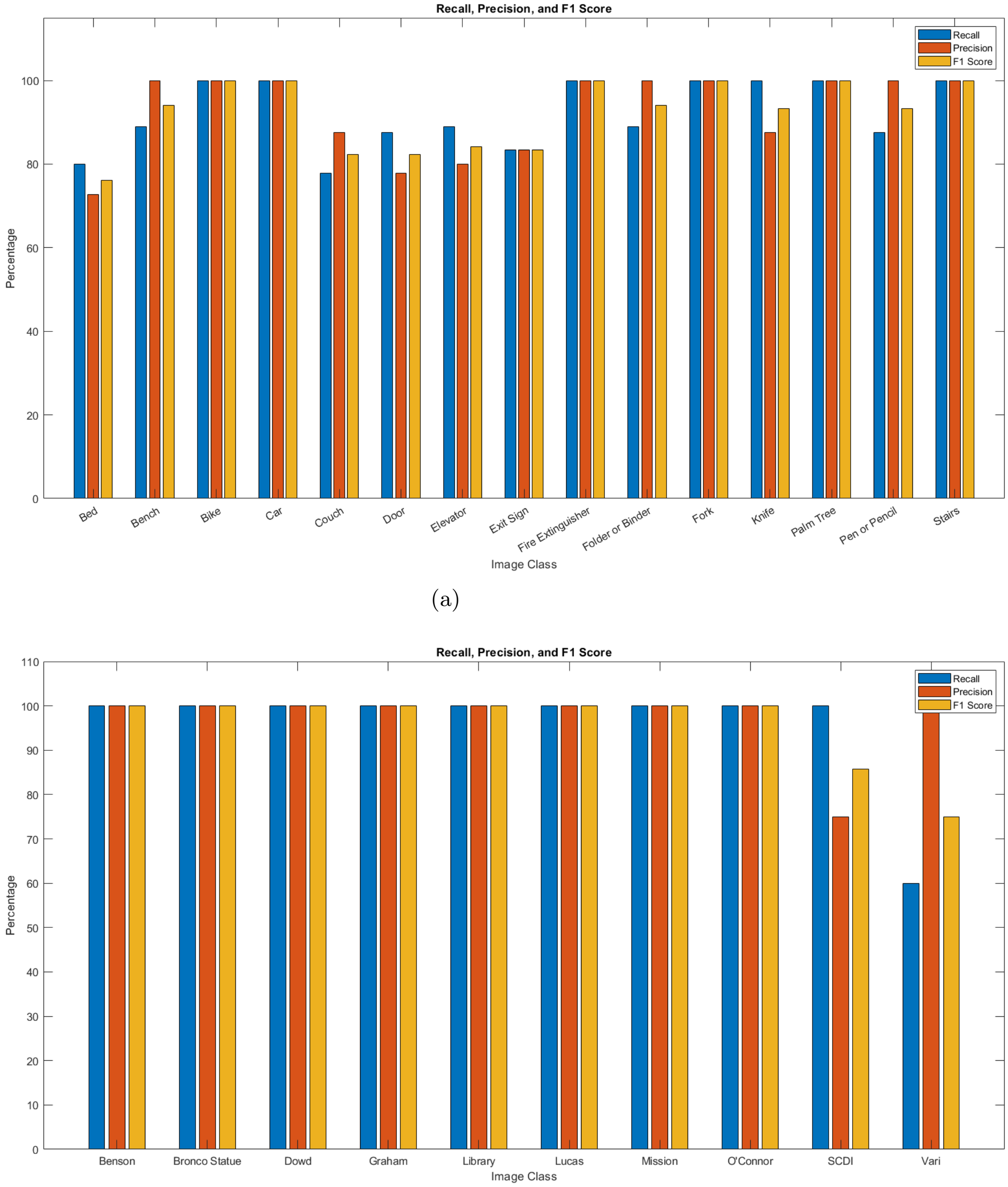
True Class

	1	2	3	4	5	6	7	8	9	10
1	7									
2		10								
3			7							
4				7						
5					9					
6						7				
7							10			
8								16		
9									6	
10									2	3

Predicted Class

(b)

Figure 9: Final Confusion Matrices for (a) GoogleNet using Group 1 Dataset and (b) ResNet50 using Group 2 Dataset



(a)

(b)

