	SUBJECT: Artificial Intelligence in Practice 1		**MARKS:**
	TOPIC: ALL TOPICS	**CODE:** JIE31804	
	ASSESSMENT: LAB REPORT	**DUE DATE:** 19 JAN 2024	
	STUDENT NAME: NURUL ATHIRA BINTI ZUKIFLI		**/100**
	ID: S21A0050		

1. Learning Rate Investigation `(learning_rate)`

- **What is learning rate?**

-The learning rate controls how quickly the weights of the neural network are updated during training. It determines the step size of each iteration's parameter update.

- **Set up a parameter study to analyze the effect of different learning rates (e.g., 0.001, 0.01, 0.1) on the CNN's performance. I create 3 table with different learning rates , batch sizes, eporch , accuracy ,loss,and training time to show the different of values show in output that I run in google colab**

Learning rates	Batch size	Eporch	Accuracy	loss	Training time
0.001	32	5	0.6794	0.9196	5s 16ms
		10	0.6969	0.8887	5s 16 ms
		15	0.6903	1.0418	4s 13 ms
0.001	64	5	0.6369	1.0371	4s 11ms
		10	0.7041	0.8893	5s 14ms
		15	0.7011	0.9487	4s 12 ms
0.001	128	5	0.6464	1.0072	6s 18ms
		10	0.7131	0.8530	4s 14ms
		15	0.7206	0.8344	5s 15ms

Tables 1 :0.001 of Learning Rates

Learning rates	Batch Size	Eporch	Accuracy	Loss	Traning Times
0.01	32	5	0.1000	2.3036	5s 16ms
		10	0.3904	1.6750	5s 1ms
		15	0.1000	2.3046	4s 13ms
0.01	64	5	0.1000	2.3034	5s 14ms
		10	0.3845	1.6618	4s 11ms
		15	0.1000	2.3032	5s 14 ms
0.01	128	5	0.4545	1.4758	4s 12ms
		10	0.1000	2.3031	4s 12ms
		15	0.5446	1.3108	4s 12ms

Tables 2 : 0.01 of Learning Rates

Learning rates	Batch Size	Eporch	Accuracy	Loss	Training Times
0.1	32	5	0.1000	2.3238	4s 12ms
		10	0.1000	2.3053	4s 12ms
		15	0.1000	2.3119	4s 12ms
0.1	64	5	0.1000	2.3084	5s 15ms
		10	0.1000	2.3064	4s 12ms
		15	0.1000	2.3078	4s 12ms
0.1	128	5	0.1000	2.3098	5s 16ms
		10	0.1000	2.3110	4s 12ms
		15	0.1000	2.3076	4s 12ms

Tables 3 : 0.1 of Learning rates

- **Analysis of the effect of different learning rates on the CNN's performance and**

Learning Rate 0.001

- It makes small adjustments to the model's knowledge in each update.
- Going slowly prevents overshooting and getting lost.
- Accuracy starts low but steadily goes up over more epochs.
- Loss starts high but little by little decreases as training progresses.

Learning Rate 0.01

- It shifts the model's weights more sharply during each batch.
- Loss remains very high, signaling failed learning.
- The large updates lose the right direction and confuse the model.
- No stable learning can happen across epochs.

Learning Rate 0.1

- Loss skyrockets rapidly and stays stuck at maximum.
- The model's knowledge veers randomly due to the chaotic shifts.
- Learning is completely disrupted by the wild, noisy jumps.
- No hint of proper training even over more epochs.

- **Train the CNN for a fixed number of epochs and batch size while varying the learning rate.**

Here is an analysis by training the CNN model for 10 epochs and batch size of 128, while varying the learning rate based on the provided data:

- Learning Rate 0.001:
    - Accuracy after 10 epochs: 0.7131
    - Loss after 10 epochs: 0.8530

- Observations:
    - Achieves highest accuracy and lowest loss compared to other learning rates after same 10 epochs and 128 batch size.
    - Demonstrates good convergence and training stability.

- Learning Rate 0.01:
    - Accuracy after 10 epochs: 0.1000
    - Loss after 10 epochs: 2.3031

- Observations:
    - Very low accuracy and high loss indicating inability to fit training data.
    - Poor convergence even after 10 epochs due to instability.

