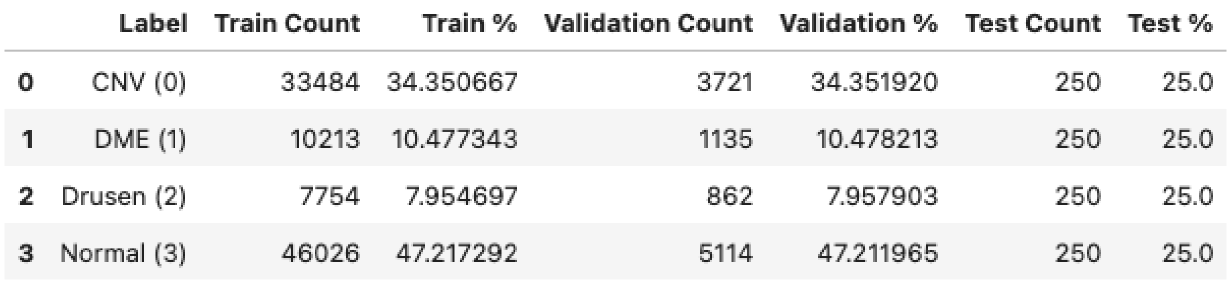
Topic : OCT MNIST Classification

Data set overview:



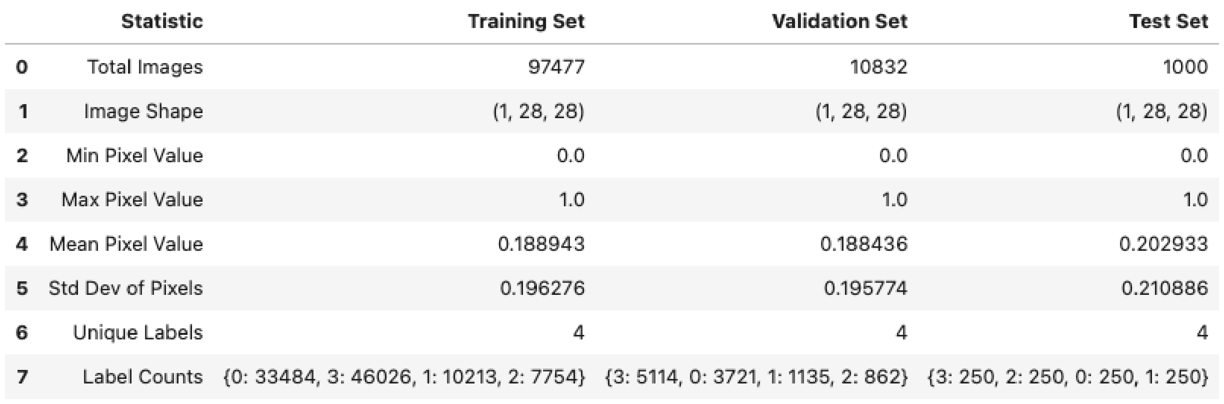
	Label	Train Count	Train %	Validation Count	Validation %	Test Count	Test %
0	CNV (0)	33484	34.350667	3721	34.351920	250	25.0
1	DME (1)	10213	10.477343	1135	10.478213	250	25.0
2	Drusen (2)	7754	7.954697	862	7.957903	250	25.0
3	Normal (3)	46026	47.217292	5114	47.211965	250	25.0

Insights from the OCT MNIST Label Distribution:

Training Set : The dataset is highly imbalanced, with the Normal Retina (Label 3) having the highest number of samples. CNV (Label 0) also has a large number of samples but is less than the normal retina class. DME (Label 1) and Drusen (Label 2) have significantly fewer samples compared to CNV and Normal Retina, indicating an imbalance that may affect model performance. Validation Set:

Similar imbalance is observed in the validation set: Normal Retina (Label 3) and CNV (Label 0) dominate. DME (Label 1) and Drusen (Label 2) have lower representation, mirroring the training set. This consistency indicates that the model's performance on validation will likely reflect similar biases as the training. Test Set:

