Tecnológico de Monterrey

First Challenge

Name and Registration Number:

Bárbara Paola Alcántara Vega	A01799609
Ulises Orlando Carrizalez Lerín	A01027715
María José Soto Castro	A01705840
Tomás Pérez Vera	A01028008
Mónica Monserrat Martínez Vásquez	A01710965

TC3006C.101 - Inteligencia artificial avanzada para la ciencia de datos I (Gpo 101)

Jose Antonio Cantoral-Ceballos

Date:

03/09/2025

Index

Introduction

This report details the data preparation process for the CIFAR-10 dataset, an essential activity for establishing a strong foundation in the PyTorch artificial intelligence framework. The primary objective was to acquire a formal understanding of how to efficiently and correctly prepare raw data for a machine learning model. By analyzing and making decisions about the CIFAR-10 image data, this project demonstrates the critical steps involved in transforming a dataset into a format suitable for a deep learning model. The following sections will provide a comprehensive explanation of the actions taken within the assigned Jupyter Notebook, offering clarity on the methodological and strategic decisions made during data processing.

Dataset Description and Loading

The dataset assigned for this activity is CIFAR-10. It is a well-known benchmark dataset in the field of computer vision, consisting of 60,000 color images, each sized at 32x32 pixels. The dataset is divided into 10 mutually exclusive, pre-labeled classes, making it ideal for supervised learning tasks. The data is pre-split into a training set of 50,000 images and a test set of 10,000 images. Each image has three color channels (Red, Green, and Blue).

To acquire the dataset, PyTorch's torchvision.datasets.CIFAR10 class was used. This class, part of the torchvision library, simplifies the process of downloading and loading common vision datasets. By simply setting the download=True parameter, the class automatically checks for the dataset's presence in a specified directory. If the data is not found, it is automatically fetched from a verified online source, ensuring data integrity and saving time on manual downloading and organization. Due to this standardized, clean, and pre-labeled structure, additional data cleaning steps are not required for this activity.

Data Transformation

The first and most crucial step in preparing the CIFAR-10 data was to convert the raw images into a format that PyTorch could process. This was accomplished by applying the initial transformation transforms.ToTensor(). This single function performs three essential and simultaneous changes to the dataset, making it suitable for subsequent deep learning operations.

First, the dimension transpose occurs. A typical image is structured with dimensions for height, width, and color channels, represented as (H, W, C). The ToTensor() transformation rearranges this to (C, H, W), which is the standard format for PyTorch tensors.

Second, the data type conversion takes place. The raw image data, which is typically stored as 8-bit unsigned integers (uint8), is converted to 32-bit floating-point numbers (float32). This is necessary because deep learning models primarily operate on floating-point data.

Third, a critical value normalization is applied. Before this transformation, the value for each color channel in every pixel ranges from 0 to 255. Once the images are converted into PyTorch tensors, the values are automatically normalized to a floating-point range between 0.0 and 1.0. This normalization is vital for improving model performance and stability, as it helps prevent large input values from disproportionately affecting the model's training process.

The initial transformation was already included in the assigned Jupyter Notebook. It was, however, our responsibility to understand its function so that we could continue to apply further transformations to the data.

Per-Channel Statistics and Standardization

After converting the raw images to PyTorch tensors, the next step was to standardize the values of each color channel to prepare the data for more efficient model training. This was achieved by first calculating the mean and standard deviation for each channel across the entire 50,000-image training dataset. This calculation involved iterating through every image and accumulating the mean and squared values for each of the three color channels (Red, Green, Blue). Using these accumulated values, a final calculation was performed to determine the precise mean and standard deviation for each channel. The results obtained for the normalized data were a mean of approximately tensor([-4.5123e-06, -2.3405e-06, -6.1984e-08]) and a standard deviation of approximately tensor([1.0000, 1.0000, 1.0000]) for each color channel, so with that, it was confirmed that the data standardization was done in a correct way. These statistical values are critical for the next step: applying a global normalization that transforms the data so its mean is approximately 0 and its standard deviation is approximately 1, a fundamental practice that significantly improves model stability and performance during deep learning.

This process was completely done by ourselves, using the mean of each color channel of each image in the training dataset to obtain a data distribution closer to a normal distribution.

Dataset Exploration and Visualization

The notebook provided a pre-written function called show_images that allowed us to visualize random samples from the CIFAR-10 dataset. This function was an invaluable tool for us to quickly verify that the data was correctly loaded and that our transformations were working as expected. It allowed us to randomly select a few images from the training set, display them, and confirm their corresponding labels, ensuring that our data was prepared correctly before moving on to model training.

