IMAGE CLASSIFICATION WITH KERAS

Name of Group Members :- ML GROUP 6

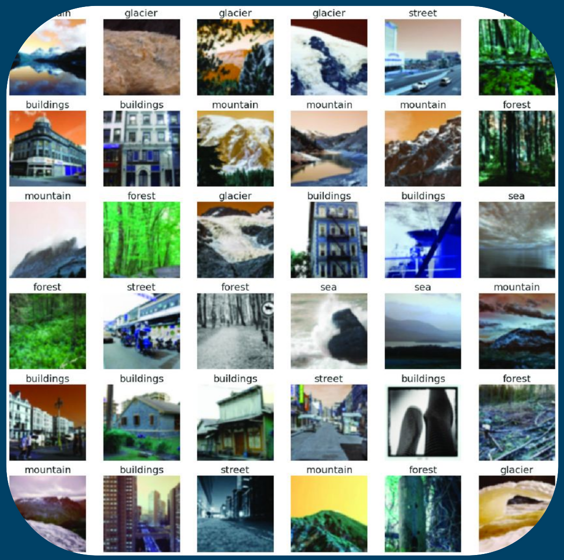
- Devanshi Pankaj Wadkar – 22065576
- Sandra Binu – 22029960
- Sohan Kumar Narayanaswamy – 22085239
- Nikhil Dinesh Soni – 22045846
- Melvin Sebastian - 22083837
- Hridhya Joy - 22074909

Google Collab Link:- Google collab link

Introduction To chosen Dataset & Brief of Pre-Trained Model

- The Intel Image Classification dataset contains images of natural scenes from around the world . The dataset comprises approximately 25,000 images, each with dimensions of 150x150 pixels, distributed across six categories:

1. Buildings
2. Forest
3. Glacier
4. Mountain
5. Sea
6. Street



Data Distribution:

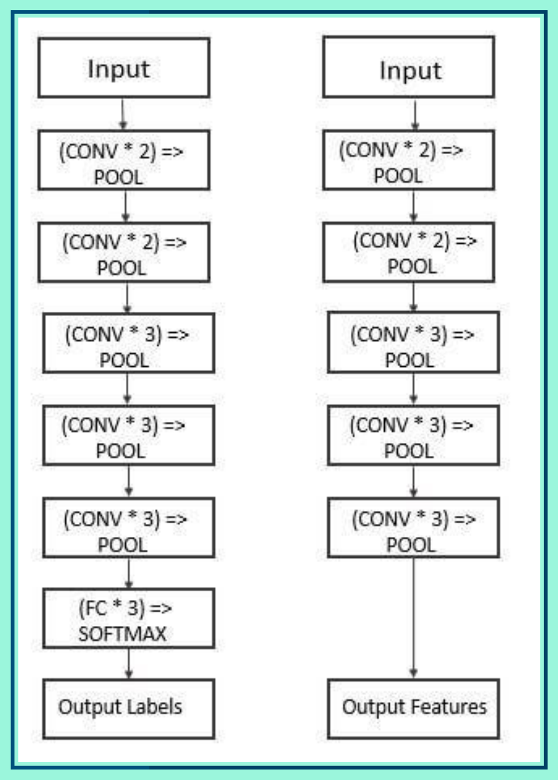
1. Training Images: 14,034 images
2. Test Images: 3,000 images
3. Prediction Images: Approximately 7,000 images

- In our study, we are using the VGG-16 which is pre-trained model on the dataset.
- **VGG-16's** advanced architecture enhances our ability to tackle the complexities of the dataset, ensuring a more accurate & effective classification process.
- Further, we compare the result of our transfer learning model with VGG 16 against the scratch model & analyze it on the basis of the training accuracy & validation accuracy, & predict the test data.

TRANSFER LEARNING AND ITS IMPORTANCE

- Transfer learning takes the help of an already developed ML model for finding out hidden trends on the data. It is more efficient as compared to a normal ML model because the amount of data required in this type of advanced ML learning is less than what is required to build an usual ML model.
- Generalization: Enable the model to learn general features from a source domain and apply them to different target domain.This technique helps us to discover solutions from earlier problems and implement the same to solve problems of similar nature
- Efficiency: Allows to leverage pre-trained models trained on large-scale datasets.Thus duration of training a transfer learning ML model is significantly lower as compared to training an usual ML model and lower computational resources.
- Improved Performance: The accuracy of a transfer learning ML model is significantly higher than the accuracy of an usual ML model by using pre-trained model's learned features, especially when the target dataset is small and lacks diversity.