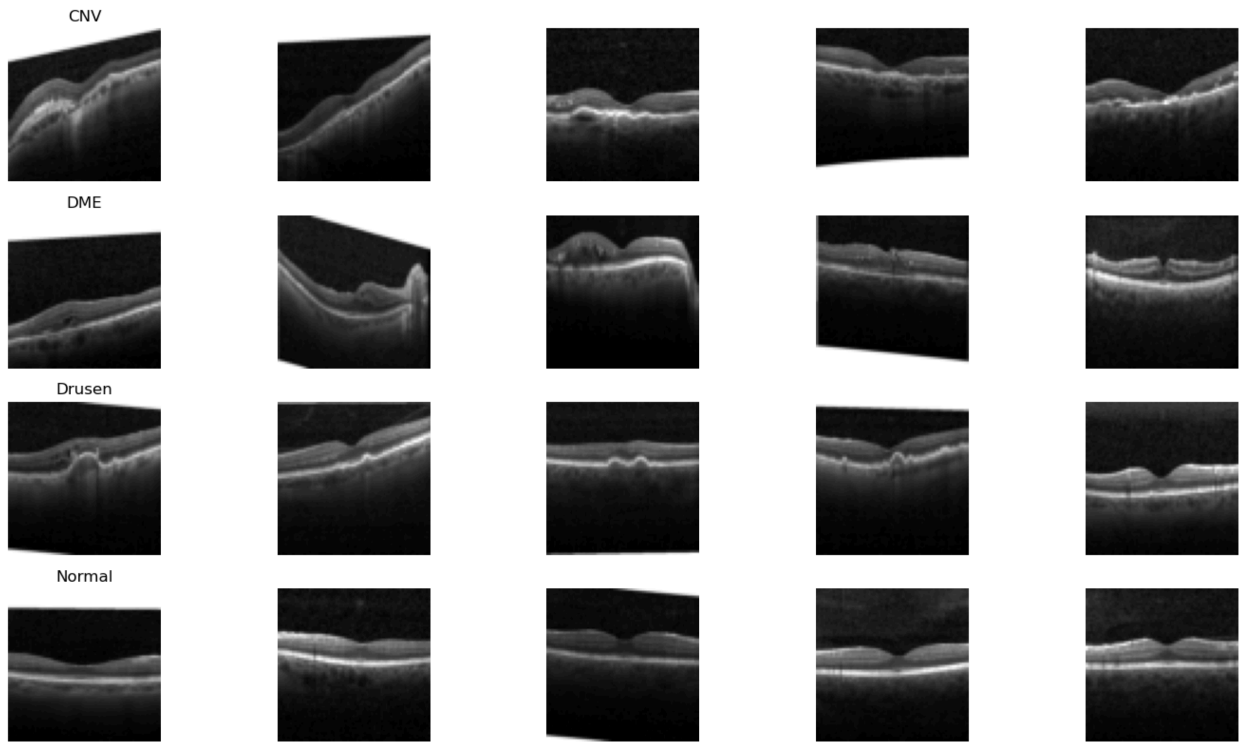
The test set is perfectly balanced with an equal number of samples for each category. This balanced test distribution ensures that the final evaluation metric will not be biased toward any one label, providing an unbiased assessment of model performance.



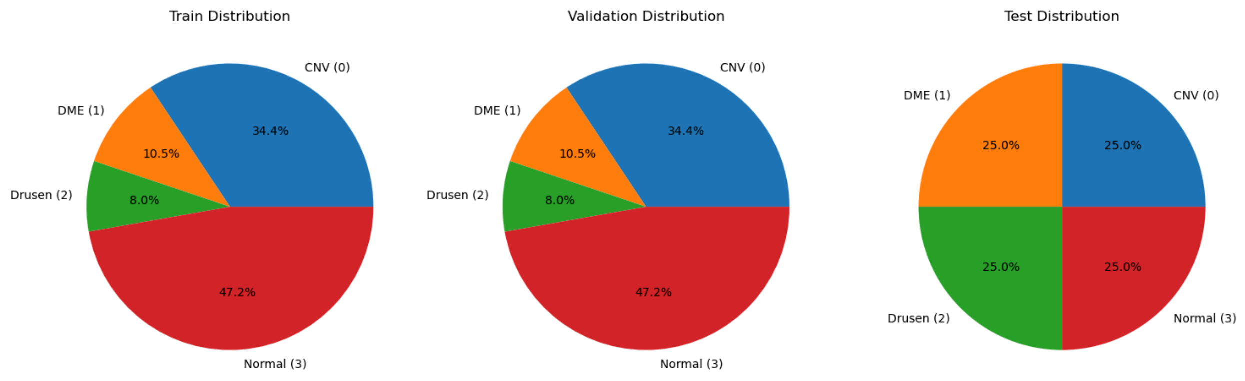
	Statistic	Training Set	Validation Set	Test Set
0	Total Images	97477	10832	1000
1	Image Shape	(1, 28, 28)	(1, 28, 28)	(1, 28, 28)
2	Min Pixel Value	0.0	0.0	0.0
3	Max Pixel Value	1.0	1.0	1.0
4	Mean Pixel Value	0.188943	0.188436	0.202933
5	Std Dev of Pixels	0.196276	0.195774	0.210886
6	Unique Labels	4	4	4
7	Label Counts	{0: 33484, 3: 46026, 1: 10213, 2: 7754}	{3: 5114, 0: 3721, 1: 1135, 2: 862}	{3: 250, 2: 250, 0: 250, 1: 250}

Graphical Representation

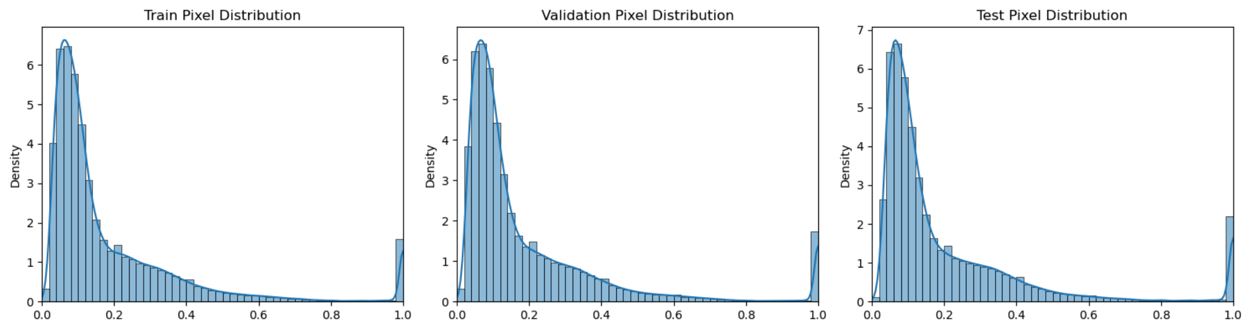
1.*Image Samples Grid*: Displays randomly sampled images per class to visually inspect data quality and annotation consistency.



