**TASK:**

Submit a research-style report, using the provided Jupyter Notebook, in which you build a neural network to investigate three small experiments and present/discuss the results.

The main task is to develop a deep convolutional neural network to perform multi-class classification. The dataset for this assignment is the CIFAR10 dataset (link), which is also available from the torchvision library in PyTorch (link). The dataset is made up of 32x32 colour images, which are split into a training set of 50,000 images and a test set of 10,000 images. The images have labels (one label per image) from 10 classes: 1) aeroplane; 2) automobile; 3) bird; 4) cat; 5) deer; 6) dog; 7) frog; 8) horse; 9) ship; 10) truck.

The network should be trained to predict the labels using the training data, and its generalisation performance is evaluated using the test data. You will be asked to perform three experiments in which you analyse the influence of different methods on the performance and functionality of your neural network

**Baseline:**

In this assignment, you need to implement a deep convolutional neural network, which you will later build upon.

The baseline version of your CNN must have:

* an input, three convolutional hidden layers, then two fully connected layers, where the second fully connected layer is 10-dimensional and provides the output classifications of your network.
* Each convolutional layer block must include pooling and a non-linear activation functionof your choice.
* You can use PyTorch (recommended), or any other deep learning frameworks (e.g. TensorFlow) with automatic differentiation.
* The neural network should be trained to classify the images from the CIFAR10 dataset.
* You can use the modules available in packages like PyTorch to load the dataset (e.g. in PyTorch, you can use torchvision.datasets.CIFAR10) and to build the layers in your model.
* The assignment is split into three experiments:

**Ex1:**

*Investigate the effect that the learning rate has on your model’s performance, i.e. it’s classification accuracy, or error rate.*

* Choose an appropriate split of the training data to include a subset used for validation.
* Compare the performance of your model for 5 different learning rates. The learning rates you choose must be high enough to allow the performance on the validation data to saturate, and they must be low enough to prevent your model from becoming unstable.
* For each learning rate, run the model at least 5 times, using different seeds to initialise any random number generators (e.g. for weight initialisation) such that each run of the model produces different results. You can then use the average of the model runs to plot the model’s mean performance with respect to the training episodes for the training and validation data (these plots are also called learning curves).
* Using these results, design a learning rate scheduler that reduces the learning rate during training, again running the model at least 5 times to build up a picture of its average performance.
* Plot the learning curves for this new version of the model.
* Compare performance on the test data, and in terms of the generalizability gap between training and validation data, between the learning rate scheduler model and the best performing model that didn’t use a learning rate scheduler (out of the five you tested previously).

**Ex2:**

*Investigate the impact that regularisation has on the performance of your model.*

* **First**, split the training data into two halves. One half will be used for training, the other half for validation.
* Then implement Dropout in the fully connected layers of your model. Investigate how the dropout rate affects average performance on the test data, and the generalisation gap between the training and validation data.
* Choose 5 Dropout rates, one of which must have a value of zero (i.e. no Dropout). Remember to run each version of the model 5 times to compute the mean performance.
* **Second**, swap around your two datasets, so that the training data is now used for validation, and the validation data is now used for training, and investigate how your model performs in a transfer learning task.
* To do this, freeze all of the weights in the convolutional layers (i.e. prevent them from being updated during optimisation), re-initialise the weights in the fully connected layers, and retrain the two fully connected layers.
* Apply only two models to the transfer learning task: one without Dropout, and the other using a non-zero Dropout rate that gave the best average performance.
* How does the average performance of these retrained models compare with the average performance of the models in which all layers were trained together? Refer to their average performance on the test data, as well as the generalisation gap between the training data and validation data.
* Provide learning curve plots to back up your analyses of these models.

