Assignment 2
CSL7590: Deep Learning
M23CSA014 - Manish V

M23CSA014_DL Assignment 2.ipynb link to the colab
[Architecture1_Model1_CNN]
[Architecture2_Head1]
[Architecture2_Head2]
[Architecture2_Head4]
references are at bottom of the page

Network Architecture:

Architecture 1:

```
----------------------
       Layer (type)
======================
           Conv1d-1
      BatchNorm1d-2
           Conv1d-3
      BatchNorm1d-4
          Dropout-5
           Conv1d-6
      BatchNorm1d-7
           Conv1d-8
      BatchNorm1d-9
          Linear-10
======================
```

Architecture 2:

```
--------------------------------
       Layer (type)
================================
           Conv1d-1
      BatchNorm1d-2
           Conv1d-3
      BatchNorm1d-4
           Conv1d-5
      BatchNorm1d-6
           Conv1d-7
      BatchNorm1d-8
 AdaptiveAvgPool2d-9
           Linear-10
           Linear-11
           Linear-12
           Linear-13
multi_head_self_attention-14
           Linear-15
           Linear-16
           Linear-17
           Linear-18
           Linear-19
multi_head_self_attention-20
           Linear-21
           Linear-22
```

Training Configuration:

Batch Size:

- Batch size = 32.

Optimizer:

- Adam Optimizer, SGD Optimizer

Loss Function:

- Cross-Entropy Loss

Learning Rate:

- lr = 0.001, 0.05

Training:

- Train for 100 epochs

Architecture 1:

Methods:

Model Implementation: PyTorch's "nn.Module" class is used to build the model:

- The activation functions of the Rectified Linear Unit (ReLU) are used after each convolutional layer.
- Multi-class classification, softmax activation are used on the output layer.

Training Loop: Two different optimizers, "Adam" and "SGD," are used to train the model, and they learn at different speeds.

- The Weights and Biases (WandB) library is used to keep track of and show the results of experiments.

Data Handling: A custom data module (called "CustomDataModule") is used to load audio data.

- The data module sets up the testing, validation, and training samples.
- Different folds of data are used for validation in k-fold cross-validation

The model is tested on a different set of data after it has been trained.

- The accuracy of the test is measured along with factors such as the F1 score and the confusion matrix.
- ROC curves are used to show how well the model works for classifying things into more than one group.

Experimentation: - Different mixtures of optimizers and learning rates are used in experiments.

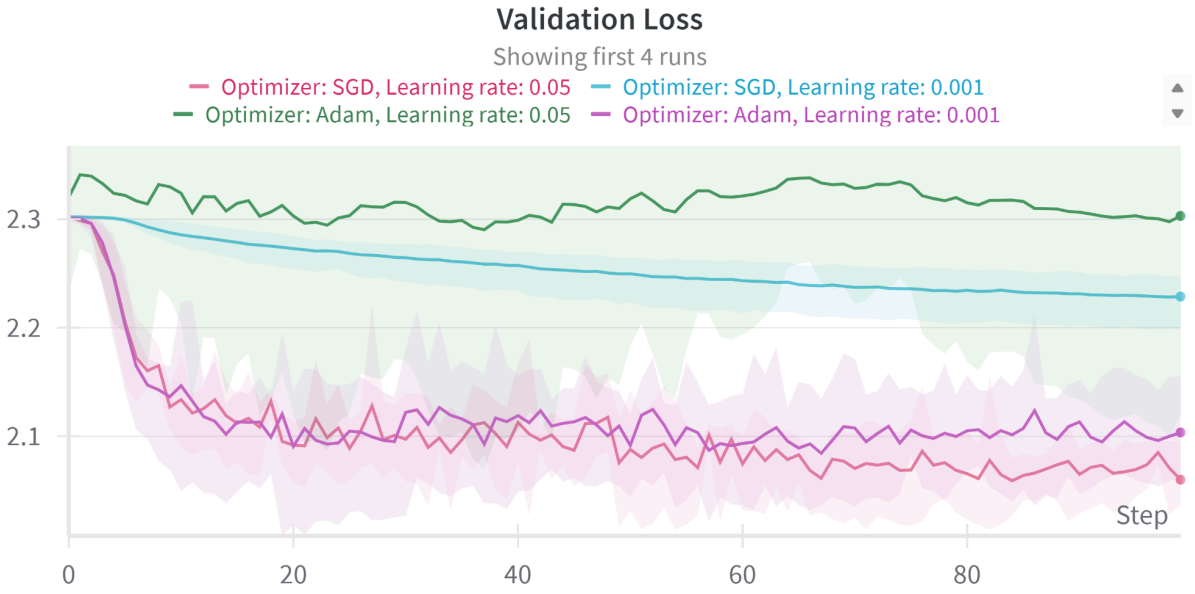
- There are logs of performance data that are compared between experiments to find the best configuration.

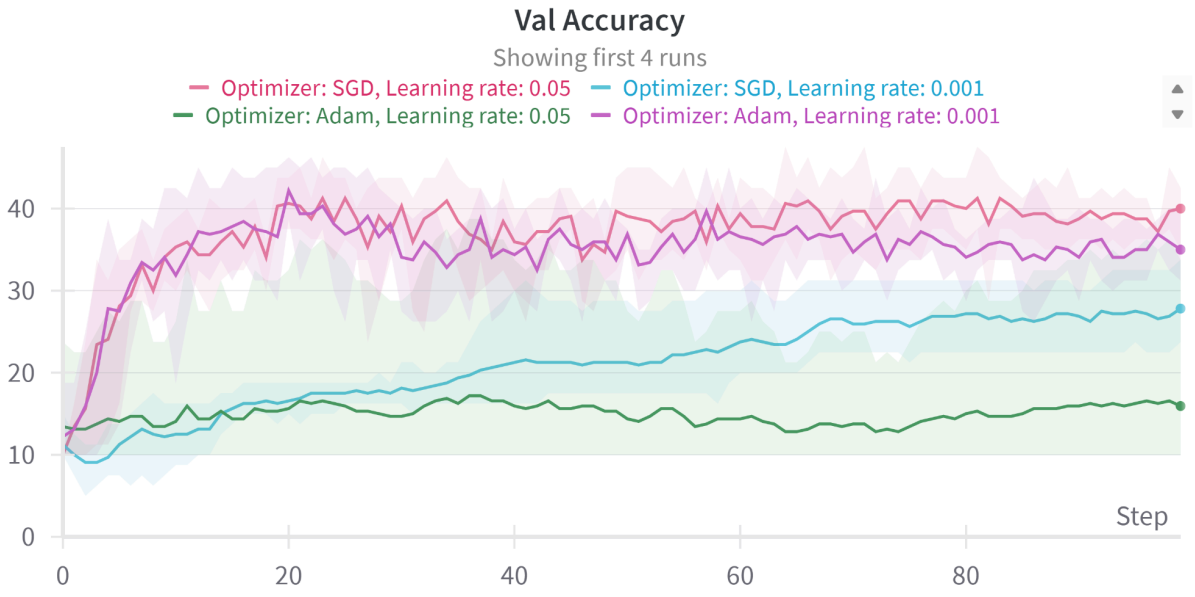
Visualization: - The Matplotlib and Seaborn libraries are used to show measures like ROC curves and confusion matrices.

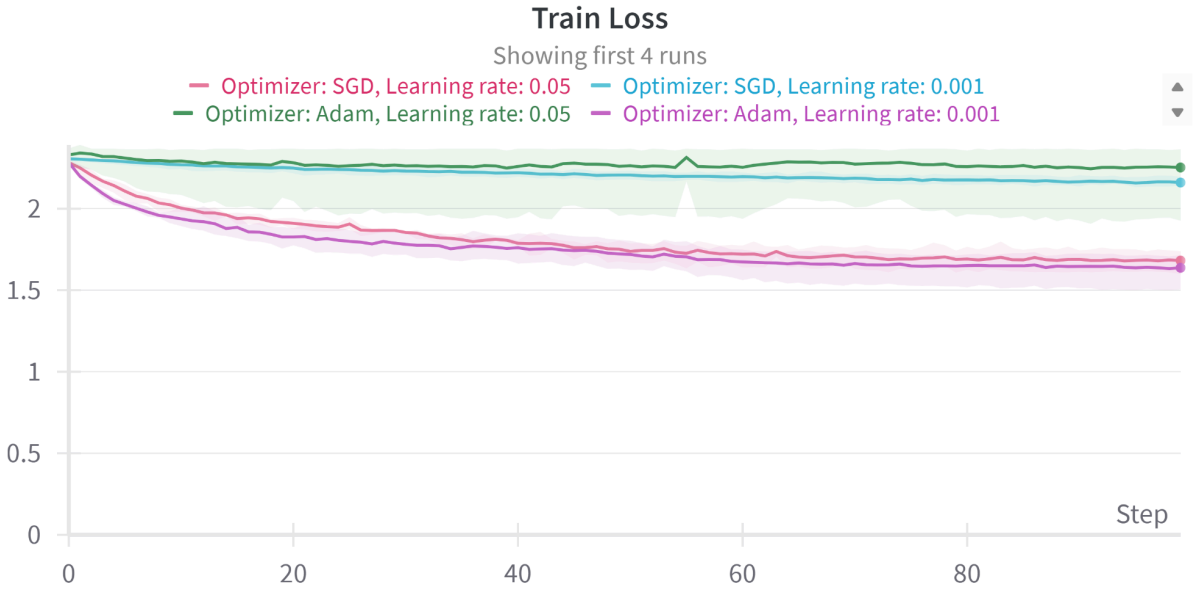
- WandB is used to keep track of visualizations and reports of experiments.