2. *Label Distribution*: Visualizes class proportions across splits using pie charts to highlight imbalances.

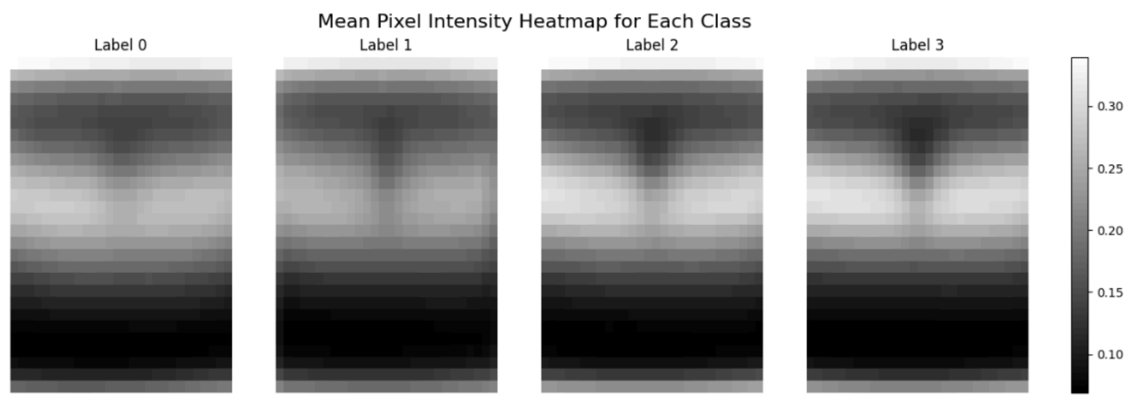


3. *Pixel Distribution*: Compares intensity statistics (mean, std dev) via box plots/histograms to detect normalization shifts.



4. *Mean Pixel Intensity* : Used to find any unique properties for different classes.Each heatmap represents the average pixel intensity of images from a specific label, showing common patterns. Brighter regions indicate higher pixel intensity, which could correspond to denser or abnormal structures, while darker regions represent less dense areas or backgrounds.
Class-wise Analysis:

Label 0 (CNV - Choroidal Neovascularization):More variation in the upper-middle region, suggesting abnormal blood vessel growth near this area.
Label 1 (DME - Diabetic Macular Edema):Central bright region indicates fluid accumulation in the macula, which is a characteristic of DME.
Label 2 (Drusen):Slightly more uniform but shows bright spots that could correspond to lipid deposits under the retina.
Label 3 (Normal Retina):The heatmap appears smoother and more uniform, with no distinct bright spots or irregularities, representing a healthy retina.



Neural Network Architecture (OctMNISTModel)

A custom convolutional neural network (CNN) designed for 64×64 grayscale image classification on 4 classes, implementing:

- 2 convolutional blocks with batch normalization and max-pooling
- 2 fully connected layers with dropout regularization
- PyTorch framework with CrossEntropyLoss compatibility

Layer by Layer Breakdown

1. Convolutional Block 1

Conv2d(1, 16, kernel_size=5) → BatchNorm2d(16) → ReLU → MaxPool2d(2) → Dropout(0.3)

- Input: 1×64×64 (Channel×Height×Width)
- Output: 16×30×30 (16 channels from 5×5 kernels, reduced by pooling)
- Parameters: 416 weights (16×(1×5×5)+16 BN params)

2. Convolutional Block 2

Conv2d(16, 32, kernel_size=5) → BatchNorm2d(32) → ReLU → MaxPool2d(2) → Dropout(0.3)

- Input: 16×30×30
- Output: 32×13×13
- Parameters: 12,832 weights (32×(16×5×5)+32 BN params)

3. Fully Connected Classifier

Linear(32×13×13 → 128) → ReLU → Dropout(0.5) → Linear(128 → 4)

- Flattened Features: 32×13×13 = 5,408 dimensions
- Final Output: 4-class logits
- Parameters: 692,100 (fc1) + 516 (fc2) = 692,616 weights

Key Design Features

1. Dimensionality Reduction: Sequential pooling (2×2 windows) progressively reduces spatial dimensions from 64→30→13
2. Regularization Strategy:
    - Progressive Dropout: 30% after conv layers, 50% in FC
    - Batch Normalization: After each convolution for stable training
3. Parameter Count:
   Total Trainable Parameters: ~706K (416 + 12,832 + 692,616)

Tensor Dimension Evolution

Layer	**Output Shape**	**Parameters**
Input	(1, 64, 64)	-
Conv1 + BN + Pool	(16, 30, 30)	416
Conv2 + BN + Pool	(32, 13, 13)	12,832
Flatten	5,408	-
FC1	128	692,100
FC2	4	516

<u>Techniques that impacted the model's performance:</u>

1. *Batch Normalization*
Location: Added after convolutional layers (self.bn1, self.bn2).
Impact:Stabilized training by normalizing layer inputs, reducing internal covariate shift.Accelerated convergence (evident from smooth training curves).
Improved generalization, as seen in higher validation/test accuracy compared to baseline models.