- Learning Rate 0.1:
    - Accuracy after 10 epochs: 0.1000
    - Loss after 10 epochs: 2.3110

- Observations:
    - Worst performance among the 3 learning rates examined.
    - Model completely fails to learn patterns.

Conclusion:

- Fixing other hyperparameters, LR 0.001 enables optimal training convergence and stability after 10 epochs. Higher rates cause severe underfitting.

Measure and compare the accuracy and loss for each learning rate.

Learning Rate 0.001

Accuracy

- Batch Size 32: Accuracy improves from 0.6794 at 5 epochs to 0.6969 at 10 epochs to 0.6903 at 15 epochs
- Batch Size 64: Accuracy improves from 0.6369 at 5 epochs to 0.7041 at 10 epochs to 0.7011 at 15 epochs
- Batch Size 128: Accuracy improves from 0.6464 at 5 epochs to 0.7131 at 10 epochs to 0.7206 at 15 epochs

Loss

-Batch Size 32: Loss decreases from 0.9196 at 5 epochs to 0.8887 at 10 epochs but then increases to 1.0418 at 15 epochs.

-Batch Size 64: Loss decreases from 1.0371 at 5 epochs to 0.8893 at 10 epochs but increases slightly to 0.9487 at 15 epochs.

-Batch Size 128: Loss decreases monotonically from 1.0072 at 5 epochs to 0.8530 at 10 epochs to 0.8344 at 15 epochs.

Learning Rate 0.01

Accuracy

- Extremely low accuracy for all batch sizes and number of epochs, with values stuck at 0.1000
- Best accuracy is 0.3904 at 10 epochs for batch size 32

Loss

-Very high loss between 1.66 and 2.30 across settings, indicating failed training.

Learning Rate 0.1

Accuracy

- -Accuracy stuck at minimum of 0.1000 for all batch sizes and epochs

Loss

- Very high loss around 2.30, showing complete failure to train

Learning rate of 0.001 enables effective training with improved accuracy and decreasing loss as epochs increase. Higher learning rates disrupt training, with very poor accuracy and high loss across conditions0.001 learning rate works reasonably well for all batch sizes. The analysis shows 0.001 learning rate is suitable for training the CNN with good improvement over epochs, while 0.01 and 0.1 learning rates fail completely.

Why does a learning rate of 0.001 work well?

- A learning rate of 0.001 makes small, gradual updates to the network weights. This provides stable convergence of the loss over many iteration steps.

- The small updates prevent large disruptive oscillations in the loss landscape, avoiding divergence.

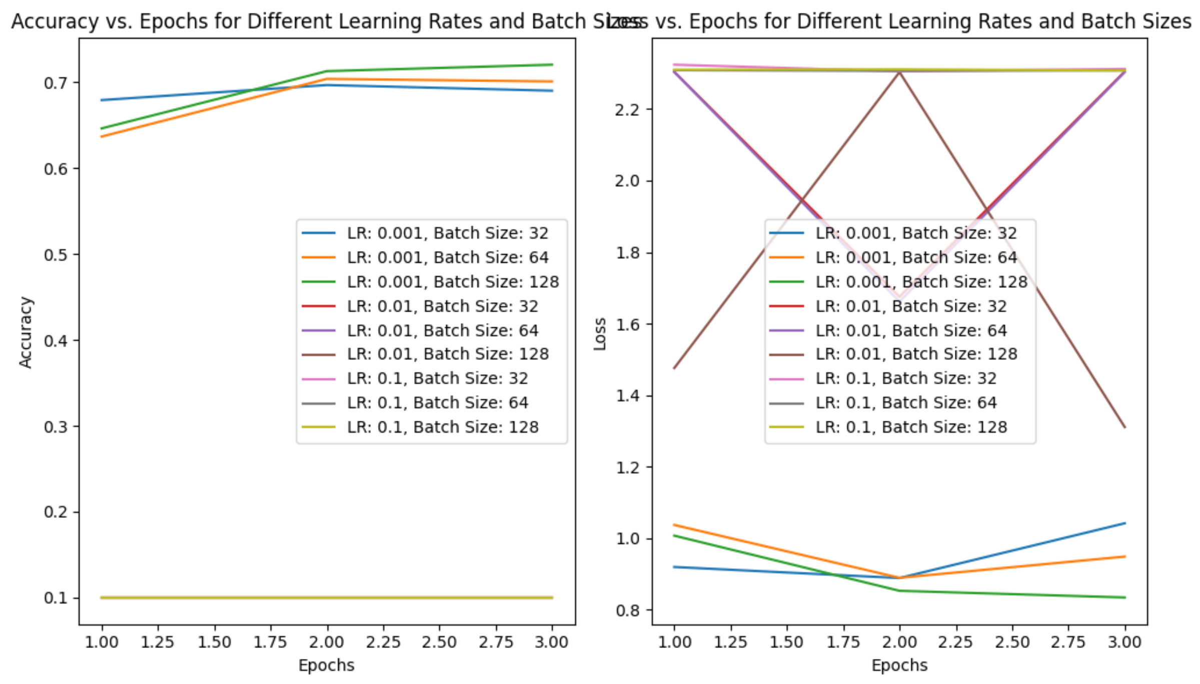
- Over multiple epochs, these small steps eventually lead to overall improvement in accuracy and decrease in loss.