**Ex3:**

*Investigate the quality of gradient flow through your network.*

* **First**, for the model that does not use Dropout, record gradients for the optimised parameters (i.e.dL/dw) in each layer of your model.
* Record the gradients over the first 5 episodes of training, and build a separate record of gradients over the final 5 episodes of training.
* Produce plots to show how the mean and standard deviation of gradients change as a function of layer number, for both the beginning (first 5 episodes) and the end (final 5 episodes) of training.
* **Second**, using the best performing, non-zero Dropout rate from Experiment 2, perform the same calculations to compute the mean and stand deviation of the gradients for all layers, and over the first and final 5 episodes of training.
* Does Dropout affect gradient flow in your model? If so, how are the gradients affected?
* **Third**, without including Dropout, add batch normalisation to each hidden layer of your model.
* Again, compute and plot the mean and standard deviation of the gradients as you had done before.
* How is gradient flow affected by batch normalisation in your model?
* **Fourth**, using learning curves for training and validation data, show how batch normalisation affects the performance of your model, referring to its average performance on the test data as well as to the generalisation gap between the average training and validation learning curves.

**1. Abstract/Introduction (15 MARKS)**

Use the next Markdown/text cell to write a short introduction to your report. This should include:

- a brief description of the topic (**image classification**) and of the dataset being used (**CIFAR10 dataset**). (2 MARKS)

- a brief description of how the CIFAR10 dataset has aided the development of neural network techniques, with examples. (3 MARKS)

- a descriptive overview of what the goal of your report is, including what you investigated. (5 MARKS)

- a summary of your major findings. (3 MARKS)

- two or more relevant references. (2 MARKS)

**2. Methodology (31 MARKS)**

Use the next cells in this Methodology section to describe and demonstrate the details of what you did, in practice, for your research. Cite at least two academic papers that support your model choices. The overarching principle of writing the Methodology is to provide sufficient details for someone to replicate your model and to reproduce your results, without having to resort to your code. You must include at least these components in the Methodology:

2.1 . Data (7 MARKS)

Describe the dataset, including how it is divided into training, validation, and test sets. Describe any preprocessing you perform on the data, and explain any advantages or disadvantages to your choice of pre-processing.

2.2. Architecture (17 MARKS)

Describe the architecture of your model, including all relevant hyperparameters. **The architecture must include 3 convolutional layers followed by two fully connected layers**. Include a figure with labels to illustrate the architecture.

2.3 Loss function (3 MARKS)

Describe the loss function(s) you are using, and explain any advantages or disadvantages there are with respect to the classification task.

2.4 Optimiser (4 MARKS)

Describe the optimiser(s) you are using, including its hyperparameters, and explain any advantages or disadvantages there are to using that optimizer.

**3. Experiments (24 MARKS)**

Describe how you conducted each experiment, including any changes made to the baseline model that has already been described in the other Methodology sections. Explain the methods used for training the model and for assessing its performance on validation/test data.

3.1 Experiment 1 (8 MARKS)

* Describe how you went about conducting experiment 1 here

3.2 Experiment 2 (8 MARKS)

* Describe how you went about conducting experiment 2 her

3.3 Experiment 3 (8 MARKS)

* Describe how you went about conducting experiment 3 here

**4. Results - (55 MARKS)**

Use the Results section to summarize your findings from the experiments. For each experiment, use the Markdown/text cell to describe and explain your results, and use the code cell (and additional code cells if necessary) to conduct the experiment and produce figures to show your results.

4.1 Experiment 1 (17 MARKS)

* Write up results for experiment 1

4.2 Experiment 2 (19 MARKS)

* Write up results for experiment 2

4.3 Experiment 3 (19 MARKS)

* Write up results for experiment 3

# **5. Conclusions and Discussion (25 MARKS)**

In this section, you are expected to:

* briefly summarise and describe the conclusions from your experiments (8 MARKS).
* discuss whether or not your results are expected, providing scientific reasons (8 MARKS).
* discuss two or more alternative/additional methods that may enhance your model, with scientific reasons (4 MARKS).
* Reference two or more relevant academic publications that support your discussion. (4 MARKS)

**6. References (instructions)**

* Use the cell below to add your references. A good format to use for references is like this:
* [AB Name], [CD Name], [EF Name] ([year]), [Article title], [Journal/Conference Name] [volume], [pa