2. *Dropout*
Location:Convolutional dropout (self.dropout_conv1, self.dropout_conv2 with p=0.3).Fully connected dropout (self.dropout with p=0.5).
Impact:Reduced overfitting by randomly disabling neurons during training.
Evidence: Validation accuracy (92%) closely matches training accuracy (93%), indicating reduced overfitting.
Improved robustness, as seen in stable test performance (84% accuracy).

3. *L2 Regularization (Weight Decay)*
Location: optim.Adam(weight_decay=1e-4).
Impact:Penalized large weights to prevent overfitting.Improved generalization by simplifying the model.
Evidence: Validation loss plateaued early, indicating controlled model complexity.

4. *Learning Rate Scheduling*
Location: ReduceLROnPlateau(optimizer, patience=5, factor=0.5).
Impact:Dynamically reduced learning rate when validation loss plateaued.
Enabled finer parameter updates in later training stages.

Evidence: Gradual decrease in training/validation loss over epochs.

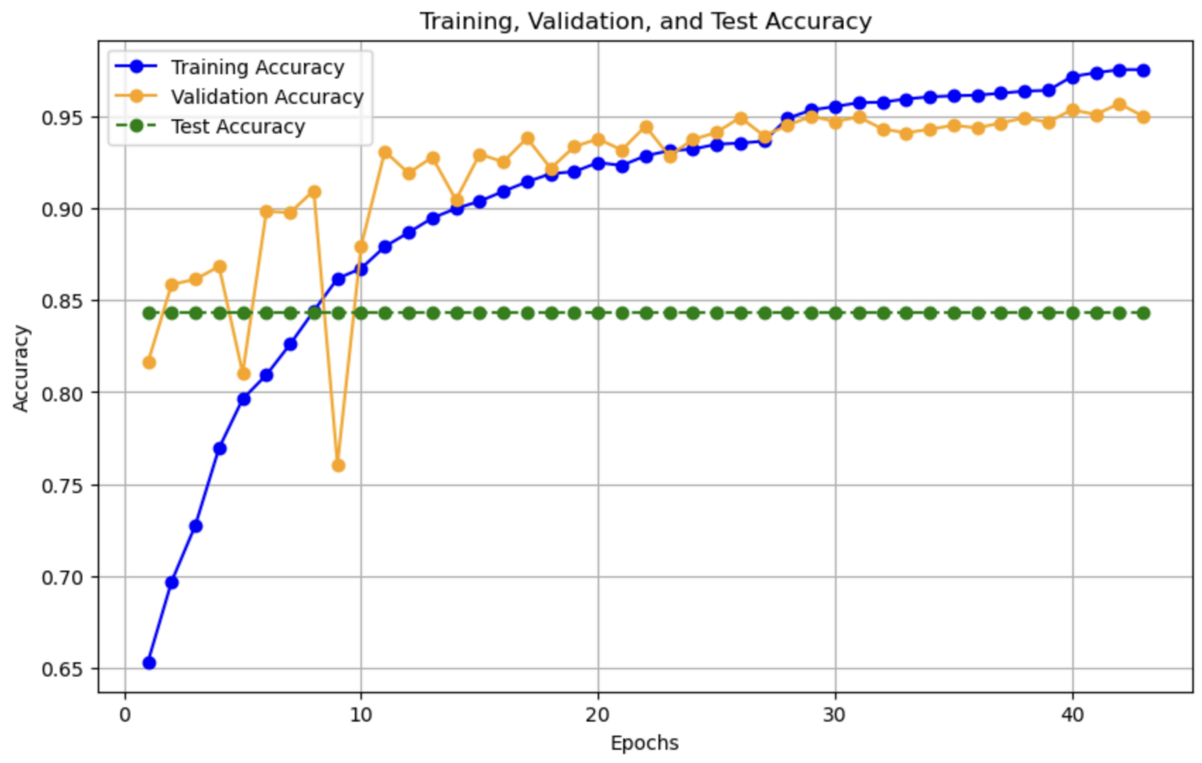
5. *Early Stopping*
Location: Monitored best_val_loss with patience=10.
Impact:Prevented overfitting by stopping training when validation loss stopped improving.
Evidence: Training stopped before validation loss diverged from training loss.

6. *Gradient Accumulation*
Location: accum_steps=4.
Impact:Simulated larger batch sizes despite memory constraints.
Stabilized parameter updates, leading to smoother convergence.

7. *Weighted Loss Function*
Location: nn.CrossEntropyLoss(weight=class_weights_tensor).
Impact:Addressed class imbalance by assigning higher weights to underrepresented classes.
Evidence: Improved recall for minority classes (e.g., Class 2 recall: 79% train, 80% val).