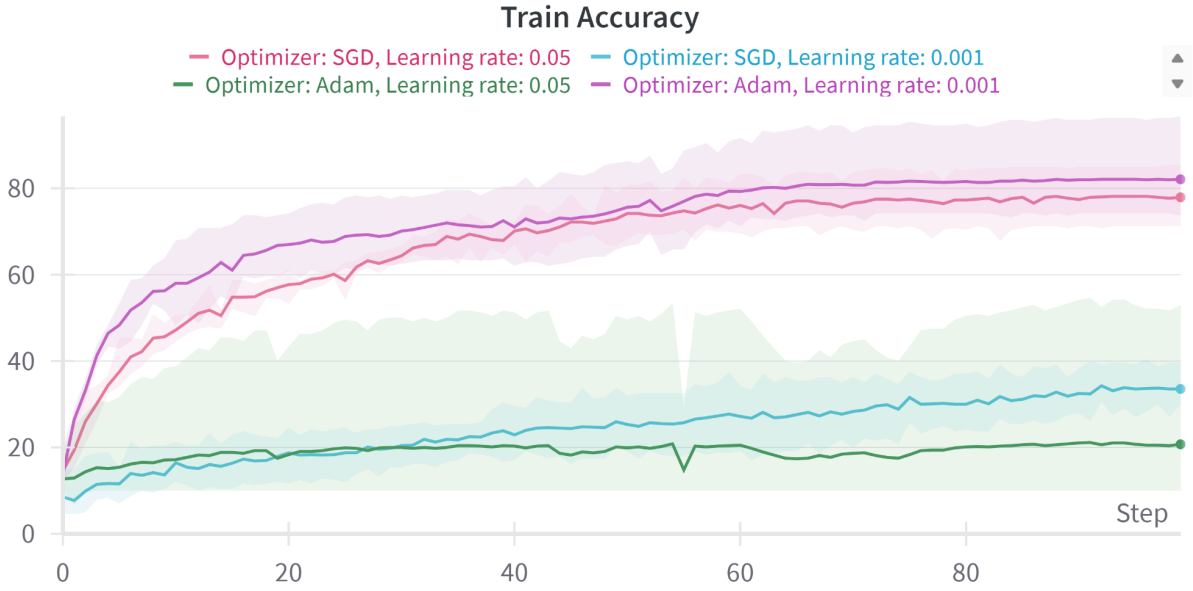
Results:

Optimizer	**Learning rate**	**Fold**	**Train acc**	**Valid acc**	**F1 score**	**Test acc**
Adam	0.001	2	96.67	40	0.39	42.5
Adam	0.001	3	75.83	31.25	0.36	41.25
Adam	0.001	4	73.75	35	0.22	26.25
Adam	0.001	5	82.08	33.75	0.29	32.5
			Avg. Valid acc	35	Best Test acc	42.5
Adam	0.05	2	10	10	0.02	10
Adam	0.05	3	10	10	0.02	10
Adam	0.05	4	52.92	33.75	0.26	35
Adam	0.05	5	10	10	0.02	10
			Avg. Valid acc	15.94	Best Test acc	35
SGD	0.001	2	30	33.75	0.18	28.75
SGD	0.001	3	30.83	23.75	0.17	21.25
SGD	0.001	4	39.58	28.75	0.23	31.25
SGD	0.001	5	33.75	25	0.15	23.75
			Avg. Valid acc	27.81	Best Test acc	31.25
SGD	0.05	2	73.75	41.25	0.28	35
SGD	0.05	3	85.42	35	0.26	28.75
SGD	**0.05**	**4**	**81.25**	**42.5**	**0.39**	**45**
SGD	0.05	5	71.25	41.25	0.33	37.5
			Avg. Valid acc	**40**	**Best Test acc**	**45**

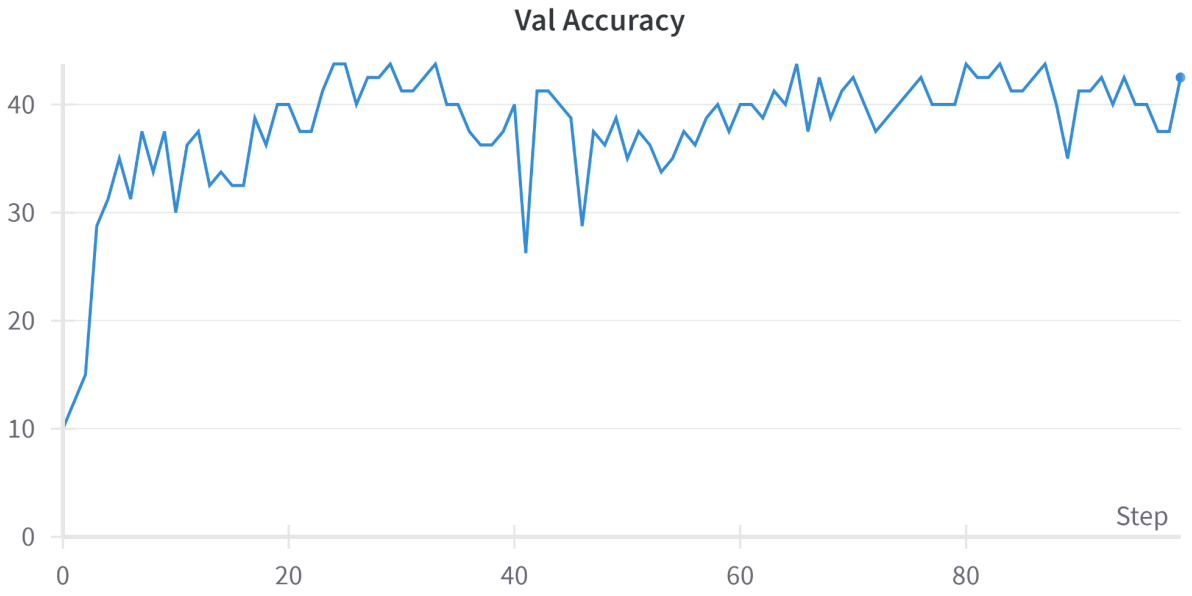

Validation Loss
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
2.3
2.2
2.1
Step
0
20
40
60
80

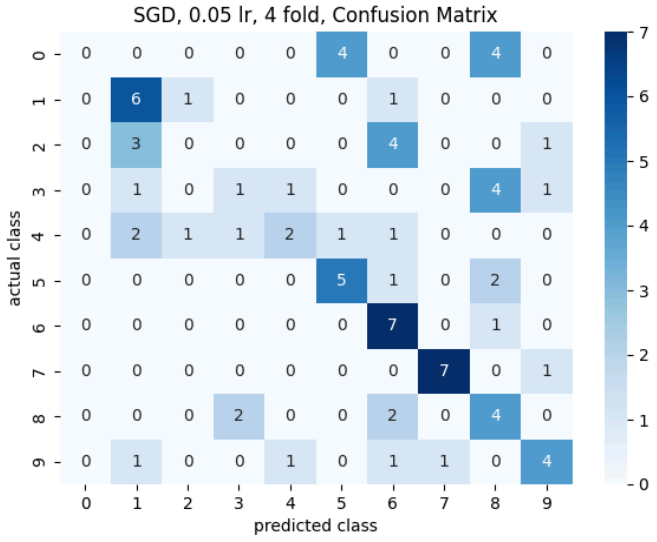

Val Accuracy
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
40
30
20
10
0
Step
0
20
40
60
80

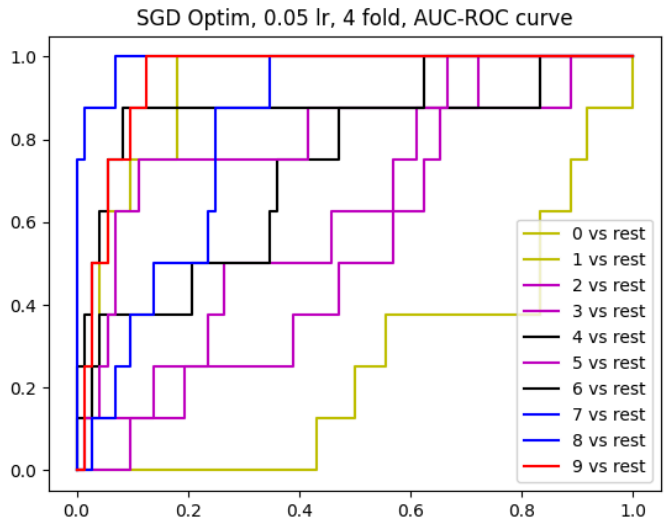




Best Model: SGD Optimizer (Learning Rate = 0.05, fold = 4)







Observations:

Adam Optimizer (Learning Rate = 0.001):

- Average validation accuracy: 35%
- Best test accuracy: 42.5%
- Shows relatively consistent performance across folds.
- Achieves a higher average validation accuracy compared to SGD with the same learning rate.
- The best test accuracy is also higher than that of SGD with the same learning rate.

Adam Optimizer (Learning Rate = 0.05):

- Average validation accuracy: 15.94%
- Best test accuracy: 35%
- Performs poorly compared to Adam with a lower learning rate and SGD with either learning rate.
- Shows consistently low accuracy across all folds.

