**Project 1: Implement a CNN-Based Image Classifier**

Earlier, you trained CNNs on datasets of small images that were available inside torchvision. For example, MNIST, FashionMNIST and CIFAR10 . Though a good starting point, with, these assignments lacked the complexity to tackle real-world problems. Also, you were given most of the code and only had to add small parts to complete the assignment.

In this project, you will write most of the code yourself ( just refer to all the course lessons and the code we have shared).

**Task**

**Implement a 3-class Image Classifier, using a CNN**, on a dataset, which consists of 1000 images each of **panda, cat and dog**. We took the dataset from [Kaggle](https://www.kaggle.com/ashishsaxena2209/animal-image-datasetdog-cat-and-panda), ande split it into 80:20 ratio for train:validation. To download, **[click here](https://www.dropbox.com/sh/n5nya3g3airlub6/AACi7vaUjdTA0t2j_iKWgp4Ra?dl=1)**.

**Build your own network. Do not use any pre-trained network/weights for this task.**

**The Purpose of This Project is Two-fold:**

1. The first is to get your hands dirty. training a neural network from scratch. Only then will you you understand how to build and train a real network on real data. You will implement everything this time - right from getting the data to analysing the final results.
2. The second is to reinforce the use of the complete training pipeline that we suggested in our previous lecture.Try to stick to this pipeline every time you work on a new problem.

**Download these two notebooks:**

1. Data\_and\_Pipeline\_Check.ipynb [30 Points]

**This notebook is mostly for your own learning**. Also, we have provided some code that helps in data exploration as well as checking the code in your training pipeline, where you train a simple model on sample data. Check for these common sources of error:

**This notebook is mostly for your own learning** and we have provided some code which goes over the first few steps of data exploration and checking the code in your training pipeline by training a simple model on a sample data. Common sources of errors which you should check are -

* input and output shape for network,
* incompatible shapes among subsequent layers,
* loss function and optimizer related,
* any image transforms,
* plotting of curves

2. Training\_from\_scratch.ipynb [70 Points]

**This notebook has to be completed by you.** You have to train a network on the full data. Experiment with different network parameters to **achieve >85% validation accuracy**. You may not get 85%on your first attempt. To achieve that, experiment with:

* Number of layers
* Parameters inside the layers
* Optimizers and learning rate schedulers [You can even get good results without a learning rate scheduler]
* Regularization techniques like Data augmentation, Dropout, BatchNorm
* Number of epochs

For GPU compute you can use Kaggle kernels, Google Colab or any other service of your choice.

Upload the completed assignment and submit the 2 notebooks on the portal for for feedback.

After you have completed the assignment, upload and submit the 2 notebooks on the portal so that we can check and provide feedback. **Make sure you have uploaded and submitted an executed notebook with proper loss and accuracy plot**. We can not give you feedback and further improvement suggestions without loss and accuracy plot.

However, if you got stuck before training your model and did not have these plots, you can submit it as it is and ask for help.

**Do not upload the model weight**