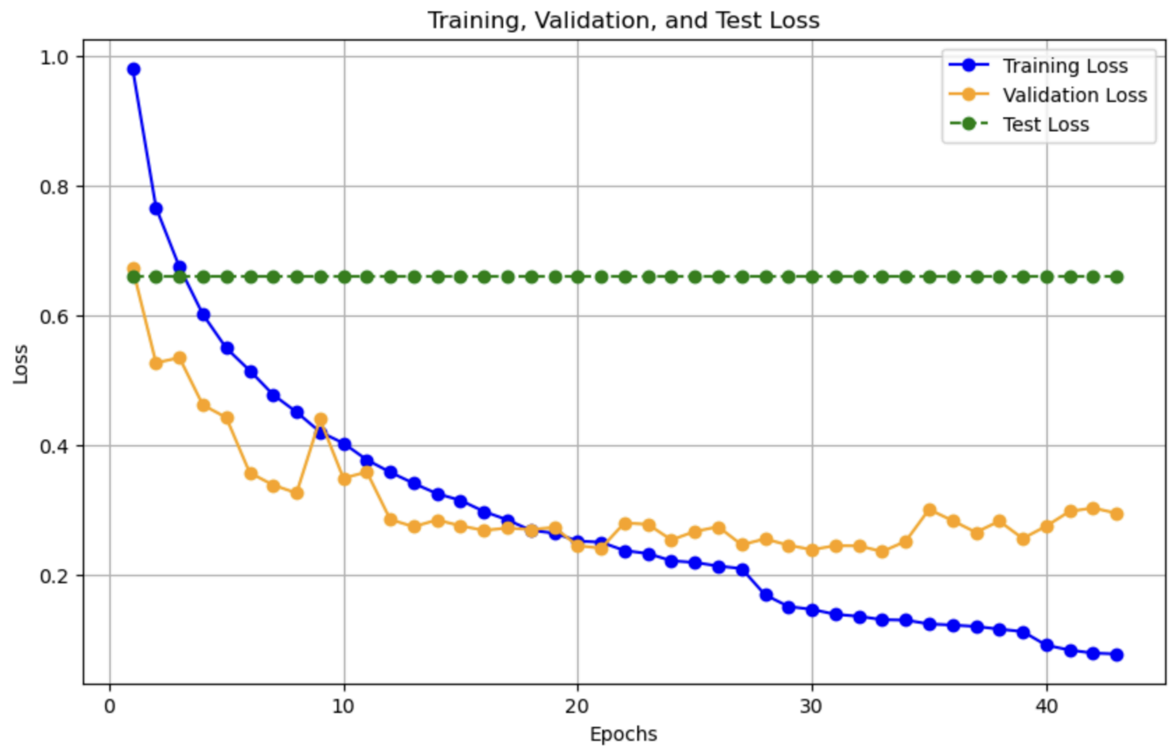
Results and relevant graphs

Training, Validation, and Test Accuracy
Training accuracy increases consistently with epochs, reaching over 95%, which shows good learning progress.
Validation accuracy fluctuates early but stabilizes around 90%, aligning closely with training accuracy after sufficient epochs.
Test accuracy remains constant at approximately 85%, indicating good but not perfect generalization to unseen data.



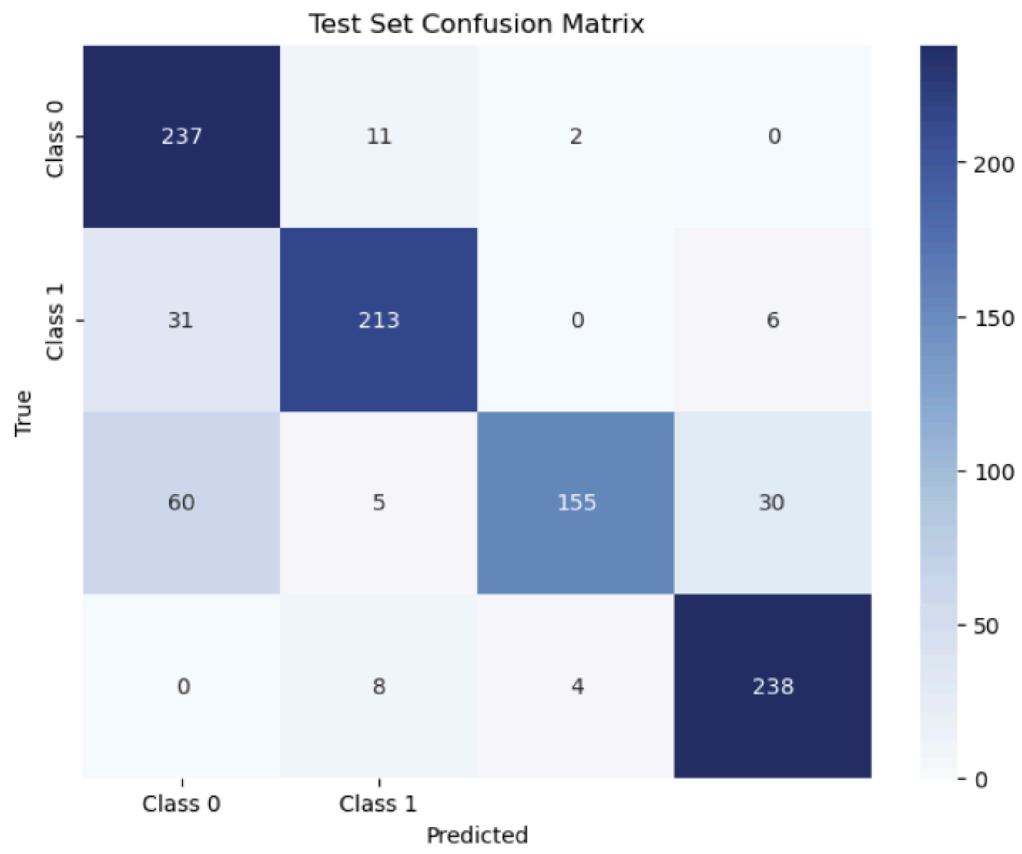
Training, Validation, and Test Loss
The training loss decreases steadily over epochs, indicating that the model is learning effectively.
Validation loss decreases initially but stabilizes after approximately 10 epochs, suggesting that the model generalizes well without significant overfitting.
Test loss remains constant throughout training, which could indicate that it was measured after training was completed or that it reflects a stable performance on