SGD Optimizer (Learning Rate = 0.001):

- Average validation accuracy: 27.81%
- Best test accuracy: 31.25%
- Slightly lower average validation accuracy compared to Adam with the same learning rate.
- The best test accuracy is also lower than that of Adam with the same learning rate.

SGD Optimizer (Learning Rate = 0.05):

- Average validation accuracy: 40%
- **Best test accuracy: 45%**
- Outperforms all other configurations in terms of both average validation and best test accuracy.
- Shows the highest performance among all models trained with different optimizers and learning rates.

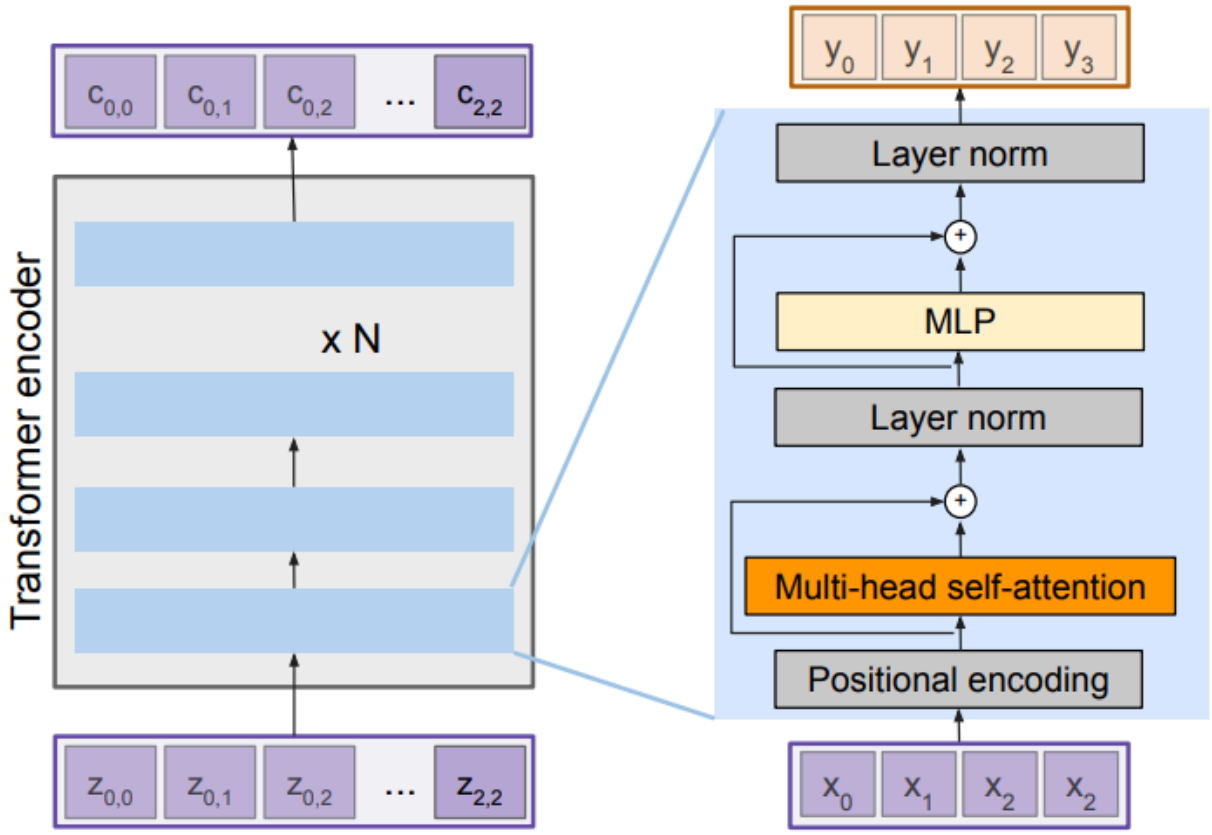
Best Model: SGD Optimizer (Learning Rate = 0.05, fold = 4)

The best performing model, trained using SGD optimizer with a learning rate of 0.05, consistently outshines other configurations in terms of both average validation accuracy and best test accuracy.

Its robust performance across different datasets suggests its effectiveness in capturing and learning the underlying patterns in the data.

SGD at a higher learning rate is demonstrating superior learning capabilities in this scenario.

Architecture 2:



The above model which was taught in the class was used to implement my transformer:

Methods:

Multi-Head Self-Attention:

- Divides the input into multiple heads and processes them separately.
- Utilizes linear projections for queries, keys, and values.
- Performs scaled dot-product attention to compute attention scores.
- It combines the outputs from different heads and applies linear transformation for unification.
- Employed two attention blocks with varying numbers of heads (1, 2, 4).

Convolutional Neural Network (CNN):

- Employs four convolutional layers with different kernel sizes and channel depths.

- Incorporates batch normalization after each convolutional layer.
- Applies adaptive average pooling to reshape the output.

Training Process:

- Uses various optimizers (Adam, SGD) and learning rates for training.
- Utilizes cross-entropy loss as the optimization criterion.
- Employs k-fold cross-validation for validation.
- Evaluates model performance using test accuracy, confusion matrix, F1 score, and ROC curves.

Transformer:

- Added a <cls> token for token-based classification.
- Positional encoding for enhancing input representations.
- Multi Head Self Attention
- Layer normalization for stabilizing training.
- Multi-layer perceptron (MLP) for additional non-linear transformations.
- Layer normalization for stabilizing training.
- Softmax activation for final classification output.

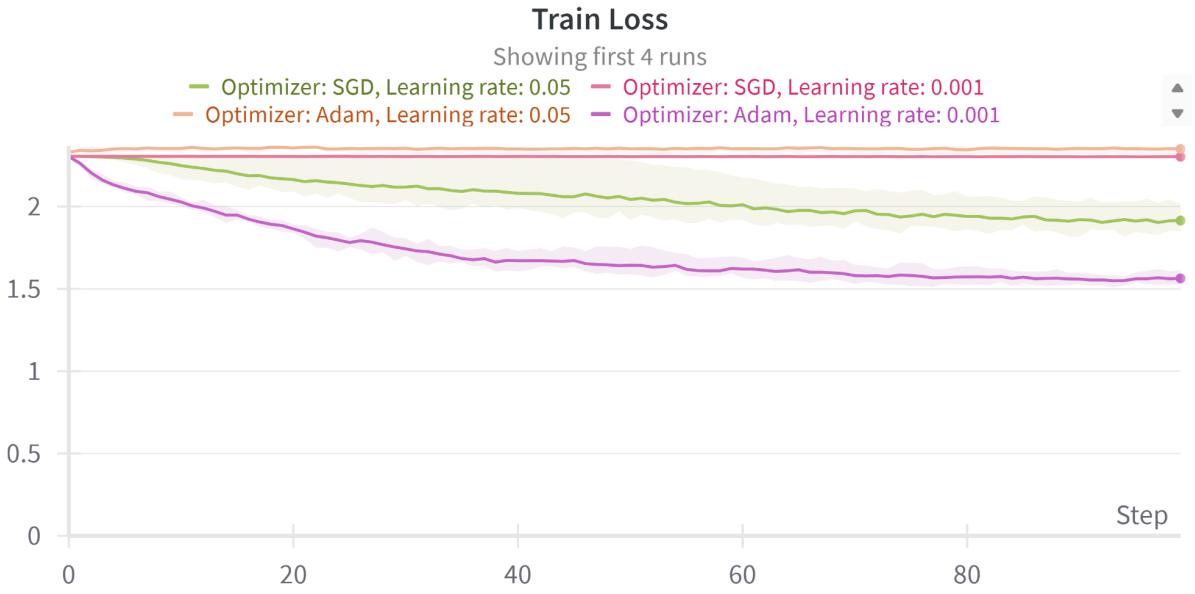
Model Evaluation:

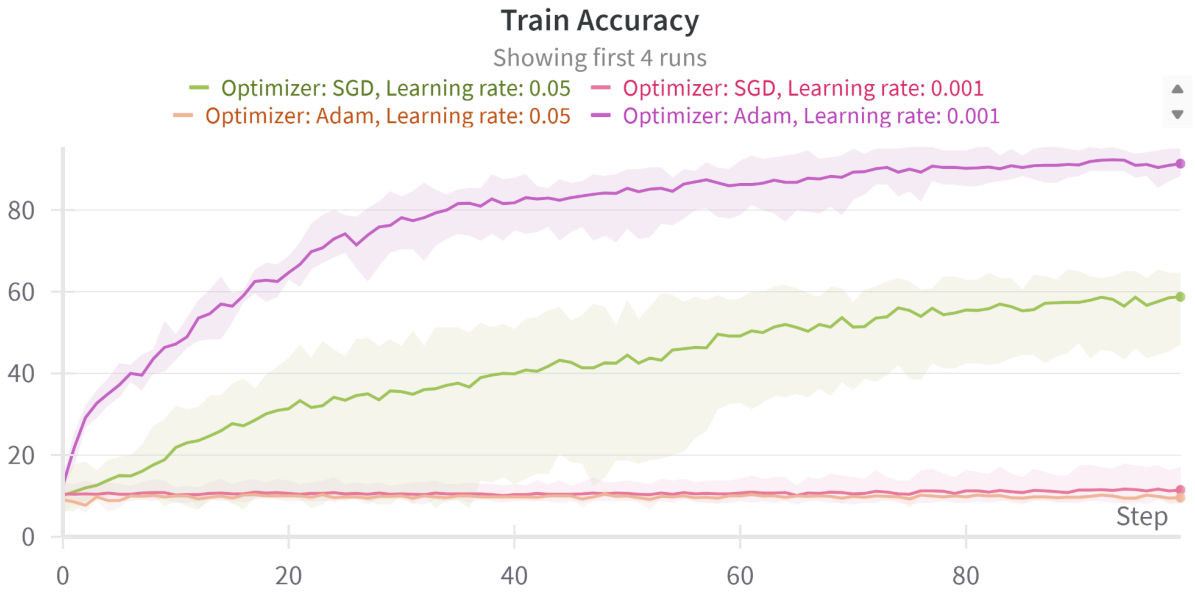
- Trained for 100 epochs with accuracy and loss plotted per epoch on the Weight and Biases (WandB) platform.
- Employed k-fold validation with k=4 to assess model performance across different data splits.
- Prepared accuracy, confusion matrix, F1-scores, and AUC-ROC curves for the test set for all network configurations.
- Reported total trainable and non-trainable parameters for each configuration.
- Conducted hyper-parameter tuning to identify the best hyper-parameter set(Optimizers and learning rates)