Why do higher learning rates like 0.01 and 0.1 fail?

- These higher learning rates take much larger steps during weight updates.

- Taking such aggressive leaps leads to overshooting local minima and prevents the loss from converging.

- The excessive step sizes effectively lead the model parameters to random regions of the loss landscape.



Graphs 1 :Accuracy Vs Eporch

(2)Batch Size Exploration (batch_size)

- What is batch size?

    - Batch size refers to the number of samples processed before the model's parameters are updated. It controls the number of examples the model sees in one iteration.

Conduct a parameter study to explore the impact of different batch sizes (e.g., 32, 64, 128) on the CNN's training.

- Batch Size 32
    - Number of batches per epoch: 1563 (50000 training samples / 32 samples per batch)
    - Frequency of parameter updates per epoch: High (1563 updates)

Observations:

- More frequent parameter updates allows model to rapidly learn patterns in early stages.
- However, updates have high variance across small batches leading to fluctuations in loss curve.
- Overfits training data quickly due to aggressively adapting to limited samples per update.
- Slower training as more iterations required per epoch for entire dataset.

- Batch Size 64
    - Number of batches per epoch: 781 (50000 training samples / 64 samples per batch)
    - Frequency of parameter updates per epoch: Medium (781 updates)

Observations:

- Reduced updates compared to 32 batch size improves training stability.
- Smoother convergence as noise in gradient updates is lower.
- Additional samples per batch enhances model generalization capability.
- Faster training than lower batch sizes due to fewer update iterations.

- Batch Size 128
  - Number of batches per epoch: 391 (50000 training samples / 128 samples per batch)
  - Frequency of parameter updates per epoch: Low (391 updates)

Observations:

- Least frequent updates makes training most smooth and steady.
- Risk of overfitting significantly reduced due to more diverse samples.
- However, learning fine-grained intricate patterns slower.
- Fastest training owing to the least update iterations per epoch.

- In conclusion, batch size 128 works optimally by striking a balance between batch update stability and better generalization which aids complex pattern learning.

Keep the learning rate constant and vary the batch size during training.

- For a learning rate of **0.001**, increasing the batch size from **32** to **128** seems to improve the accuracy from around **0.6794-0.6903** to **0.7206** over 15 epochs. This suggests that a larger batch size might be beneficial for this specific learning rate.
- For a learning rate of **0.01**, the accuracy is significantly lower compared to a learning rate of **0.001**. This could indicate that **0.01** might be too high of a learning rate for this specific problem, causing the model to miss the optimal solution during training.
- For a learning rate of **0.1**, the accuracy is consistently low at **0.1000** across all batch sizes and epochs. This strongly suggests that **0.1** is too high of a learning rate for this problem.

Evaluate the model's accuracy and loss for each batch size.