Figure 10: Final Recall, Precision, and F1 Score metrics for (a) GoogleNet using Group 1 Dataset and (b) ResNet50 using Group 2 Dataset

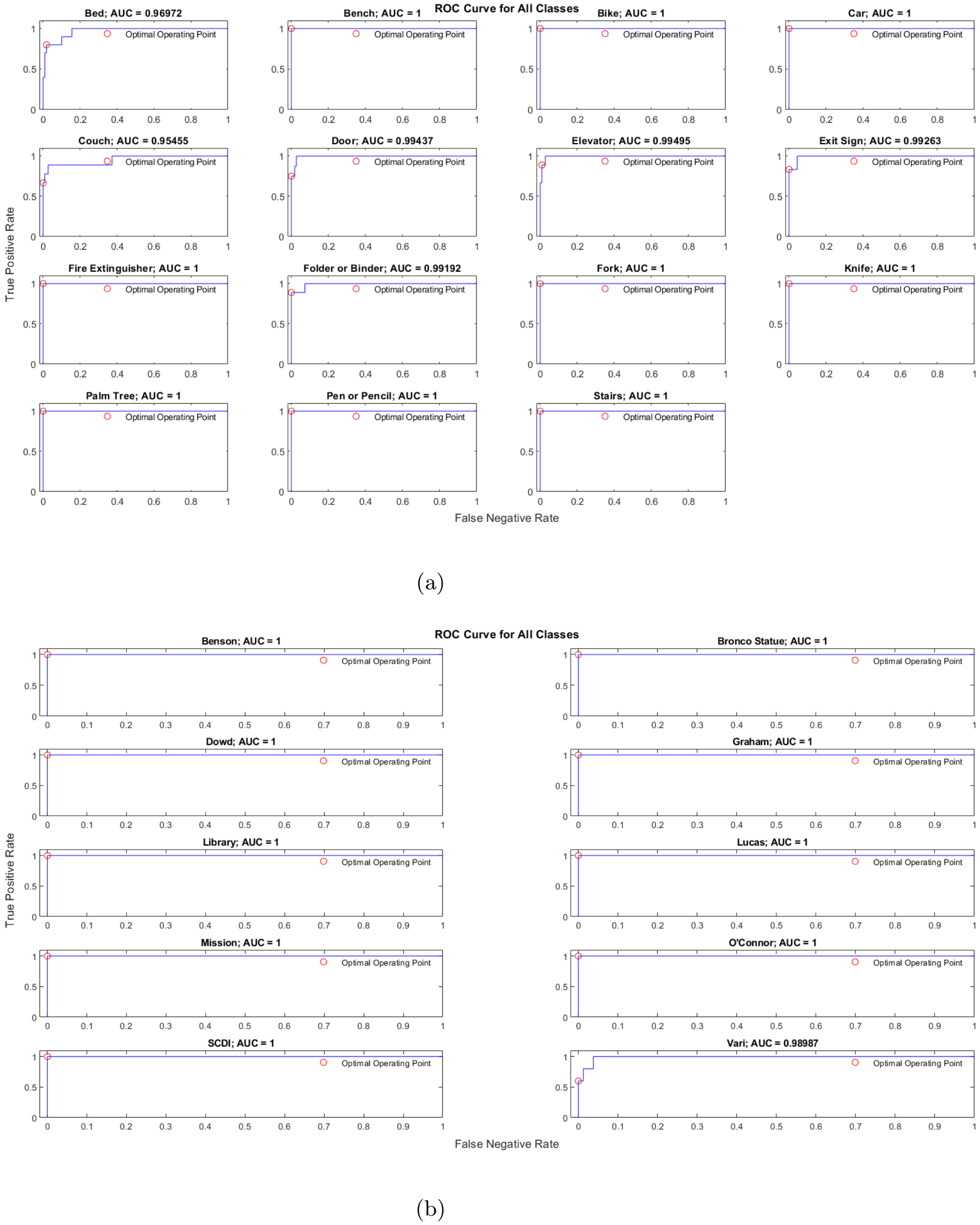


Figure 11: Final ROC Curves for each class for (a) GoogleNet using Group 1 Dataset and (b) ResNet50 using Group 2 Dataset

during training leads to increases accuracy, as the system is able to better avoid overfitting the training data and more optimally classify the validation data.