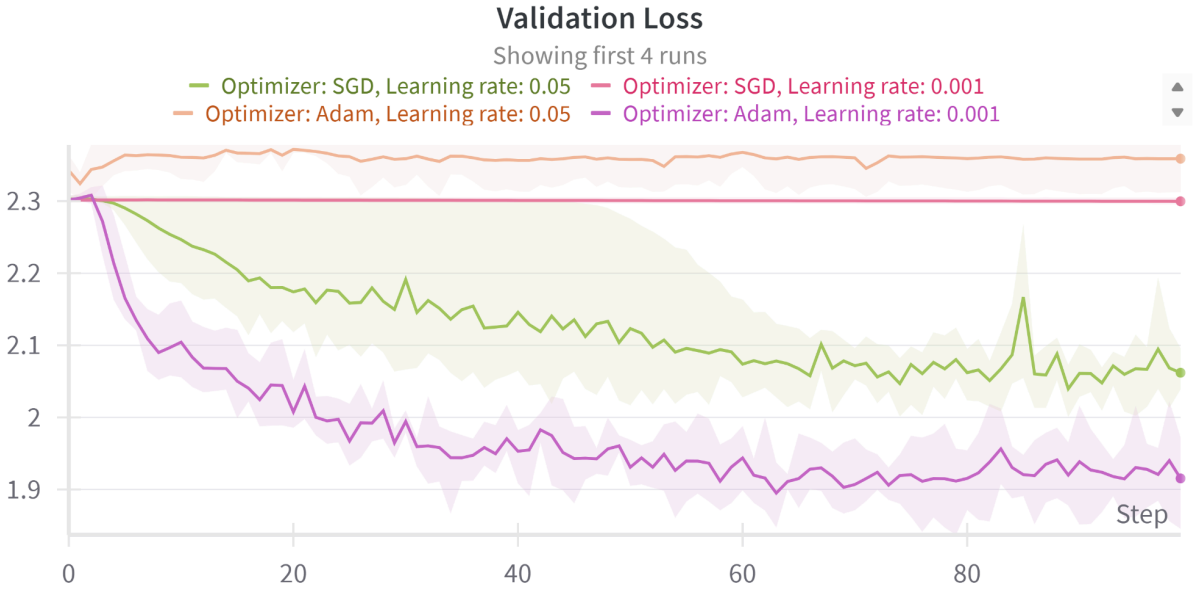
Heads = 1

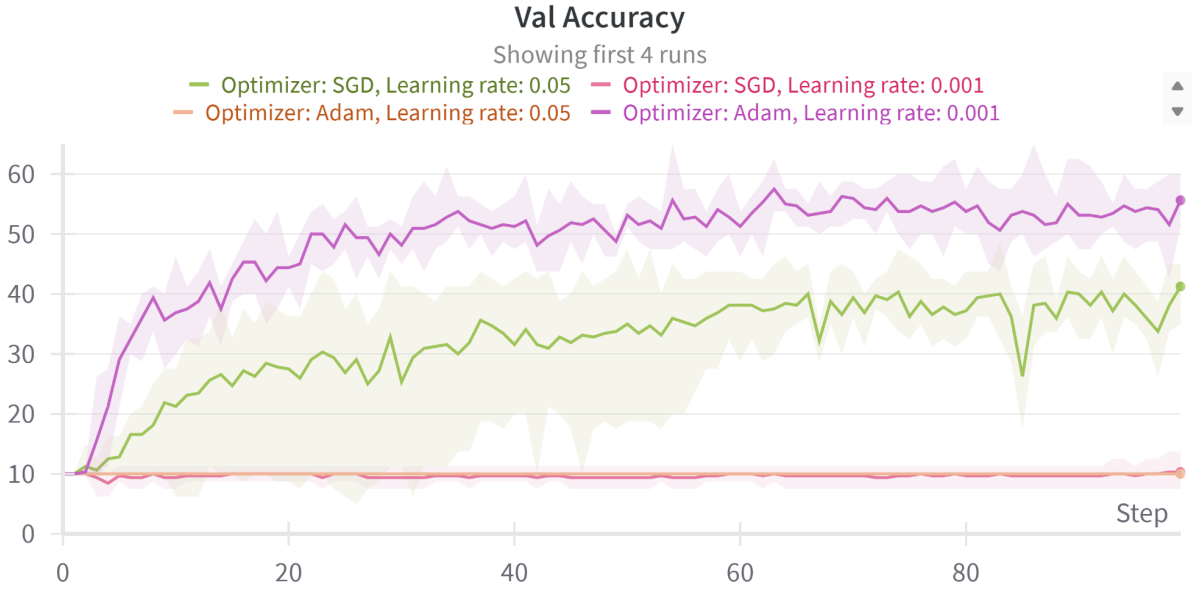
Results:

Optimizer	**Learning rate**	**Fold**	**Train acc**	**Valid acc**	**F1 score**	**Test acc**
Adam	0.001	2	88.33	57.5	0.48	51.25
Adam	0.001	3	92.5	51.25	0.44	46.25
Adam	0.001	4	95	60	0.46	51.25
Adam	**0.001**	**5**	**89.58**	**53.75**	**0.52**	**53.75**
			Avg. Valid acc	**55.62**	**Best Test acc**	**53.75**
Adam	0.05	2	10	10	0.02	10
Adam	0.05	3	10	10	0.02	10
Adam	0.05	4	8.33	10	0.02	10
Adam	0.05	5	10	10	0.02	10
			Avg. Valid acc	10	Best Test acc	10
SGD	0.001	2	8.33	10	0.26	7.5
SGD	0.001	3	10	10	0.02	10
SGD	0.001	4	17.08	13.75	0.08	15
SGD	0.001	5	10.41	7.5	0.02	10
			Avg. Valid acc	10.31	Best Test acc	15
SGD	0.05	2	47.08	41.25	0.31	38.75
SGD	0.05	3	64.58	35	0.21	26.25
SGD	0.05	4	61.25	45	0.31	37.5
SGD	0.05	5	62.08	43.75	0.35	40
			Avg. Valid acc	41.25	Best Test acc	40

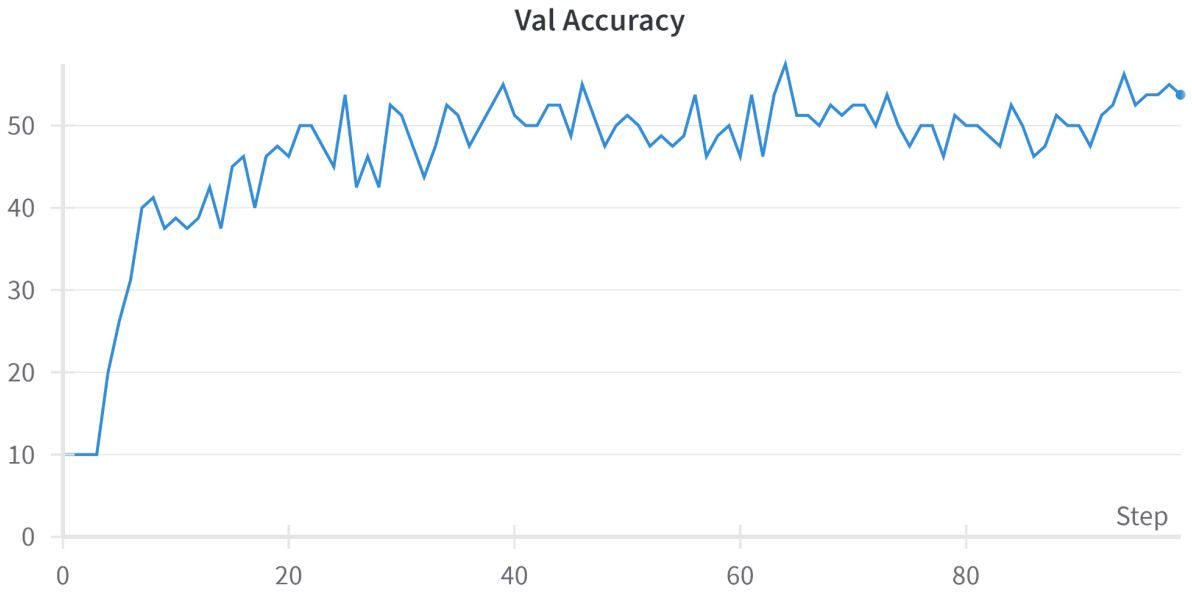

Train Loss
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
2
1.5
1
0.5
0
Step
0
20
40
60
80

