Convolutional Neural Networks using PyTorch

Goal: You are given an architecture of a CNN and must implement it using PyTorch to classify satellite images from the EuroSAT dataset. You will also explore and improve this model using your understanding of deep learning and PyTorch.

Disclaimer: System configurations, GPU availability, and compute limits may vary depending on your environment (e.g., Colab, local machine, university cluster). Adjust the batch size, training epochs, or image size accordingly. We recommend using Google to download the EuroSAT dataset. A different version of PyTorch may not include the EuroSAT dataset.

CNN architecture: The CNN is composed of two convolutional layers, two max pooling layers, followed by fully connected layers. A batch normalization, ReLU activation and Max pooling layers follow each convolutional layer. Additionally, the second convolutional layer features a dropout layer. Only the first fully connected layer is followed by a dropout layer. The CNN architecture is shown in Figure 1.

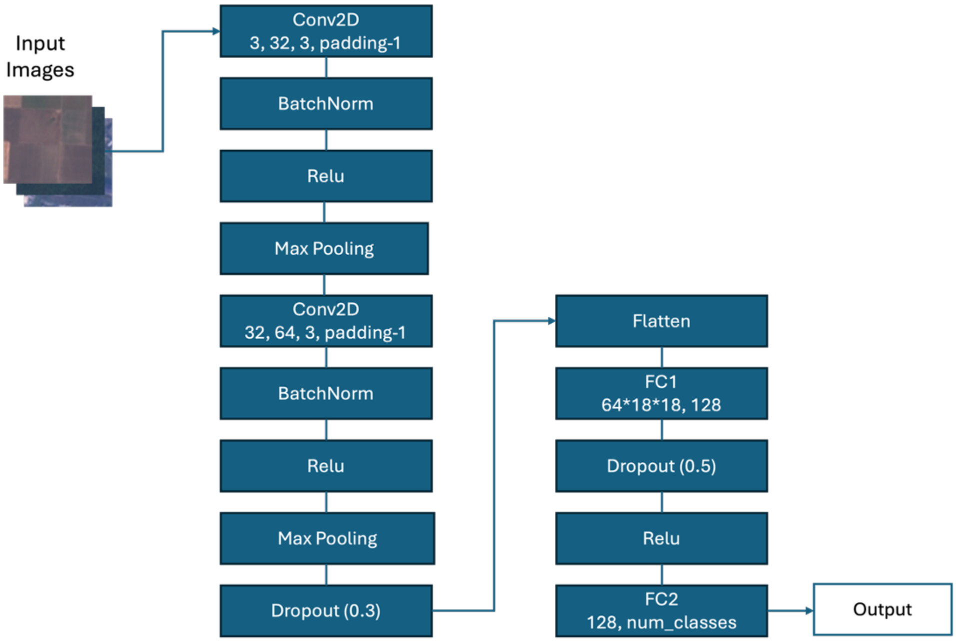

Figure 1. The CNN architecture

Your Task

- Implement the full CNN model based on the given architecture diagram
    - You must define each layer explicitly in *__init__* and write your own *forward()* method

  - You are **not** allowed to use *nn.Sequential*
  - You should verify that the output shapes match those shown in the diagram below.
- Implement Compete the *train()* and *evaluate()* functions
- Tuning hyperparameters (learning rate, batch size, optimizer, dropout, etc.)
  - Exploring regularization and data augmentation strategies
  - Tracking input/output shapes at each layer
- Observing and interpreting validation behavior
- Use visualisations to help interpret your model. This could include:
  - Plotting training and validation loss
  - Viewing predictions vs. ground truth
  - Observing patterns in the data
  - Any additional plot or insight that helps you explain your model's behavior

Experiment and Reflection

- What choices affect the model's accuracy?
- How do you determine whether the model is overfitting or underfitting?
- What happens when you add or remove components, for example, dropout or batch normalisation?
- What layer causes the highest number of parameters?
- What happens if you remove all pooling layers in a CNN?
- In a CNN block, can you arrange the layers (i.e. Conv, ReLU, BatchNorm, Dropout) in any order? Explain your answer

Submition

- You should submit your .ipynb notebook, clearly showing:
  - CNN model implemented from the given diagram
  - Code used to train and evaluate
  - Visualizations and/or printed shapes
  - Final reflections at the bottom by answering the questions in **Experiment and Reflection**