Another interesting observation is that these results were obtained when each dataset contained only a little more than 100 extra images compared to the baseline results. This demonstrates the value of having high-quality data to help the CNN learn the relevant features in each images class, as well as the value of transfer learning, where only a few hundred images are needed for good results. The quality of data is one reason I believe the Group 2 dataset ended up giving better final results than the Group 1 dataset when the opposite was true for the baseline case.

Initial Results				
	AlexNet	VGG-16	GoogleNet	ResNet50
Accuracy (%)	83.33	-	-	-
Loss	0.6094	-	-	-
Final Results without Data Augmentation				
Accuracy (%)	84.75	85.60	86.00	89.83
Loss	0.4692	0.3683	0.3765	0.4219
Final Results with Data Augmentation				
Accuracy (%)	88.03	88.98	91.60	90.76
Loss	0.4647	0.4005	0.2837	0.4091

(a) Group 1

Initial Results				
	AlexNet	VGG-16	GoogleNet	ResNet50
Accuracy (%)	65.39	-	-	-
Loss	1.0004	-	-	-
Final Results without Data Augmentation				
Accuracy (%)	83.13	86.75	80.72	83.13
Loss	0.7862	0.6354	0.698	0.6985
Final Results with Data Augmentation				
Accuracy (%)	85.54	90.48	94.05	97.62
Loss	0.5854	0.4330	0.4349	0.1212

(b) Group 2

Table 3: Final Validation Accuracy and Cross-entroy Loss Results. These tables show the baseline results, final results with new data but no data augmentation, and final results with new data and data augmentaion for (a) Group 1 and (b) Group 2 dataset. The baseline results from the original work were only obtained using AlexNet, so other CNNs architectures have no baseline results

5.3 Further Experimentation with Data Augmentation

For this experiment, a more deliberate demonstration of the value of data augmentation is shown. Using AlexNet with my updated Group 1 dataset, I tested the behavior of the validation accuracy for different types of data augmentation. This should give a sense of the types of data augmentation that work the best for this given dataset. The five test case used are given in Table 4, along with their final validation accuracies. Figure 12 shows how the validation accuracies changed over all iterations of training.

Case	Description	Final Accuracy (%)
1	- Base Case: No augmentation	84.75
2	- Random horizontal and vertical reflections	83.90
3	- Random horizontal and vertical reflections - Random rotations up to 20 degrees	87.29
4	- Random horizontal and vertical translations up to 50 pixels	87.29
5	- Random horizontal and vertical shearing - Random scaling between 0.5 and 2	78.81

Table 4: Data Augmentaion Test Cases and Final Validation Accuracy

An important observation to note is that not all cases of the data augmentation give better performance than the base case with no augmentation. This is because when performing data augmentation on a dataset, it's important to consider if the transformations being applied is still similar to real life data. If that is the case, you will get increased performance and avoid overfitting. However, if you augmented data is not similar to real life data, it can actually lead to underfitting and lead to worse performance. Looking at the graph, you can see that Cases 3 and 4 provide better validation accuracy with data augmentation, but cases 2 and 5 give similar or worse performance than than if you didn't use data augmentation, indicating the those transformations don't match real-world data that much.

5.4 Training Times

Finally, we look at the approximate training times for each CNN architecture and discuss the implications for real-time systems. The training time per epoch as well as the number of parameters for each CNN architecture is shown in Table 5.

	Training Time/Epoch (s)	# of parameters (millions)
AlexNet	124	62.3
VGG-16	160	138
GoogleNet	56	6.7977
ResNet50	83	23

Table 5: Approximate Training Times

The specific training times themselves don't really matter hear. There are many things that can affect training times, such as number of epochs, how often to test the validation data, learning rate, batch size, among others. The interesting observation here is that there is a clear correlations between