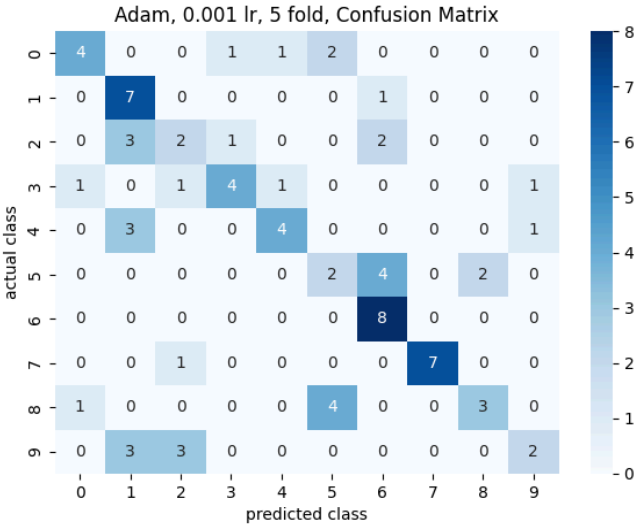

Train Accuracy
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
80
60
40
20
0
Step
0
20
40
60
80

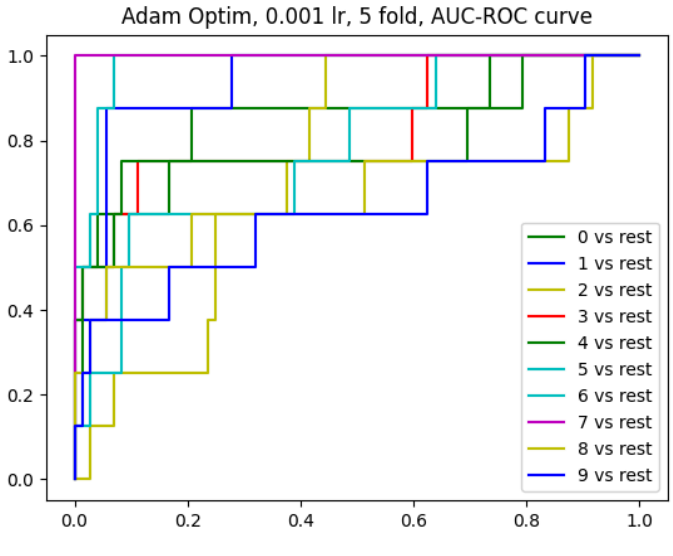




Best model: Adam, lr = 0.001, fold = 5







Observations:

Adam with Learning Rate 0.001:

- Achieves the highest average validation accuracy of 55.62%.
- Best test accuracy is 53.75%.
- Performance varies across folds, with fold 4 showing the highest validation accuracy (60%).

Adam with Learning Rate 0.05:

- Uniformly poor performance with an average validation accuracy of 10% across all folds.
- All test accuracies are 10%, indicating a lack of model generalization.

SGD with Learning Rate 0.001:

- Achieves an average validation accuracy of 10.31%.
- The best test accuracy among SGD configurations is 15%, observed in fold 4.

SGD with Learning Rate 0.05:

- Demonstrates better performance compared to the higher learning rate Adam variant.
- Average validation accuracy is 41.25%.
- Fold 3 exhibits the highest validation accuracy (64.58%), but its test accuracy is relatively lower.

Best Model:

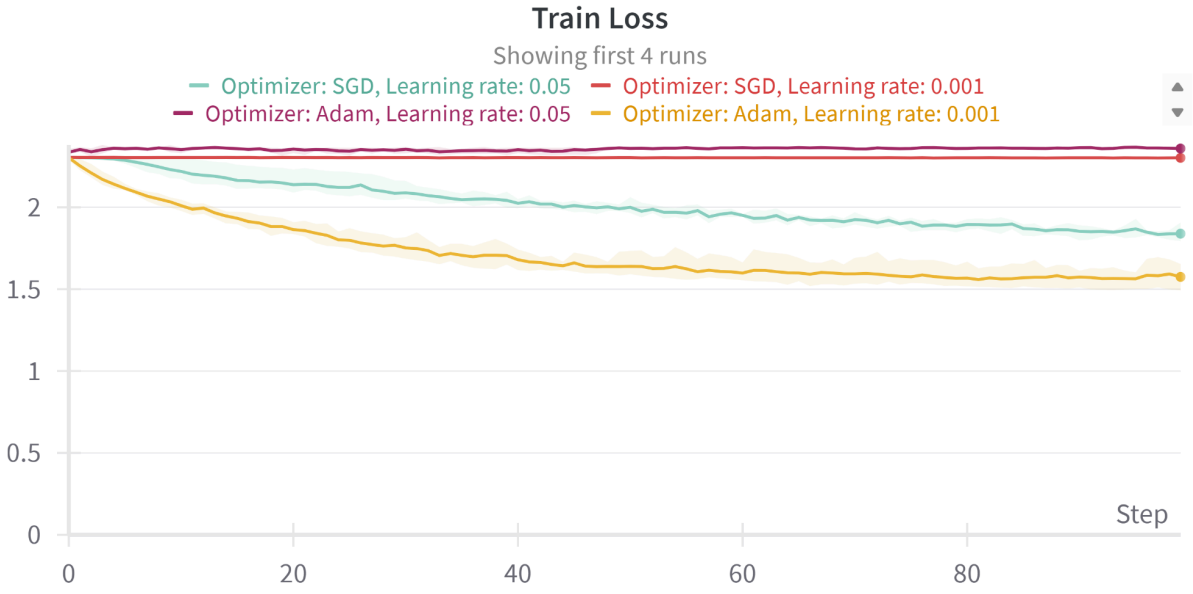
- The best model is trained using the Adam optimizer with a learning rate of 0.001.
- It achieves the highest test accuracy of 53.75% among all configurations.
- This model demonstrates good performance on the validation set, particularly in fold 4, where it achieves a validation accuracy of 60%.
- The model shows effective generalization to unseen data, indicating its robustness and suitability for the classification task.