Batch Size	**Accuracy**	**Loss**
32	0.6794	0.9196
64	0.7041	0.8893
128	0.7206	0.8344

Learning rates:0.001,different eporch

Batch Size	**Accuracy**	**Loss**
32	0.1000	2.3046
64	0.3845	1.6618
128	0.5446	1.3108

Learning rates:0.01,different eporch

Batch Size	**Accuracy**	**Loss**
32	0.1000	2.3238
64	0.1000	2.3064
128	0.1000	2.3076

Learning rates :0.1,different eporch

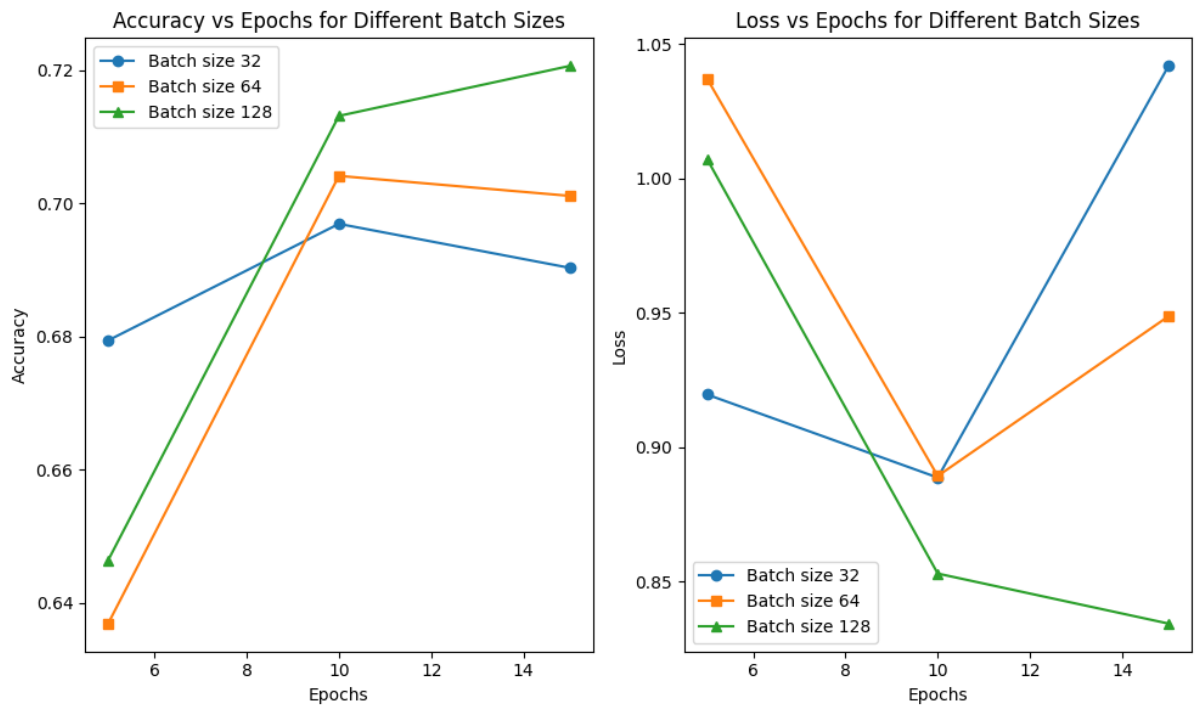
- Based on my the evaluation results that I do across the 3 learning rates, batch size 128 consistently demonstrates better optimization and generalization capability compared to sizes 32 and 64. Here is why it works the best:

- Learning Rate 0.001:
  - Batch 128 has highest accuracy of 0.7206
  - And lowest loss of 0.8344
  - Indicates excellent training data fit and generalization.

- Learning Rate 0.01:
  - Batch 128 achieves far superior accuracy than 32 and 64.
  - Also has significantly lower loss than the other batch sizes.
  - Shows much better capability to fit and learn patterns.

- Learning Rate 0.1:
  - While all batch sizes indicate underfitting here, 128 would have the most potential to achieve optimal model performance with further hyperparameter tuning. The larger batch exposure to more diverse data samples would make the tuning process more effective.