unseen data.



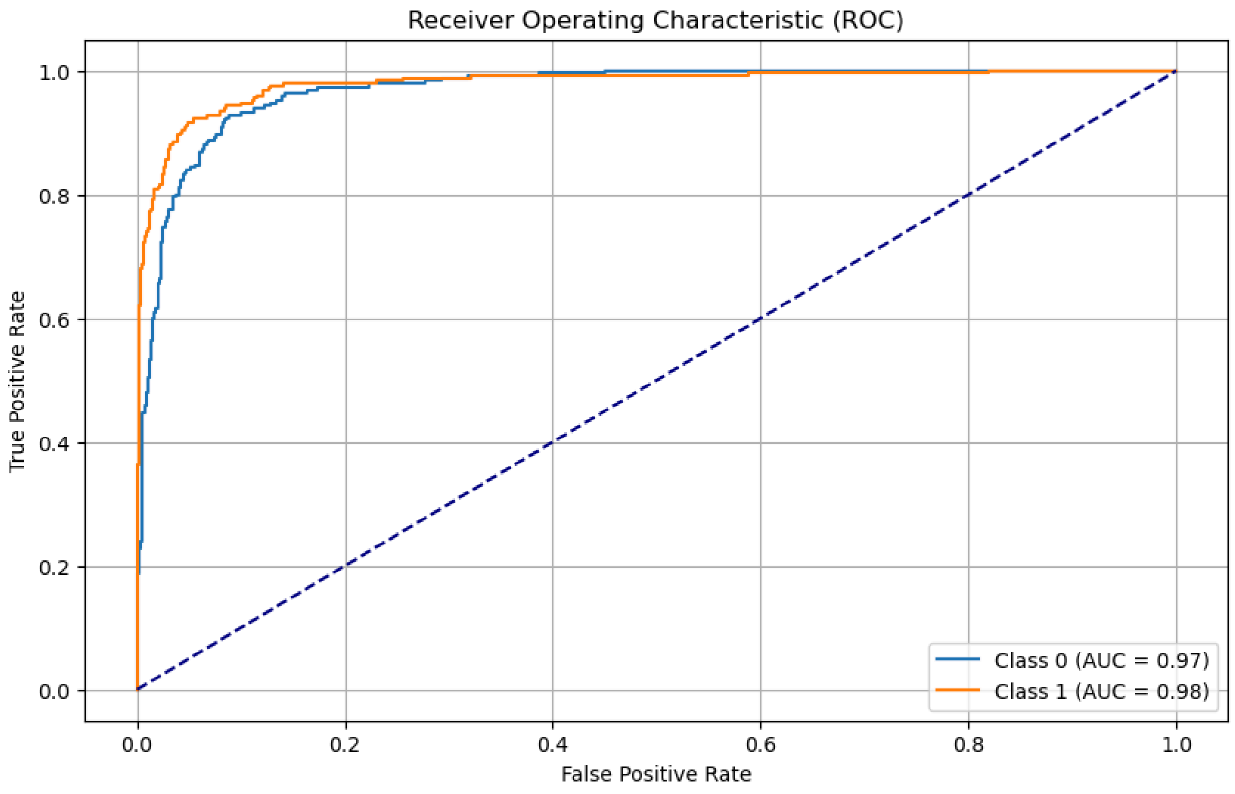
Confusion Matrix

The confusion matrix provides detailed insights into the model's classification performance on the test set:



Key Observations:
The model performs well in correctly classifying both classes, as evidenced by high diagonal values.
Misclassifications:
For Class 0: Only a small number of samples (11+2 = 13) were misclassified as Class 1.
For Class 1: A moderate number of samples (31+60 = 91) were misclassified as Class 0.
The confusion matrix highlights that while overall performance is good, there is room for improvement in reducing false negatives for Class 1.

Receiver Operating Characteristic (ROC) Curves
The ROC curves for both Class 0 and Class 1 demonstrate strong performance:
Class 0: Area Under the Curve (AUC) = 0.97
Class 1: Area Under the Curve (AUC) = 0.98
These high AUC values indicate that the model is highly capable of distinguishing between the two classes. The curves are close to the top-left corner, which reflects a high true positive rate (sensitivity) with a low false positive rate.