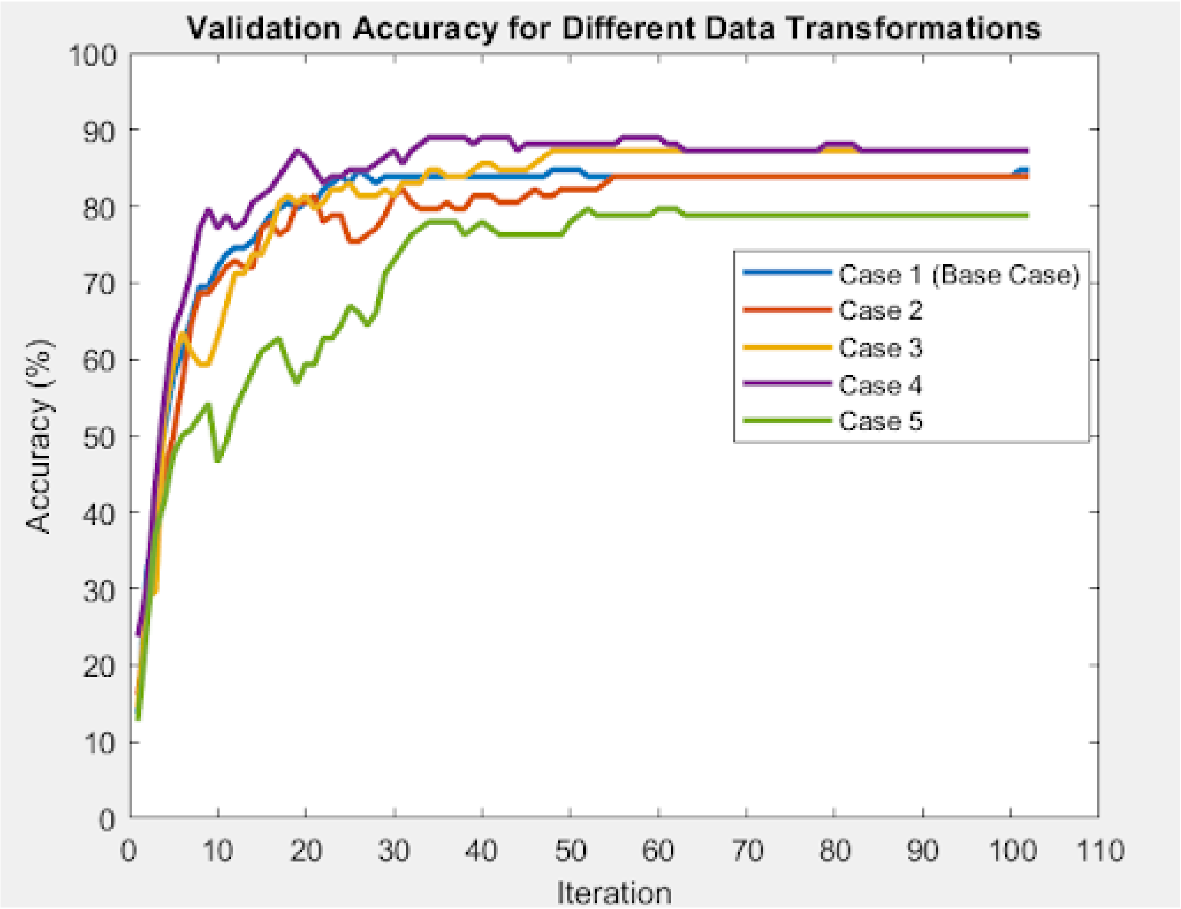


Figure 12: Validation Accuracy for Data Augmentation cases

training time and the size of the CNN architecture. This is an important consideration if you were to develop this application into a real system. Many times there will be a trade-off between network performance and network size. For example, the Group 2 dataset achieved the best results with ResNet50. However, if we were to integrate this into a system where inference time is a consideration, you might want to take a small hit to accuracy and opt for a smaller network like GoogleNet.

6 Conclusion and Future Work

This work has shown what features are important to improving performance for image recognition problems. First, we saw that by increasing the amount of high-quality data used to train a neural network, the neural network can better learn the relevant features of the data and give better prediction accuracy. Secondly, by using new CNN architectures that were made to handle drawbacks in simpler architecture like vanishing gradients and scale invariance, we can achieve faster convergence to the minimum of the cost function and get better performance. Lastly, we saw that data augmentation has a significant effect on performance. If done rightly, it can avoid overfitting a generalize well to test data, but if done in a way where the augmented data is inconsistent with real-world data, can actually hurt performance.

For future work, I would like to train these CNN architectures with one dataset, combining Groups 1 and 2, to see the effect it has on performance. This is a logical next step, as if this were to be used in a real system, it would be very beneficial to use one CNN for predicting all image classes instead of using two.