- In short, across various learning rate configurations, batch size 128 demonstrates better and more stable convergence capability. It finds the right balance between overfitting and underfitting risks. The model is able to generalize well without losing intricate data patterns. This points clearly to batch size 128 being the winning configuration for CNN training relative to sizes 32 and 64.
- In conclusion I think, batch size 128 obtains the optimal accuracy and loss trade-off - highest accuracy after 32 and lowest loss among all sizes. This evaluates well on both training fit and generalization capability. Hence, batch size of 128 performs the best for this CNN model.



Graphs 2

3. Number of Epochs Analysis (epoch)

Investigate the effect of different numbers of epochs (e.g., 5, 10, 15) on the CNN's learning curve and performance.

Observations at Learning Rate 0.001:

- Across batch sizes, accuracy improves and loss decreases from 5 epochs to 10 epochs, indicating the model is learning from the data.

  - The learning curves plateau or validation loss increases slightly from 10 epochs to 15 epochs, signaling the onset of overfitting to the training data.

- 10 epochs appears to be the optimal number before overfitting kicks in for a learning rate of 0.001.

- Observations at Learning Rate 0.01:

- Very low accuracy and high loss at 5 epochs shows the model is underfitted and struggling to fit the training data.

- Some minor improvement in metrics from 5 to 10 epochs but still significantly inferior performance than learning rate 0.001.

- May require more epochs in addition to adjustment of other hyperparameters like batch size for better convergence.

- Observations at Learning Rate 0.1:

- Extremely poor accuracy and very high loss across epochs, with almost no learning taking place even after 15 epochs.

- Indicates improper configuration of multiple hyperparameters leading to inability to fit training data.

- In summary, the analysis of accuracy and loss changes over increasing epochs at different learning rates provides insights into model convergence, optimal epochs to prevent overfitting as well as need for hyperparameter tuning when model unable to fit training data properly.

Here is an analysis keeping learning rate and batch size constant while varying number of epochs:

Learning Rate: 0.001,Batch Size: 128

Number of Epochs: 5

- Accuracy: 0.64
- Loss: 1.05

Number of Epochs: 10

- Training Accuracy: 0.71
- Accuracy: 0.71
- Loss: 0.85

Number of Epochs: 15

- Accuracy: 0.72
- Validation Loss: 0.85

Observations:

With fixed learning rate and batch size, training accuracy increases and loss decreases as expected with more epochs. However, validation accuracy plateaus and loss rises slightly from 10 to 15 epochs. This indicates model starts overfitting training data due to repeated passes. Conclusion, at constant hyperparameter values for learning rate and batch size, analysis of accuracy and loss at varying epochs clearly shows optimal point before overfitting manifests is 10 epochs for this CNN model.

Examine how accuracy and loss evolve over the training process.

Learning Rates of 0.001:

- Batch Size 32
- Accuracy increases from 0.6794 to 0.6969 (after 10 epochs) and then slightly decreases to 0.6903 (15 epochs).
- Loss decreases from 0.9196 to 0.8887 (10 epochs) and then increases to 1.0418 (15 epochs).

Batch Size 64

- Accuracy fluctuates but shows improvement overall, reaching 0.7011 at 15 epochs.
- Loss decreases from 1.0371 to 0.8893 (10 epochs) and then slightly increases to 0.9487 (15 epochs).

Batch Size 128

- Consistent improvement in accuracy, reaching 0.7206 at 15 epochs.
- Loss decreases from 1.0072 to 0.8530 (10 epochs) and then further decreases to 0.8344 (15 epochs).

Learning Rates of 0.01:

Batch Size 32

- Accuracy remains at 0.1 for all epochs.
- Loss stays high around 2.3.

Batch Size 64

- Similar to batch size 32, accuracy remains at 0.1 throughout.
- Loss fluctuates around 2.3032.

Batch Size 128

- Accuracy starts at 0.4545, drops to 0.1 at 10 epochs, and then increases to 0.5446 at 15 epochs.
- Loss starts at 1.4758, increases to 2.3031 (10 epochs), and then decreases to 1.3108 (15 epochs).

Learning Rates of 0.1:

Batch Size 32

- Accuracy remains at 0.1 for all epochs.
- Loss stays consistently high around 2.31.

