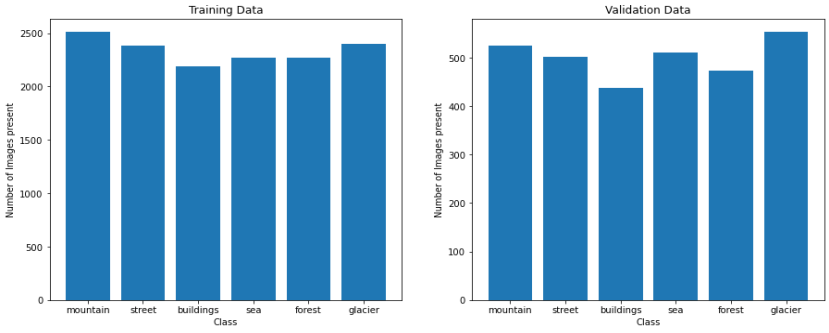
ML Hackathon Techkriti 2k21

Problem Statement: Classifying images into 6 categories (buildings, forest, glacier, mountain, sea, and street)

ML library used: TensorFlow (`!pip install tensorflow` to install tensorflow)

GPU used: NVIDIA TESLA P100

Dataset Details: Following is the distribution of data



We can see that the training data and test data are **uniformly distributed** over 6 classes. So, **Accuracy is the best metric** that can be used to test our model.

Training Details:

- Data pipeline is prepared using `tf.keras.preprocessing.ImageDataGenerator` and augmentation is added to Training data to make sure that the model is not over-fitting. (More details can be found in **training.ipynb** file)
- Pretrained models (`tf.keras.applications`) are used as base model for training and few FC layers are added to top of base model.
- **Loss** : `Categorical Cross Entropy` ; **Metric** : `Categorical Accuracy` ; **Optimizer** : `Adam optimizer`.
- **Call-backs** used : `Reduce ROn plateau,`

Model checkpoint,Early stopping

- Best model is obtained with high validation categorical accuracy and it is further used to test on test data (seg_pred,seg_test).

Results: (Go to experiments folder, more details in experiments.csv file)

Notebook	exp0	exp1	exp2	exp3	exp4	exp5	exp6	exp7	exp8	exp9	exp10	exp 11	exp12
Accuracy	0.935	0.9413	0.941	0.9417	0.935	0.9343	0.937	0.938	0.938	0.946	0.945	0.937	0.942
Run time	5458	4570	7960	5996	3271	2161	6945	7459	3799	2560	3001	2593	8291

Steps to run scripts:

- Run the **training.ipynb** file to retrain the model. This will save the model in the **./** directory as **best_model.h5** . **(on an avg 4-5min per epoch)**
- Run the **test.py** file to obtain the **test.csv** file. Also, Accuracy on test data is also printed. **(1-2min time).** Format of csv is as follows

original_label	predicted_value	result
0	0	correct
0	1	incorrect

- Run **prediction.py** to obtain **prediction.csv** file in in which the predictions column contains the name of image in the lexicographical order of images. **(1-2 min time)**
- **Experiments (in codes folder)** folder contains all the notebooks of various models and variations I tried.
- In test.csv file labels are : **{0: 'buildings', 1: 'forest', 2: 'glacier', 3: 'mountain', 4: 'sea', 5: 'street'}**
- It is suggested to run the scripts on GPU /Colab/Kaggle as required RAM and computation are high.
- Accuracy is checked from **test.csv** file.

Best Accuracy on test data : 0.945677

Base pretrained model : DenseNet201 (no freeze layers)

```
pretrained = tf.keras.applications.DenseNet201(include_top=False,
                                                weights='imagenet',
                                                pooling="avg",
                                                input_shape=[HEIGHT,WIDTH,
3])
    x = pretrained.output
    x = tf.keras.layers.Dropout(0.3) (x)
    x = tf.keras.layers.Dense(128) (x)
    x = tf.keras.layers.LeakyReLU(alpha=0.2) (x)
    x = tf.keras.layers.GaussianDropout(0.4) (x)
    outputs = tf.keras.layers.Dense(NUM_CLASSES,activation="softmax",
dtype='float32')(x)
    model = tf.keras.Model(pretrained.input, outputs)
```