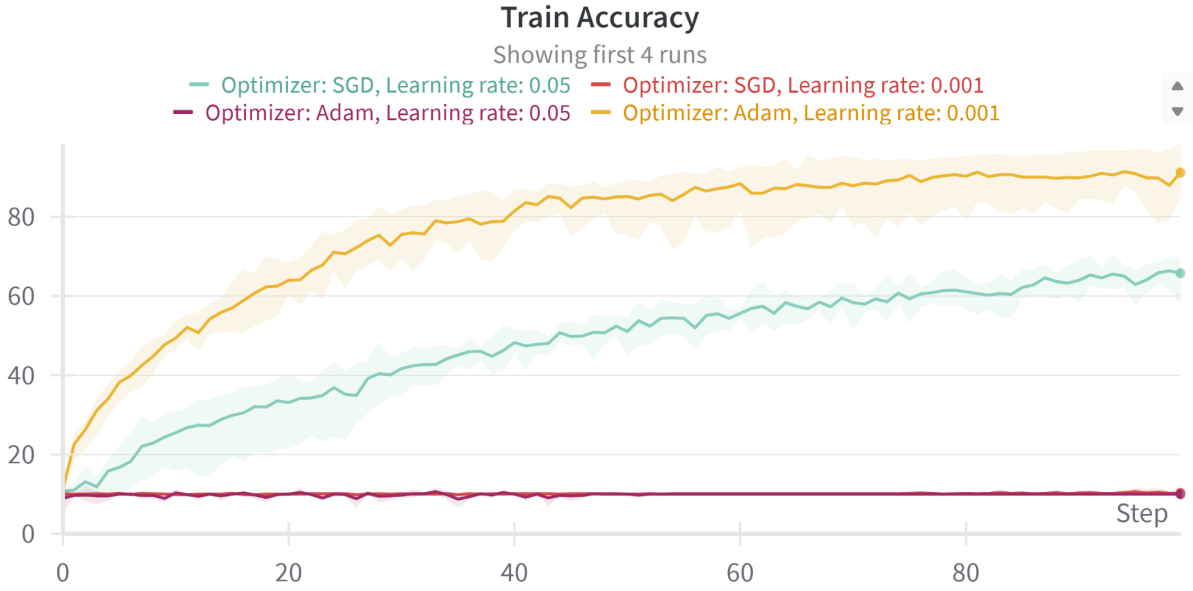
Heads = 2

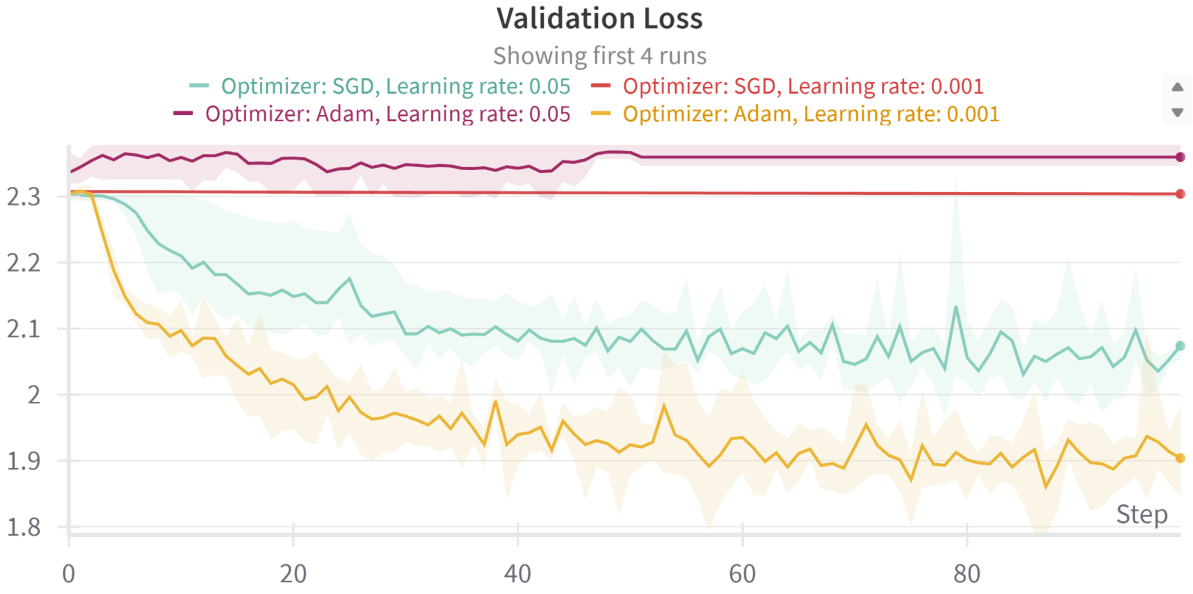
Results:

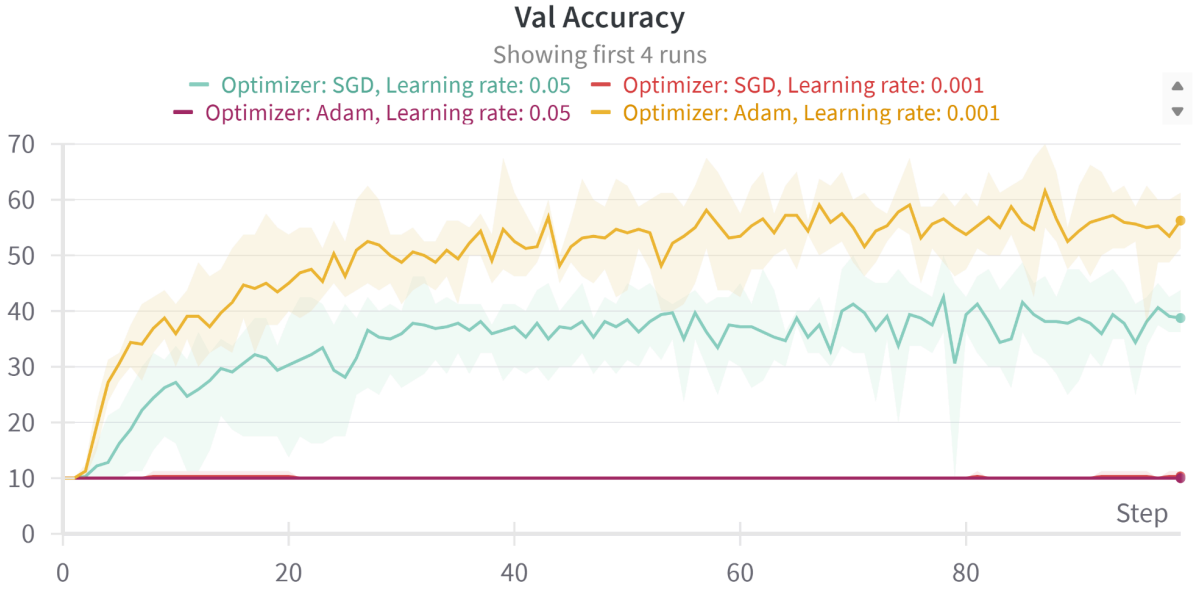
Optimizer	**Learning rate**	**Fold**	**Train acc**	**Valid acc**	**F1 score**	**Test acc**
Adam	0.001	2	89.17	51.25	0.41	41.25
Adam	0.001	3	98.33	53.75	0.46	48.75
Adam	**0.001**	**4**	**85**	**61.25**	**0.59**	**60**
Adam	0.001	5	92.08	58.75	0.53	55
			Avg. Valid acc	**56.25**	**Best Test acc**	**60**
Adam	0.05	2	10	10	0.02	10
Adam	0.05	3	10	10	0.02	10

Adam	0.05	4	10	10	0.02	10
Adam	0.05	5	10	10	0.02	10
			Avg. Valid acc	10	Best Test acc	10
SGD	0.001	2	10.83	10	0.02	10
SGD	0.001	3	10.83	11.25	0.04	11.25
SGD	0.001	4	10	10	0.02	10
SGD	0.001	5	9.58	10	0.02	10
			Avg. Valid acc	10.313	Best Test acc	11.25
SGD	0.05	2	57.92	36.25	0.23	30
SGD	0.05	3	69.17	38.75	0.32	37.5
SGD	0.05	4	69.58	43.75	0.31	37.5
SGD	0.05	5	66.25	36.25	0.31	36.25
			Avg. Valid acc	38.75	Best Test acc	37.5

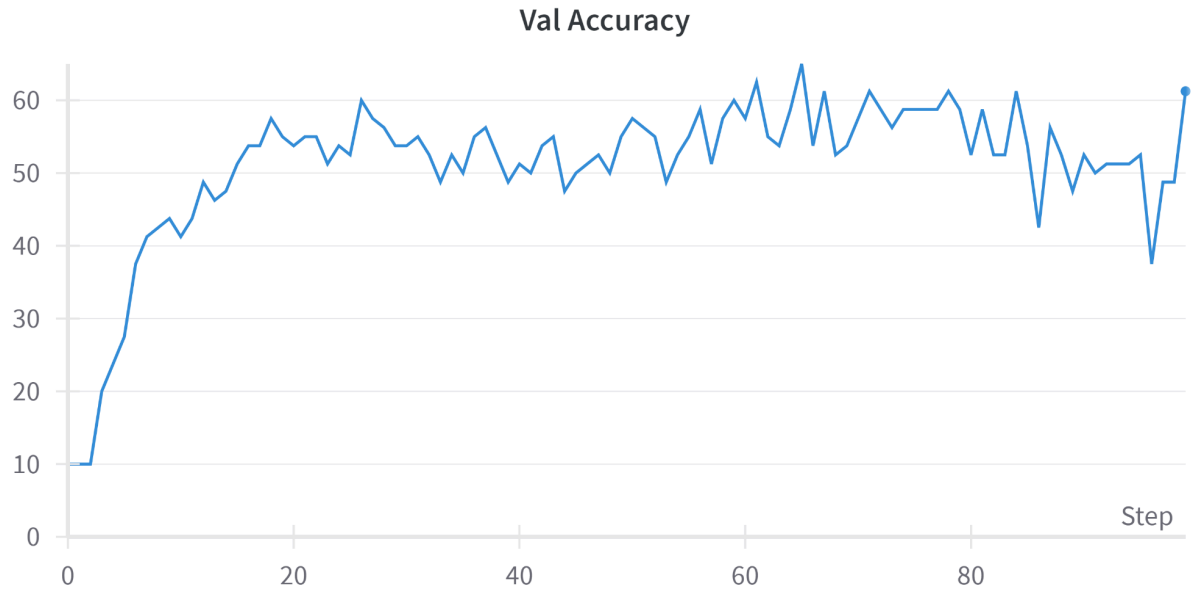

Train Loss
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
2
1.5
1
0.5
0
Step
0
20
40
60
80

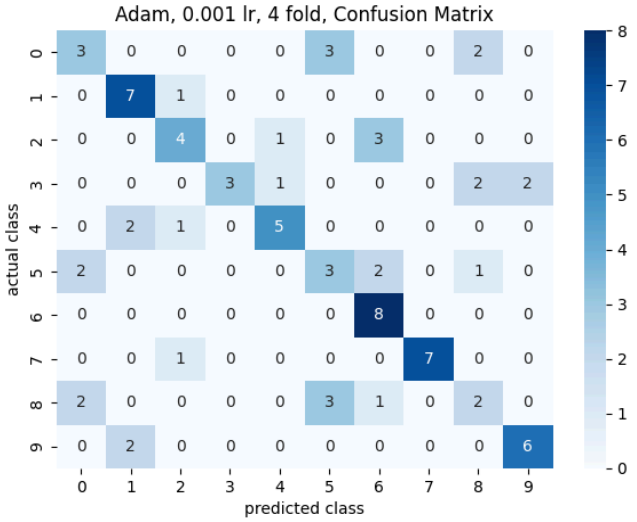

Train Accuracy
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
80
60
40
20
0
Step
0
20
40
60
80

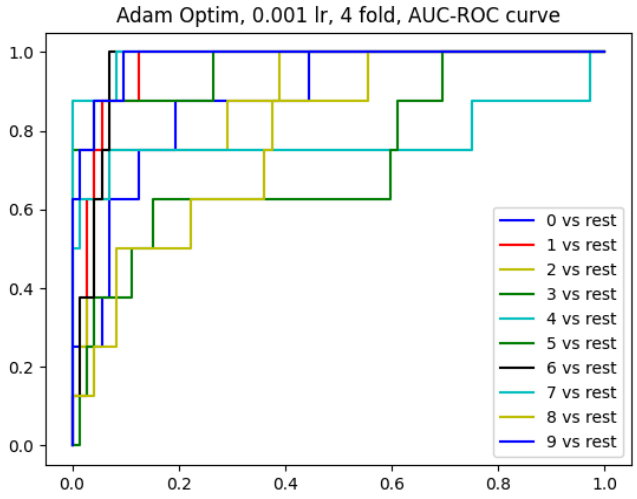




Best model: Adam, lr = 0.001, fold = 4







Observations:

Adam with Learning Rate 0.001:

- Achieves the highest test accuracy of 60%.
- Demonstrates consistent performance across folds, with the highest validation accuracy observed in fold 4 (61.25%).
- Shows an average validation accuracy of 56.25%.