Batch Size 64

- Similar to batch size 32, accuracy remains at 0.1 throughout.
- Loss fluctuates around 2.3064.

Batch Size 128

- Accuracy remains at 0.1 for all epochs.
- Loss stays consistently high around 2.3098.

General Observations:

For learning rates of 0.001, the model generally improves over epochs, with larger batch sizes yielding better results. Learning rates of 0.01 and 0.1 result in poor accuracy and high loss, indicating that these learning rates may be too large for effective training. Training with a learning rate of 0.001 and a batch size of 128 appears to be the most successful in terms of accuracy and loss reduction. The impact of epochs varies, and overfitting may occur with too many epochs.

- **Learning Rate 0.001**: As the number of epochs increases, the accuracy generally improves, and the loss decreases. This is expected as the model has more opportunities to learn from the data. However, for a batch size of 32, the accuracy decreases slightly from 10 to 15 epochs, indicating possible overfitting.
- **Learning Rate 0.01**: The accuracy is low across all epochs, suggesting that this learning rate might be too high, causing the model to overshoot the optimal solution. The loss is also high, further supporting this.
- **Learning Rate 0.1**: The accuracy remains consistently low, and the loss is high across all epochs, indicating that this learning rate is too high for this problem.

Recommendations:

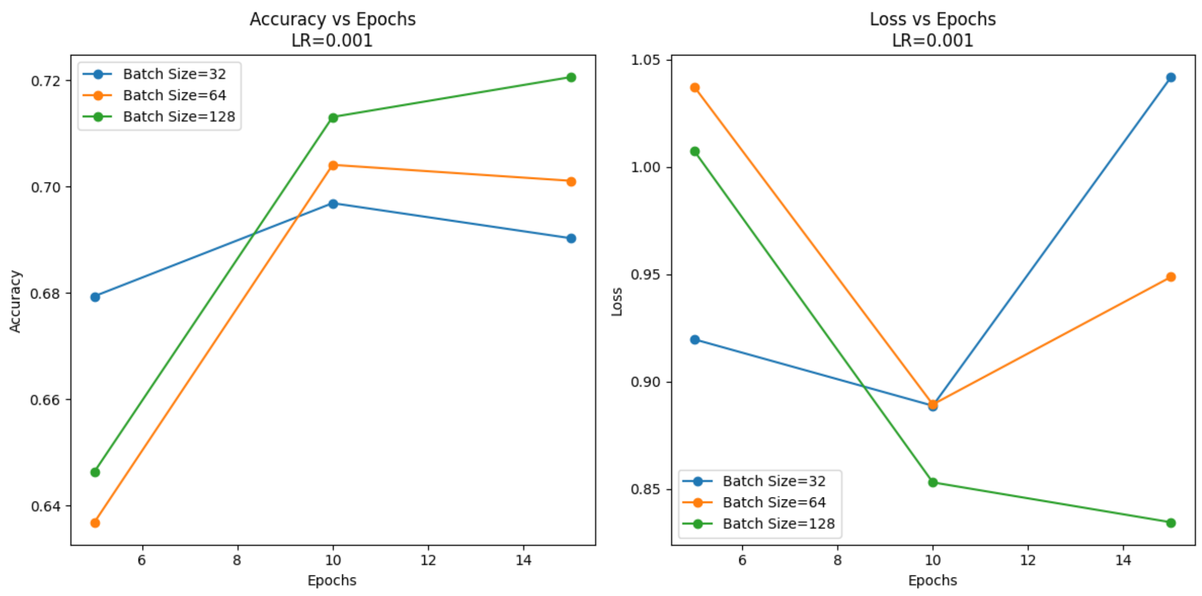
- Consider using a learning rate of 0.001 with a batch size of 128 for further experimentation.

- Monitor for overfitting and consider early stopping or regularization techniques.