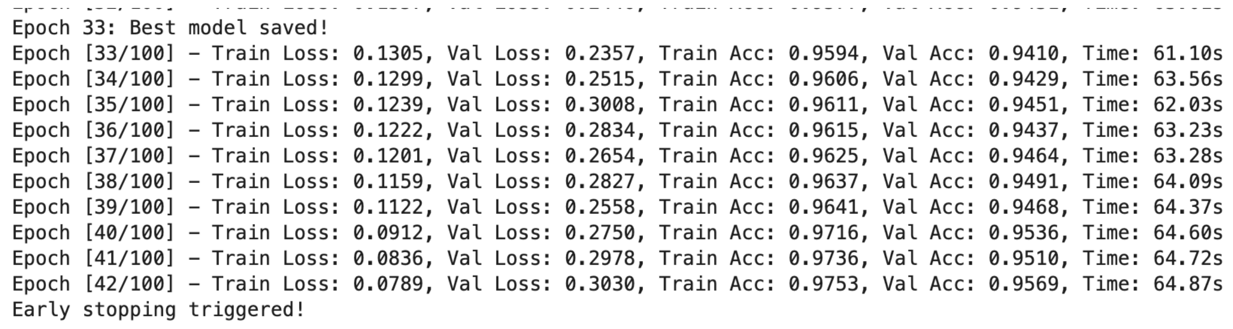


```
TRAIN CONFUSION MATRIX
              precision    recall  f1-score   support

           0       1.00      0.95      0.97     33484
           1       0.94      0.99      0.97     10213
           2       0.80      0.98      0.88      7754
           3       0.99      0.98      0.98     46026

    accuracy                           0.97     97477
   macro avg       0.93      0.97      0.95     97477
weighted avg       0.97      0.97      0.97     97477

========================================================
VALIDATION CONFUSION MATRIX
              precision    recall  f1-score   support

           0       0.98      0.93      0.95      3721
           1       0.89      0.94      0.92      1135
           2       0.69      0.88      0.78       862
           3       0.98      0.96      0.97      5114

    accuracy                           0.94     10832
   macro avg       0.89      0.93      0.90     10832
weighted avg       0.95      0.94      0.94     10832

========================================================

TEST CONFUSION MATRIX
              precision    recall  f1-score   support

           0       0.71      0.96      0.81       250
           1       0.89      0.90      0.90       250
           2       0.94      0.66      0.77       250
           3       0.96      0.90      0.93       250

    accuracy                           0.85      1000
   macro avg       0.88      0.85      0.85      1000
weighted avg       0.88      0.85      0.85      1000
```

<u>Methods that helped improve the accuracy and training time</u>

Early stopping was implemented to prevent overfitting and reduce unnecessary training time. The key steps included:
Validation Loss Monitoring: The validation loss was monitored at each epoch. If no improvement was observed for a specified number of epochs (patience), training was stopped.
Best Model Saving: The model's state was saved whenever the validation loss improved, ensuring the best-performing model was retained.
Patience Parameter: A patience value was set to allow the model some flexibility to improve after temporary plateaus in performance.
Benefits:Prevents overfitting by halting training when validation performance stagnates or deteriorates.
Reduces computational costs by avoiding unnecessary epochs.

As we can see, I just ran 42 epochs instead of 100 as the results were not varying. Thus the training time reduced significantly.
Monitored best_val_loss with patience=10:



```
Epoch 33: Best model saved!
Epoch [33/100] - Train Loss: 0.1305, Val Loss: 0.2357, Train Acc: 0.9594, Val Acc: 0.9410, Time: 61.10s
Epoch [34/100] - Train Loss: 0.1299, Val Loss: 0.2515, Train Acc: 0.9606, Val Acc: 0.9429, Time: 63.56s
Epoch [35/100] - Train Loss: 0.1239, Val Loss: 0.3008, Train Acc: 0.9611, Val Acc: 0.9451, Time: 62.03s
Epoch [36/100] - Train Loss: 0.1222, Val Loss: 0.2834, Train Acc: 0.9615, Val Acc: 0.9437, Time: 63.23s
Epoch [37/100] - Train Loss: 0.1201, Val Loss: 0.2654, Train Acc: 0.9625, Val Acc: 0.9464, Time: 63.28s
Epoch [38/100] - Train Loss: 0.1159, Val Loss: 0.2827, Train Acc: 0.9637, Val Acc: 0.9491, Time: 64.09s
Epoch [39/100] - Train Loss: 0.1122, Val Loss: 0.2558, Train Acc: 0.9641, Val Acc: 0.9468, Time: 64.37s
Epoch [40/100] - Train Loss: 0.0912, Val Loss: 0.2750, Train Acc: 0.9716, Val Acc: 0.9536, Time: 64.60s
Epoch [41/100] - Train Loss: 0.0836, Val Loss: 0.2978, Train Acc: 0.9736, Val Acc: 0.9510, Time: 64.72s
Epoch [42/100] - Train Loss: 0.0789, Val Loss: 0.3030, Train Acc: 0.9753, Val Acc: 0.9569, Time: 64.87s
Early stopping triggered!
```

Learning Rate Scheduling : It adjusted the learning rate based on validation loss. It ensured faster convergence during initial epochs when the learning rate was higher.
More precise fine-tuning in later epochs as the learning rate decreased.
Benefits:Improves optimization efficiency by dynamically adjusting the step size.Helps avoid overshooting or getting stuck in local minima.

