Q1.) Analyze the variations in loss and testing accuracy over epochs to assess

model performance.

By 4 epochs , the training accuracy increased to a number greater than 95 percent and ended at almost 99% indicating a really good model over training data. The losses tend to come down from 2.99 to 2.3 by 10 epochs

Q2.) Write your observations on Overfitting issues in this project. How does the CNN

handle this issue? Will the Overfitting further reduce if you use another dropout layer (say with rate0.3) after the Convolution layers?

Convolutional Neural Networks (CNNs) can suffer from overfitting, especially when the model capacity is too high relative to the size of the training dataset or when the dataset itself is noisy or not diverse enough. Overfitting occurs when the model learns to memorize the training data rather than generalize patterns from it, leading to poor performance on unseen data. Here are some common techniques to address overfitting in CNNs:

Dropout: Introducing dropout layers during training, which randomly deactivate a certain percentage of neurons in the network during each training iteration. This prevents the network from relying too much on specific neurons and encourages it to learn more robust features.

Regularization: Applying penalties to the model's weights during training to prevent them from becoming too large. L2 regularization (weight decay) and L1 regularization are common techniques used to achieve this. These penalties discourage the model from fitting the noise in the training data.

Reduce Model Complexity: Decreasing the capacity of the model by reducing the number of layers, the number of neurons in each layer, or the size of the filters. This helps prevent the model from fitting the training data too closely and encourages it to learn more generalizable features.

Batch Normalization: Normalizing the activations of each layer within a mini-batch during training. This helps stabilize the training process and reduces the risk of overfitting by reducing internal covariate shift.

Overfitting occurs when the model learns to memorize the training data rather than generalize well to unseen data. In this project, overfitting is evident from the increasing validation loss while the training loss keeps decreasing. Overfitting can lead to a high training accuracy but poor performance on unseen data.

Structure Property:

The structure property refers to the hierarchical organization of visual information in images. In natural images, there are often recurring spatial patterns and structures that are meaningful for recognition tasks. For example, in an image of a cat, there are structural elements such as edges, corners, and textures that collectively define the visual appearance of the cat.CNNs leverage the structure property by using layers such as convolutional layers and pooling layers to capture and extract hierarchical features from images. Convolutional layers apply filters to detect low-level features like edges and textures, while pooling layers aggregate spatial information to capture higher-level structures. By exploiting the hierarchical structure property of images, CNNs can effectively learn to represent and recognize objects at different levels of abstraction.

Invariance Property:

The invariance property refers to the ability of a model to recognize objects despite variations in their appearance due to factors such as translation, rotation, scaling, or illumination changes. In real-world scenarios, objects may appear in different orientations, sizes, or lighting conditions, yet humans can still recognize them reliably.CNNs exhibit certain degree of invariance property due to the shared weights and hierarchical feature extraction. For example, a CNN trained to recognize cats should ideally be able to identify a cat regardless of whether it's in the center or corner of the image, or whether it's facing left or right. This is because lower-level features like edges and textures are detected locally and then combined hierarchically to represent more complex patterns, leading to some level of robustness against variations in object appearance.Data augmentation techniques, such as rotation, translation, and flipping, can further enhance the invariance property of CNNs by exposing the model to a wider range of variations during training. Additionally, techniques like dropout and batch normalization can also help improve the model's generalization performance by reducing sensitivity to specific variations in the training data.Overall, the invariance property is crucial for ensuring that CNNs can generalize well to unseen variations in the input data, making them effective for tasks like image classification in real-world scenarios.