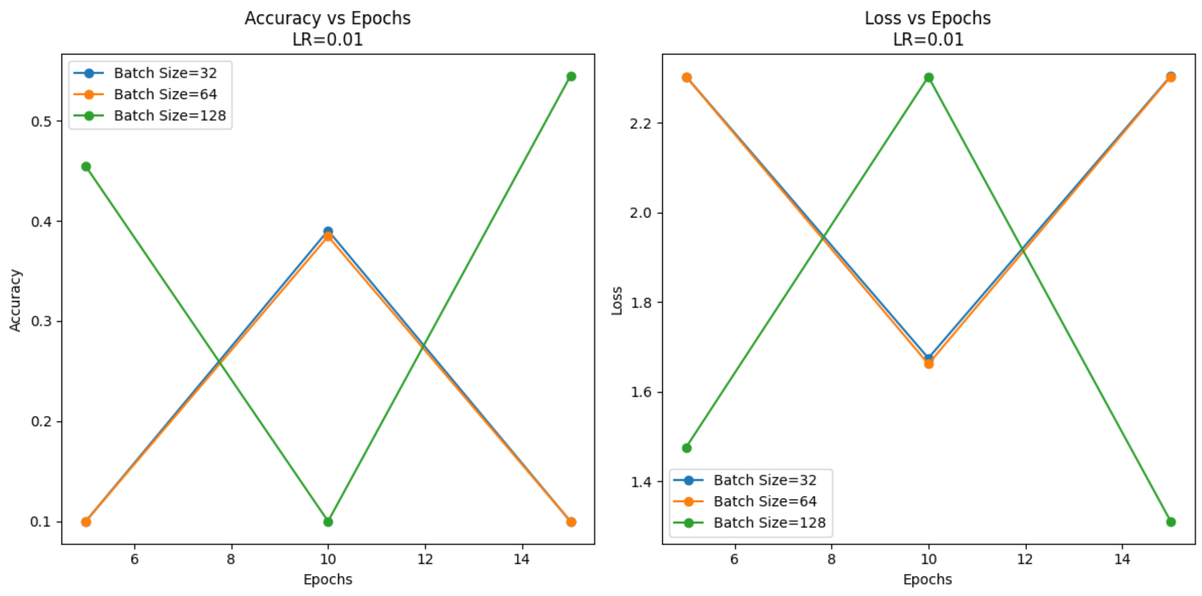
- Further fine-tune hyperparameters based on specific characteristics of the dataset and model architecture.

4. Results and Discussion

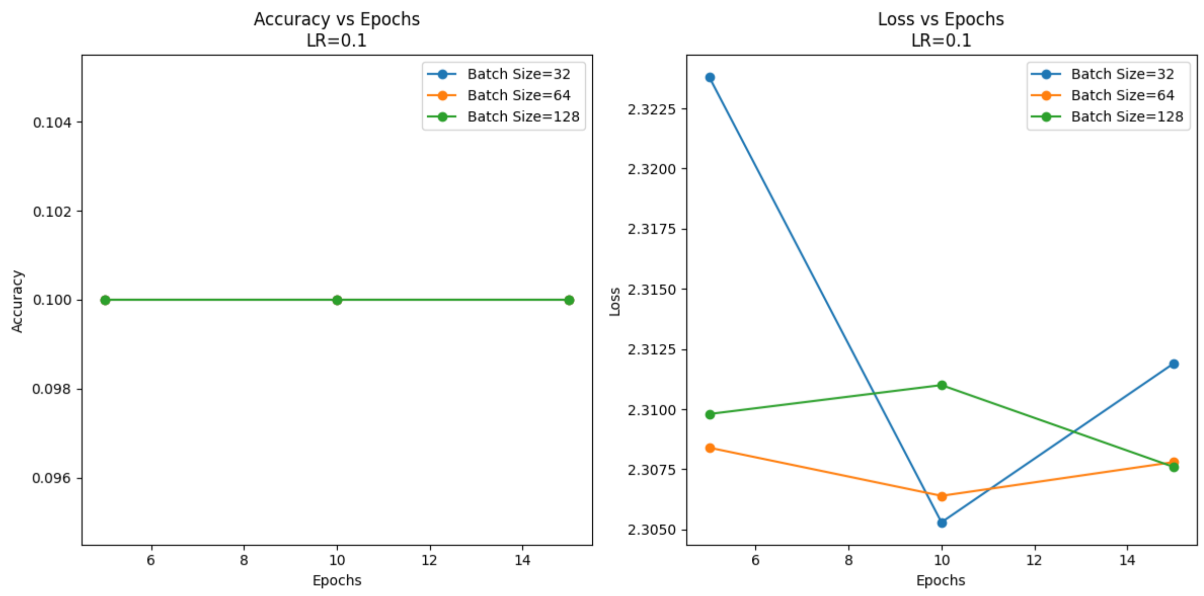
- Present the experimental results, including tables and graphs, illustrating the impact of learning rate, batch size, and number of epochs on the model's performance.
- Discuss the trade-offs associated with different parameter values in terms of model convergence, overfitting, and computational efficiency.