On the same sense, in order to verify the data distribution, we create two histograms in order to visualize the number of samples for each class on both, the training dataset and the test data set, obtaining:

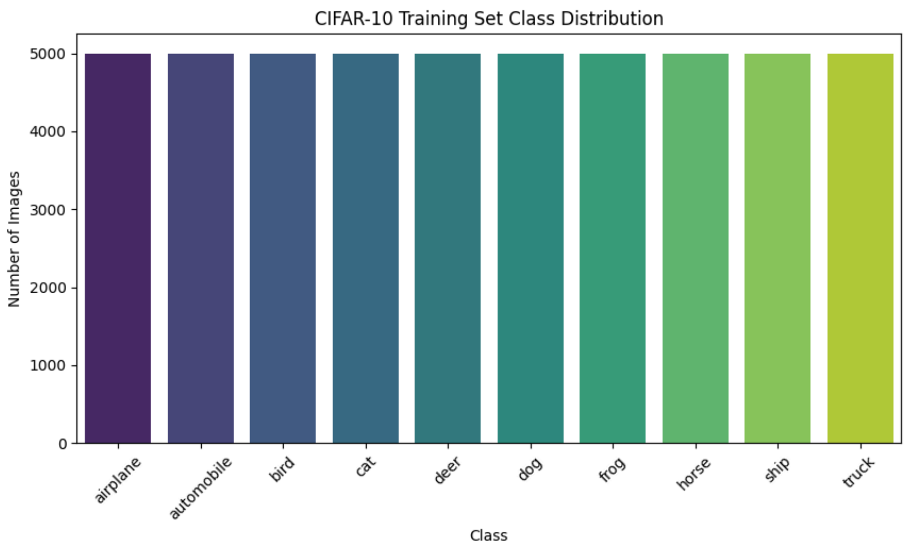

Figure 1. Class distribution on training set

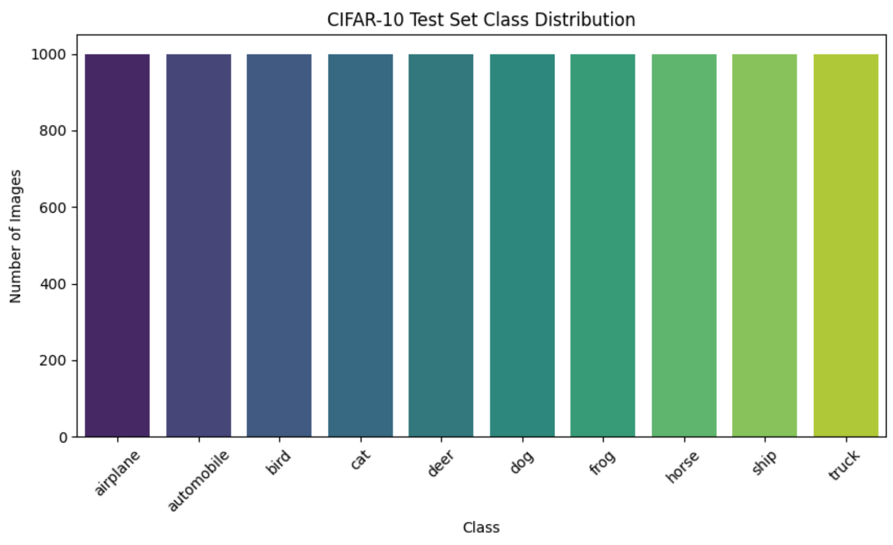

Figure 2. Class distribution on test set

Based on the histograms above, we conclude that there was a uniform distribution for each class in each data set, and also noticing that there were 50,000 samples on the training set, and 10,000 samples on the test dataset, so with that, we confirm that the CIFAR-10 data set, was already clean.

Dataset Augmentation

To enhance the model's ability to generalize and to effectively increase the number of training samples without adding new images, a series of data augmentations were applied to the dataset by ourselves. These transformations were applied on-the-fly, as part of the data loading pipeline, ensuring that a new, randomly augmented version of each image is generated every time it is accessed. The pipeline begins with transforms.RandomHorizontalFlip(p=0.5), which randomly flips the image to make the model invariant to object orientation. This is followed by transforms.RandomCrop(32, padding=4), which helps the model learn to recognize objects even when they are not perfectly centered. transforms.ColorJitter(0.2, 0.2, 0.1) then randomly alters the image's brightness, contrast, and saturation, which makes the model more robust to lighting variations. After these augmentations, the image is converted into a PyTorch tensor with transforms.ToTensor(), and finally, the entire pipeline is concluded with normalization, which applies the previously calculated mean and standard deviation to standardize the data, making it ready for training.

Conclusion

In summary, our work on the CIFAR-10 dataset was a valuable learning experience, providing us with a deeper understanding of the challenges and best practices in data preparation. We successfully navigated several challenges, including memory management issues when attempting to load the entire dataset at once, which led us to adopt a more efficient batch processing approach. We also had to be mindful of the different tensor formats used by PyTorch ([C, H, W]) and Matplotlib ([H, W, C]), a critical detail for successful data visualization. A particularly important challenge was correctly verifying our normalization calculations, as we did not proceed until our post-normalization mean and standard deviation were nearly [0,0,0] and [1,1,1] respectively, ensuring our data was truly standardized. Finally, we learned that careful consideration of the data is necessary for choosing augmentations that are appropriate for small images like those in CIFAR-10, as excessive transformations can degrade the semantic content.

Through this process, we gained several key learnings:

- The Importance of Normalization: We saw firsthand how much data values can vary and the positive impact of standardizing them for model training.
- Batch Processing: We learned a crucial skill for handling large datasets efficiently without overwhelming system memory.

- Reproducibility: The act of setting a fixed random seed reinforced the importance of creating reproducible machine learning experiments.
- Data Validation: We understood that every step of data transformation must be validated to ensure the integrity and quality of the final dataset.

Overall, this activity provided us with a solid foundation in the data preparation workflow, which we now know is a critical prerequisite for building and training effective deep learning models.

References

- CIFAR-10 and CIFAR-100 datasets. (1992). https://www.cs.toronto.edu/~kriz/cifar.html