References

[1] GBD 2019 Blindness and Vision Impairment Collaborators; Vision Loss Expert Group of the Global Burden of Disease Study. Trends in prevalence of blindness and distance and near vision impairment over 30 years: an analysis for the Global Burden of Disease Study. Lancet Glob Health. 2021 Feb;9(2):e130-e143. doi: 10.1016/S2214-109X(20)30425-3. Epub 2020 Dec 1. PMID: 33275950; PMCID: PMC7820390.

[2] A. Shelton and T. Ogunfunmi, "Developing a Deep Learning-enabled Guide for the Visually Impaired," 2020 IEEE Global Humanitarian Technology Conference (GHTC), Seattle, WA, USA, 2020, pp. 1-8, doi: 10.1109/GHTC46280.2020.9342873.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural

Information Processing Systems - Volume 1 (NIPS'12). Curran Associates Inc., Red Hook, NY, USA, 1097–1105.

[4] Simonyan, K. & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. CoRR, abs/1409.1556.

[5] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

[6] He, Kaiming et al. "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 770-778.

[7] A. Kariluoto, J. Kultanen, J. Soininen, A. Pärnänen and P. Abrahamsson, "Quality of Data in Machine Learning," 2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C), Hainan, China, 2021, pp. 216-221, doi: 10.1109/QRS-C55045.2021.00040.

[8] Ganesan, Jothi & Azar, Ahmad & Alsenan, Shrooq & Ahmad Kamal, Nashwa & Qureshi, Basit & Hassanien, Aboul. (2022). Deep Learning Reader for Visually Impaired. Electronics. 11. 3335. 10.3390/electronics11203335.

[9] H. -Y. Tsai, H. Zhang, C. -L. Hung and F. -R. Hsu, "A Deep Learning Models for Blind Guidance by Integrating CNN and ELM," 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, UK, 2018, pp. 1229-1234, doi: 10.1109/HPCC/SmartCity/DSS.2018.00207.

[10] D. Al-Rubaye and S. Ayvaz, "Deep Transfer Learning and Data Augmentation for Food Image Classification," 2022 Iraqi International Conference on Communication and Information Technologies (IICCIT), Basrah, Iraq, 2022, pp. 125-130, doi: 10.1109/IICCIT55816.2022.10010432.

[11] S. Chiba and H. Sasaoka, "Basic Study for Transfer Learning for Autonomous Driving in Car Race of Model Car," 2021 6th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand, 2021, pp. 138-141, doi: 10.1109/ICBIR52339.2021.9465856.

[12] C. Li et al., "Data Augmentation for Inertial Sensor Data in CNNs for Cattle Behavior Classification," in IEEE Sensors Letters, vol. 5, no. 11, pp. 1-4, Nov. 2021, Art no. 7003104, doi: 10.1109/LSENS.2021.3119056.

[13] M. Shiotani, S. Iguchi and K. Yamaguchi, ”Research on data augmentation for vital data using conditional GAN,” 2022 IEEE 11th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 2022, pp. 344-345, doi: 10.1109/GCCE56475.2022.10014132.

[14] J. Kim, S. Kim, T. Lee, Y. Lim and J. Lim, ”Smart Glasses using Deep Learning and Stereo Camera,” 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 2019, pp. 294-295.

[15] S. AbuSalim, N. Zakaria, N. Mokhtar, S. A. Mostafa and S. J. Abdulkadir, ”Data Augmentation on Intra-Oral Images Using Image Manipulation Techniques,” 2022 International Conference on Digital Transformation and Intelligence (ICDI), Kuching, Sarawak, Malaysia, 2022, pp. 117-120, doi: 10.1109/ICDI57181.2022.10007158.

[16] Gandhi, A. (n.d.). Data Augmentation - How to use Deep Learning when you have Limited Data - Part 2. Nanonets. Retrieved February 20, 2023, from https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

[17] Russakovsky, O., Deng, J., Su, H. et al. ImageNet Large Scale Visual Recognition Challenge. Int J Comput Vis 115, 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y

[18] A. A. Almisreb, N. Jamil and N. M. Din, ”Utilizing AlexNet Deep Transfer Learning for Ear Recognition,” 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), Kota Kinabalu, Malaysia, 2018, pp. 1-5, doi: 10.1109/INFRKM.2018.8464769.

[19] P. S. Kumar and S. Pranavi, ”Performance analysis of machine learning algorithms on diabetes dataset using big data analytics,” 2017 International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), Dubai, United Arab Emirates, 2017, pp. 508-513, doi: 10.1109/ICTUS.2017.8286062.