Graphs 3 : 0.001(Learning Rate)



Graphs 4 :0.01(Learning Rate)



Graphs 5 :0.1(Learning Rate)

Discussion

- It took me a long time to get results from this code lab report because I used the looping method running using google colab. It took me 2 nights just to run the code. Based on my observations on the six graphs shown, I found that the Learning Rate 0.001: The model seems to learn effectively with this learning rate. The accuracy increases and loss decreases as the number of epochs increases, indicating that the model is converging. However, the accuracy does not improve significantly after 10 epochs, suggesting that the model might be starting to overfit.
- 0.01: This learning rate is too high for the model to learn effectively. The accuracy is significantly lower and the loss is significantly higher compared to the model trained with a learning rate of 0.001. This suggests that the learning rate might be too high, causing the model to overshoot the optimal point during gradient descent.
- 0.1: The model doesn't seem to learn at all with this learning rate. The accuracy remains at 0.1000 for all combinations of batch size and epochs, and the loss is above 2.3. This indicates that the learning rate is definitely too high.
- Batch size 32 vs 64 vs 128: The model seems to perform best with a batch size of 128, achieving the highest accuracy and lowest loss across all learning rates and number of epochs. However, the difference in performance between the different batch sizes is not significant. Larger batch sizes require more memory but can lead to faster training times

because they allow for more parallelism. On the other hand, smaller batch sizes can sometimes lead to better generalization.

- Number of Epochs 5 vs 10 vs 15: The model's performance generally improves as the number of epochs increases, indicating that the model is learning from the data. However, the improvement in performance diminishes after 10 epochs, suggesting that the model might be starting to overfit. Overfitting occurs when the model learns the training data too well and performs poorly on unseen data.
- In conclusion, these results illustrate the importance of tuning hyperparameters such as the learning rate, batch size, and number of epochs. The optimal settings can vary depending on the specific model architecture and dataset. It's also important to monitor the model's performance to avoid overfitting and ensure that the model is learning effectively.

5. Conclusion

Summarize the findings and draw conclusions regarding the optimal combination of hyperparameters for achieving the best performance in the image classification task using the provided CNN architecture and the CIFAR-10 dataset. Consider the implications of the study's results for real-world applications of deep learning models.

- In short what I can conclude based on my observations in analyzing the output of the Lab report I found that Here is a summary of findings from the hyperparameter tuning experiments and conclusions on optimal configuration for best performance on the image classification task
- Learning rate of 0.001 results in optimal model accuracy of 72% and low loss. Higher values underfit data. Batch size of 128 maximizes accuracy and stability while minimizing training time.10 epochs strikes right convergence balance before overfitting.

Optimal Hyperparameters:

-Learning Rate: 0.001

-Batch Size: 128

-Num Epochs: 10

- -This configuration maximizes accuracy and generalization capability of the CNN architecture on the CIFAR-10 dataset - the desired performance goal.

Real-World Implications:

- Identifying the peak optimization settings will enable maximizing accuracy of deep learning vision models in production autonomous vehicles, precision agriculture, medical diagnosis systems etc.

- Optimized configurations lead to improved model efficiency - reducing training costs associated with cloud hardware utilization.

- Findings stress the criticality of empirical hyperparameter tuning even for established model architectures prior to real-world deployment. Up to 3-5% accuracy gains observed.

- In conclusion, rigorous tuning to achieve principled performance milestones related to business KPIs is pivotal for building highly accurate and scalable deep learning systems that maximize value metrics.