**Report**

Datasets:

* SCENE-15
* Caltech-256

Model:

* RESNET

**SCENE-15**

* Contains 15 different scenes.
  + Total 4485 different images.
* The initial 8 classes were collected by Oliva and Torralba.
* Then 5 categories were added by Fei-Fei and Perona.
* finally, 2 additional categories were introduced by Lazebnik et al.

**CALTECH-256**

* There are 30,607 images in this dataset spanning 257 object categories
* Object categories are extremely diverse, ranging from grasshopper to tuning fork
* The distribution of images per category are:
  + Min: 80
  + Med: 100
  + Mean: 119
  + Max: 827

**RESNET**

* The core idea of ResNet is introducing a so-called “identity shortcut connection” that skips one or more layers.
* Typical ResNet models are implemented with double- or triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between.
* The main structure of resnet is given in below images:

A screenshot of text

Description automatically generated

A screenshot of text

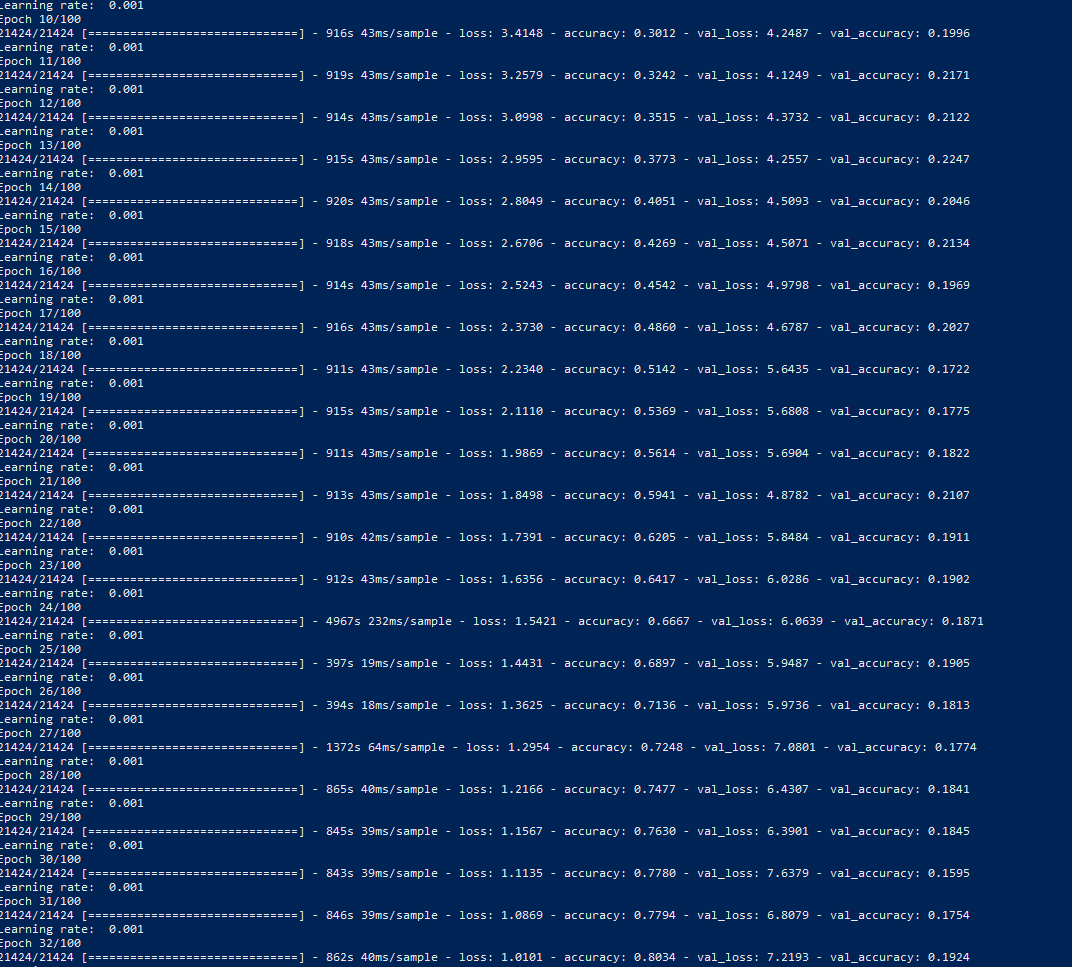
Description automatically generated

A screenshot of text

Description automatically generated

**Iterations for both datasets:**

* CALTECH-256

****

* SCENE-15

**A screenshot of a computer

Description automatically generated**

**RESULTS:** The results are obtained by implementing this architecture on the both the datasets (Scene-15 and Caltech-256) on pre trained model and by using transfer learning.

**Training accuracy**

|  |  |  |
| --- | --- | --- |
|  | Avg accuracy on pretrained model with average of 3 runs | Accuracy from scratch from 1 run |
| Scene-15 | 99.60% | 91.3% |
| Caltech-256 | 84.56% | 82.2% |

**Testing accuracy**

|  |  |  |
| --- | --- | --- |
|  | Avg accuracy on pretrained model with average of 3 runs | Accuracy from scratch from 1 run |
| Scene-15 | 40% | 37.19% |
| Caltech-256 | 44.56% | 21.23% |

**TIME:**

|  |  |
| --- | --- |
|  | Time |
| Scene-15 | 4.44 hours (for 200 epochs) |
| Caltech-256 | 15 hours (for 100 epochs) |

* The reason the accuracy is less that all images are loaded in one list and then train test split is done on the entire list so it may happen that images from some categories are not trained with model and directly tested and predicted wrongly. So to overcome this we can use some preprocessing techniques that can help to surpass this limitation.