Adam with Learning Rate 0.05:

- All configurations show poor performance with an average validation accuracy of 10% and a test accuracy of 10%.

SGD with Learning Rate 0.001:

- Achieves an average validation accuracy of 10.313%.
- The best test accuracy among SGD configurations is 11.25%, observed in fold 3.

SGD with Learning Rate 0.05:

- Demonstrates better performance compared to the higher learning rate Adam variant.
- Average validation accuracy is 38.75%.
- The best test accuracy is 37.5%, observed in fold 3.

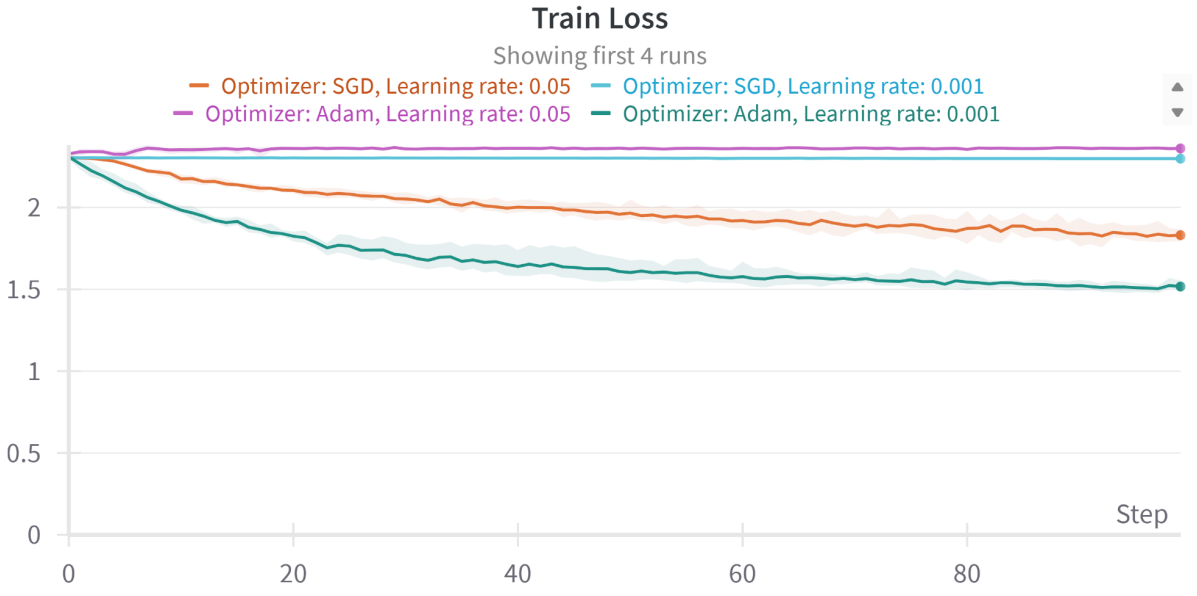
Overall, the best model is trained with Adam and a learning rate of 0.001, achieving the highest test accuracy of 60%.

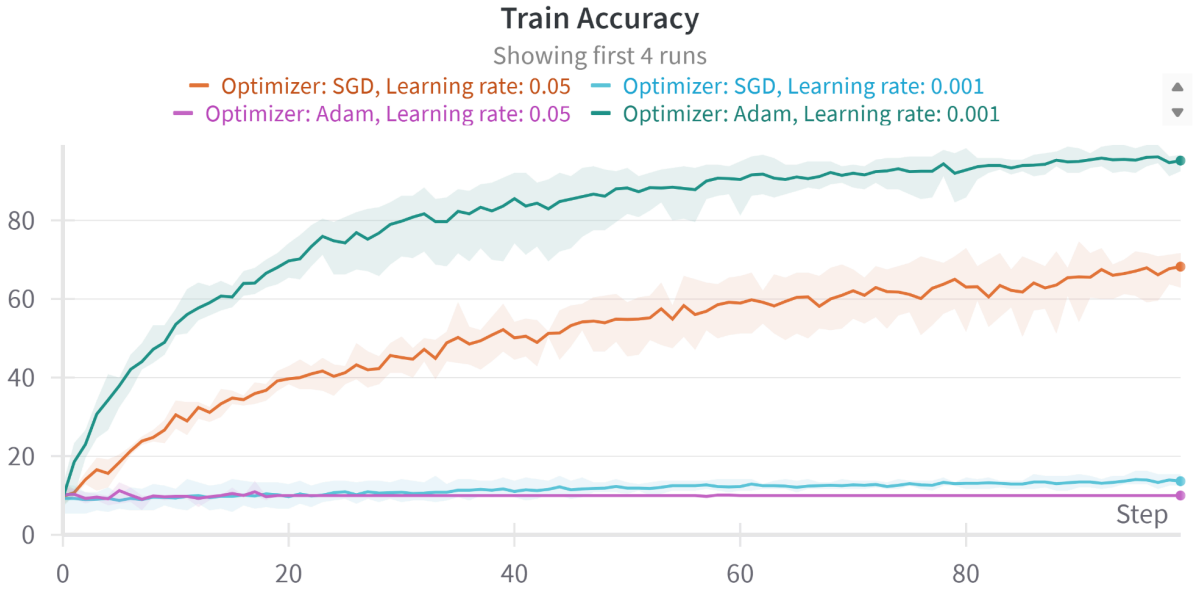
Heads = 4

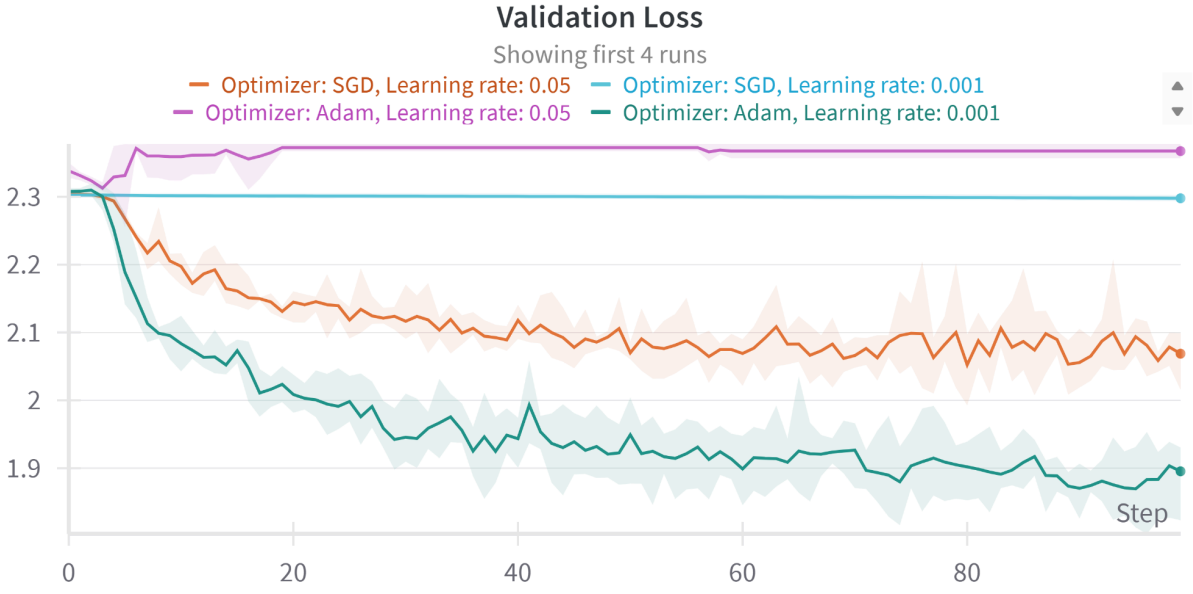
Results:

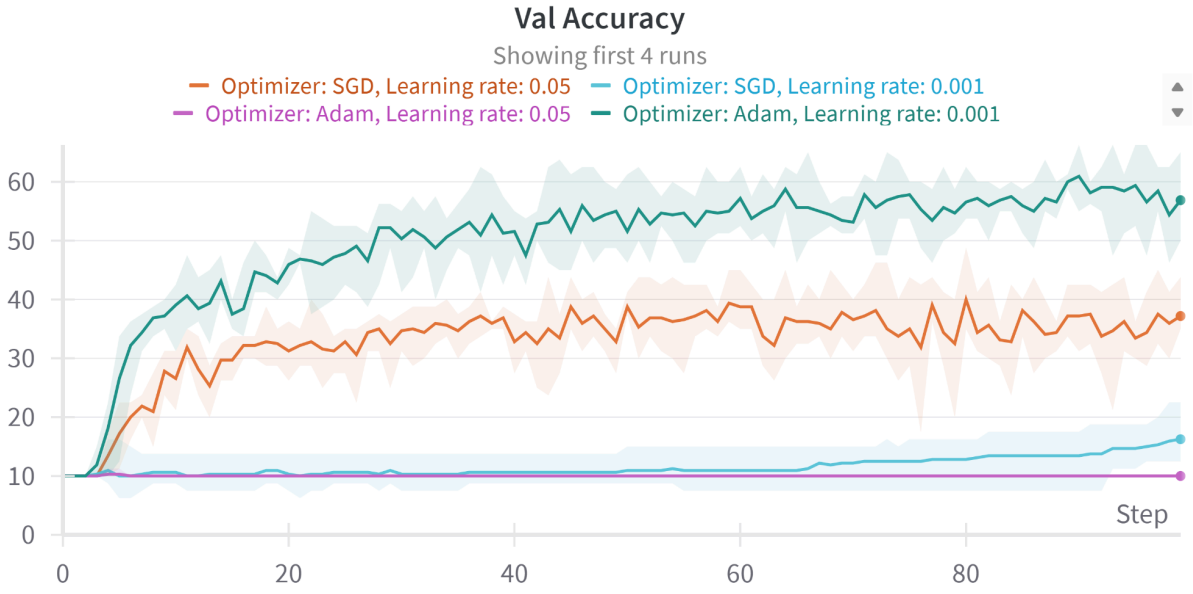
Optimizer	**Learning rate**	**Fold**	**Train acc**	**Valid acc**	**F1 score**	**Test acc**
Adam	0.001	2	92.5	50	0.46	47.5
Adam	0.001	3	96.67	65	0.52	55
Adam	0.001	4	95.42	57.5	0.50	50
Adam	**0.001**	**5**	**96.25**	**55**	**0.55**	**57.5**
			Avg. Valid acc	**56.88**	**Best Test acc**	**57.5**
Adam	0.05	2	10	10	0.02	10
Adam	0.05	3	10	10	0.02	10
Adam	0.05	4	10	10	0.02	10
Adam	0.05	5	10	10	0.02	10
			Avg. Valid acc	10	Best Test acc	10

SGD	0.001	2	13.75	16.25	0.04	11.25
SGD	0.001	3	15.42	22.5	0.08	15
SGD	0.001	4	12.92	12.5	0.04	12.5
SGD	0.001	5	12.5	13.75	0.06	10
			Avg. Valid acc	16.25	Best Test acc	12.5
SGD	0.05	2	68.33	35	0.30	37.5
SGD	0.05	3	62.92	35	0.23	28.75
SGD	0.05	4	70	43.75	0.38	42.5
SGD	0.05	5	71.67	35	0.26	35
			Avg. Valid acc	37.19	Best Test acc	42.5

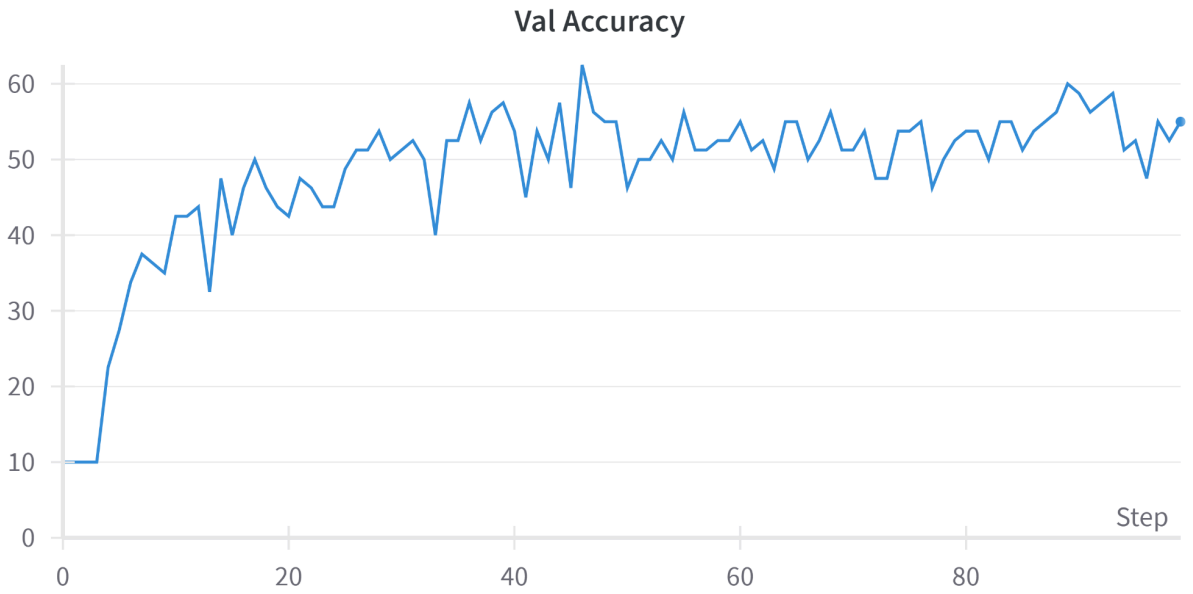


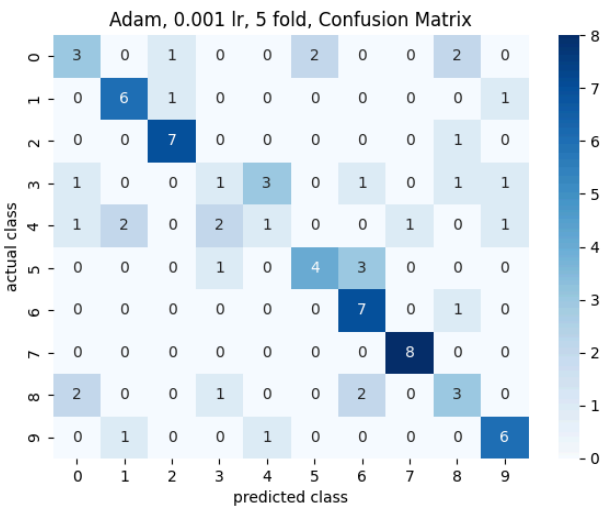

Train Accuracy
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
80
60
40
20
0
Step
0
20
40
60
80

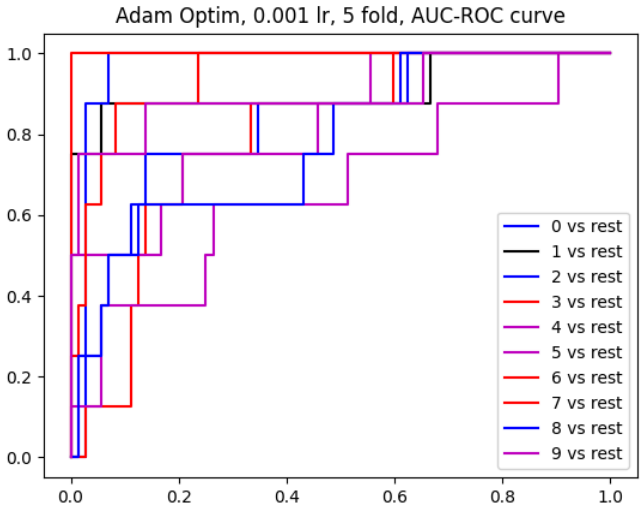

Validation Loss
Showing first 4 runs
Optimizer: SGD, Learning rate: 0.05
Optimizer: SGD, Learning rate: 0.001
Optimizer: Adam, Learning rate: 0.05
Optimizer: Adam, Learning rate: 0.001
2.3
2.2
2.1
2
1.9
Step
0
20
40
60
80



Best model: Adam, lr = 0.001, fold = 5







Observations:

Adam with Learning Rate 0.001:

- Achieves the highest test accuracy of 57.5%.
- Demonstrates varying performance across folds, with the highest validation accuracy observed in fold 3 (65%).
- Shows an average validation accuracy of 56.88%.

Adam with Learning Rate 0.05:

- All configurations show poor performance with an average validation accuracy of 10% and a test accuracy of 10%.

SGD with Learning Rate 0.001:

- Achieves an average validation accuracy of 16.25%.
- The best test accuracy among SGD configurations is 12.5%, observed in fold 4.

SGD with Learning Rate 0.05:

- Demonstrates better performance compared to the higher learning rate Adam variant.
- Average validation accuracy is 37.19%.
- The best test accuracy is 42.5%, observed in fold 4.

Overall, the best model is trained with Adam and a learning rate of 0.001, achieving the highest test accuracy of 57.5%.

Final Observations:

The CNN model has a lower test accuracy of 45%
But the transformer, since they take into account the context with positional encodings and using multi head attention, gives best accuracy of 60% for heads = 2
Heads with 1 or 4 are too little or too much, making it more complex or not even being able to use it well enough.
Adam's learning rate needs to be around 0.001 but SGD needs 0.05 or higher so that it can learn fast

Parameters:

	Total Parameters	Trainable Parameters	No trainable parameters
Architecture 1	12212	12212	0
Architecture 2	98730	98730	0

References:

Architecture:
Attention.pdf

1) Positional Encoding:

A Gentle Introduction to Positional Encoding in Transformer Models, Part 1 - MachineLearningMastery.com

2) Multi head attention:

Tutorial 5: Transformers and Multi-Head Attention — PyTorch Lightning 2.2.0.post0 documentation