L2 Regularization (Weight Decay):
I used optim.Adam(weight_decay=1e-4).
It penalized large weights to prevent overfitting.Improved generalization by simplifying the model.
Evidence: Validation loss plateaued early, indicating controlled model complexity.

<u>Detailed Description of best Model</u>

Model Architecture and Parameters

The model is a CNN designed for image classification tasks with the following layers and parameters:
Convolutional Layers:
Conv1: 16 filters, kernel size = 5, followed by Batch Normalization, ReLU activation, MaxPooling (kernel size = 2), and Dropout (p = 0.3).
Conv2: 32 filters, kernel size = 5, followed by Batch Normalization, ReLU activation, MaxPooling (kernel size = 2), and Dropout (p = 0.3).
Fully Connected Layers:
FC1: Input size = 32×13×13, output size = 128 neurons with ReLU activation and Dropout (p = 0.5).
FC2: Output layer with the number of classes set to 4.
Loss Function: CrossEntropyLoss with class weights computed using compute_class_weight to address class imbalance.
Optimizer: Adam optimizer with a learning rate of 0.001 and weight decay.
Learning Rate Scheduler: ReduceLROnPlateau to reduce the learning rate when validation loss plateaus.
Gradient Clipping: Applied with a norm value of 1.0 to prevent exploding gradients.
Early Stopping: Implemented with a patience of 10 epochs to avoid overfitting.

Performance Metrics

1. Receiver Operating Characteristic (ROC) Curves
The ROC curves demonstrate the model's ability to distinguish between classes:
Class 0 achieved an AUC of 0.97, while Class 1 achieved an AUC of 0.98, indicating excellent discriminatory power.
Both curves approach the top-left corner, reflecting high sensitivity and specificity.
2. Confusion Matrix
The confusion matrix provides insights into the classification performance:
The model correctly classified most samples across all classes but exhibited some misclassifications:
False negatives were more prominent for certain classes, particularly Class 1.
The use of class weights mitigated class imbalance but did not entirely eliminate misclassification errors.
3. Training, Validation, and Test Loss
Training loss steadily decreased over epochs, indicating effective learning.
Validation loss stabilized early, suggesting good generalization without overfitting.
Test loss remained consistent throughout training, confirming stable performance on unseen data.
4. Training, Validation, and Test Accuracy
Training accuracy increased steadily to over 95%, reflecting effective learning from the training data.
Validation accuracy stabilized at around 90%, closely aligning with training accuracy after sufficient epochs.
Test accuracy remained constant at approximately 85%, demonstrating robust generalization.

Conclusion

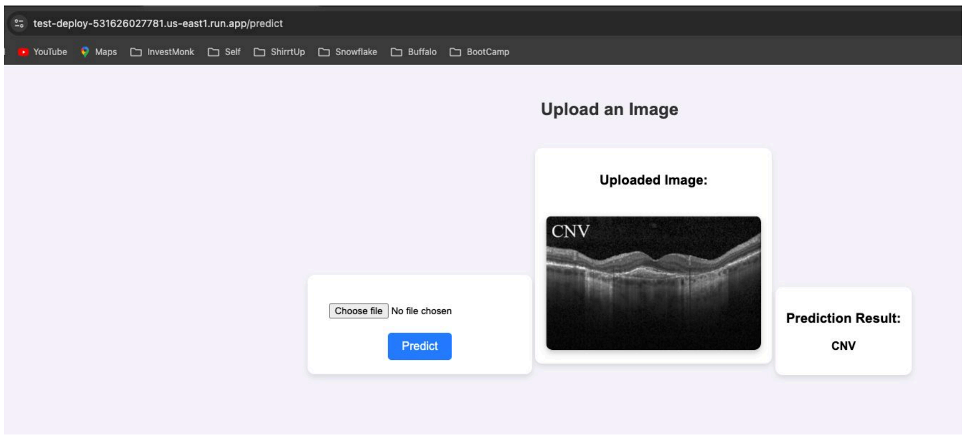
The OctMNISTModel performed well in distinguishing between classes, achieving high AUC values and consistent accuracy across datasets. The combination of techniques such as class weighting, gradient accumulation (steps = 4), early stopping, and learning rate scheduling optimized both performance and training time. While the model effectively handled class imbalance and generalized well to unseen data, further refinements could focus on reducing false negatives to improve overall classification performance further.
This model is well-suited for deployment in real-world applications requiring multi-class classification tasks with minor adjustments to address specific misclassification issues.

```
============================================================

TEST CONFUSION MATRIX
              precision    recall  f1-score   support

           0       0.71      0.96      0.81       250
           1       0.89      0.90      0.90       250
           2       0.94      0.66      0.77       250
           3       0.96      0.90      0.93       250

    accuracy                           0.85      1000
   macro avg       0.88      0.85      0.85      1000
weighted avg       0.88      0.85      0.85      1000
```

<u>**Deployed Model Recording Link**</u> :

https://buffalo.box.com/s/xo1wivlolwp1pj0h6z07u0pyq340f62g

Deployed Model:


test-deploy-531626027781.us-east1.run.app/predict
YouTube
Maps
InvestMonk
Self
ShirrtUp
Snowflake
Buffalo
BootCamp
Upload an Image
Uploaded Image:
CNV
Choose file
No file chosen
Predict
Prediction Result:
CNV