L07 Chihuahua or Muffin with CNN Reflection

The Convolutional Neural Network (CNN) architecture used in this lab consisted of three convolutional layers, each followed by a ReLU activation function and max-pooling operation, and two fully connected (dense) layers. The convolutional layers were responsible for learning spatial hierarchies of features from the input images, such as edges, textures, and more complex patterns. The fully connected layers acted as a classifier by mapping the extracted features to the final output classes (Chihuahua or Muffin).

Unlike traditional neural networks (NNs) used in the previous workshop, CNNs excel at processing grid-like data, such as images. Traditional NNs treat input features independently, which makes them less effective at capturing spatial relationships in images. In contrast, CNNs leverage local receptive fields and weight sharing to recognize patterns regardless of their position in the image, which reduces the number of parameters and improves efficiency.

The CNN achieved a final validation accuracy of 93.33%, significantly higher than the performance of the traditional NN used previously. The training accuracy also increased steadily across epochs, demonstrating the model's capacity to learn features from the dataset effectively. Interestingly, during the early epochs, the model struggled with images that had features resembling both Chihuahuas and muffins, leading to some misclassifications. For example, images with complex textures, like a Chihuahua's fur resembling muffin toppings, were occasionally misclassified. These patterns indicate that

CNN's ability to generalize improves with training. Additionally, the use of data augmentation, such as random rotations and flips, contributed to the model's robustness by exposing it to varied data distributions.

One challenge encountered during the lab was ensuring that the data directory structure matched the code's expectations. Initially, missing, or misaligned directories caused runtime errors. This was resolved by verifying the directory paths and using the `os` module to inspect file contents programmatically. Another challenge was tuning hyperparameters such as the learning rate, batch size, and number of epochs. The model occasionally overfits the training data when trained for too many epochs or with an excessively high learning rate. Introducing dropout layers and experimenting with learning rates between 0.001 and 0.0005 mitigated this issue. Lastly, the warning about excessive `num_workers` in the `DataLoader` was addressed by reducing the value to 2, which prevented potential slowdowns or freezes.

This type of image classification model has numerous real-world applications. Examples include healthcare, such as diagnosing diseases from medical imaging (e.g., identifying cancerous tumors in X-rays or MRIs), security applications like recognizing faces or objects in surveillance footage, retail tasks for categorizing products or detecting damaged goods in warehouses, and autonomous vehicles that identify pedestrians, traffic signs, and other vehicles. For this specific task, the ability to distinguish visually similar objects (e.g., Chihuahuas and muffins) could be extended to areas like quality control, where subtle differences in appearance determine product categories.

This lab was an exciting opportunity to deepen my understanding of CNNs and their applications. The experience highlighted the importance of architectural design, data preprocessing, and model evaluation. By addressing challenges systematically and reflecting on ethical considerations, I gained a deeper understanding of how to develop robust and responsible machine learning solutions. This project has reinforced my passion for exploring the intersection of machine learning and real-world problem solving.