SELECTION OF PRE-TRAINED MODEL (VGG-16)



For conducting the transfer learning analysis, we would use the VGG16 model to decipher the labels of the Natural Scenes around the world. This model is a type of advanced ML technique which would predict the class of images

The main benefit of using a VGG-16 model is the moderate count of hyperparameters which is used to build the model. There are convolution layers present in this model which uses a filter of shape 3*3 with a stride of 1. There are maxpool layers in the model which uses a filter of shape 2*2 with a stride of 2.

The VGG-16 model in a pre-trained form has a higher accuracy during classifying the image labels as compared to other advanced learning ML models. The reason for the same is the implementation of multiple small filters in place of a large filter

The VGG-16 model in a pre-trained form has lower amount of training time as compared to other advanced learning ML models. This model performs equally good with large amount of data

EDA and FINE TUNING

- The training and test images of the Natural Scenes around the world has been retrieved and the dataset have been explored using the following EDA methods:
  - ❖ Visualize images from each class.
  - ❖ Analysed the shape of image datasets.
  - ❖ Analysed the name and number of classes.
  - ❖ Number of images per class.

- The images are rescaled and a defined target size has been assigned to the training dataset, validation dataset and test dataset.

- During the fine tuning stage, we have refined the abstract representations of the VGG-16 model which is being reused. This would make the model more efficient to classify the images

- The fine tuning has been done by freezing all the layers and excluding the top layer(fully connected layer) convolutional layer of the VGG-16 model

- In order to limit the magnitude of changes done to the pretrained model, we would lower the learning rate of the model

Fine Tuning:- A machine learning technique where a pre-trained model is adjusted for a new task with a smaller dataset, optimizing its parameters for specific requirements following steps are involved in fine tuning process:-

1. Data Preprocessing:-By using the rescaling technique adjusting the scale of the pixel values within an image, a process essential for bringing pixel intensities into a standardized range.

2. Modification of the last classification Layers:- Adapting the output layer to match the number of classes in the target task.

3. Freezing certain layers:-It involves fixing weights in specific initial layers of pre-trained model preserves encoded features, preventing updates during new task training. During using the pre-trained model we freeze the VGG-16 layers & add my own dense layer we do this is because we want to use the knowledge acquired by pre-trained model & we fine tuned it for our task & achieve the specific outcome.

4. Batch Normalization & Dropout:- The Batch Normalization helps with training stability while, Dropout helps to prevent the overfitting.

Result Comparison: VGG-16 Model

```
Epoch 1/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 821s 7s/step - accuracy: 0.6707 - loss: 0.9875 - val_accuracy: 0.8687 - val_loss: 0.3664
Epoch 2/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 827s 8s/step - accuracy: 0.8642 - loss: 0.3781 - val_accuracy: 0.8633 - val_loss: 0.3538
Epoch 3/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 831s 8s/step - accuracy: 0.8799 - loss: 0.3269 - val_accuracy: 0.8877 - val_loss: 0.3200
Epoch 4/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 859s 8s/step - accuracy: 0.8936 - loss: 0.2934 - val_accuracy: 0.8883 - val_loss: 0.3136
Epoch 5/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 863s 8s/step - accuracy: 0.9022 - loss: 0.2710 - val_accuracy: 0.8873 - val_loss: 0.3079
Epoch 6/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 855s 8s/step - accuracy: 0.9102 - loss: 0.2455 - val_accuracy: 0.8803 - val_loss: 0.3123
Epoch 7/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 841s 8s/step - accuracy: 0.9136 - loss: 0.2316 - val_accuracy: 0.8913 - val_loss: 0.3097
Epoch 8/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 838s 8s/step - accuracy: 0.9214 - loss: 0.2067 - val_accuracy: 0.8903 - val_loss: 0.3076
Epoch 9/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 848s 8s/step - accuracy: 0.9329 - loss: 0.1887 - val_accuracy: 0.8860 - val_loss: 0.3099
Epoch 10/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 889s 8s/step - accuracy: 0.9325 - loss: 0.1870 - val_accuracy: 0.8883 - val_loss: 0.3294
```

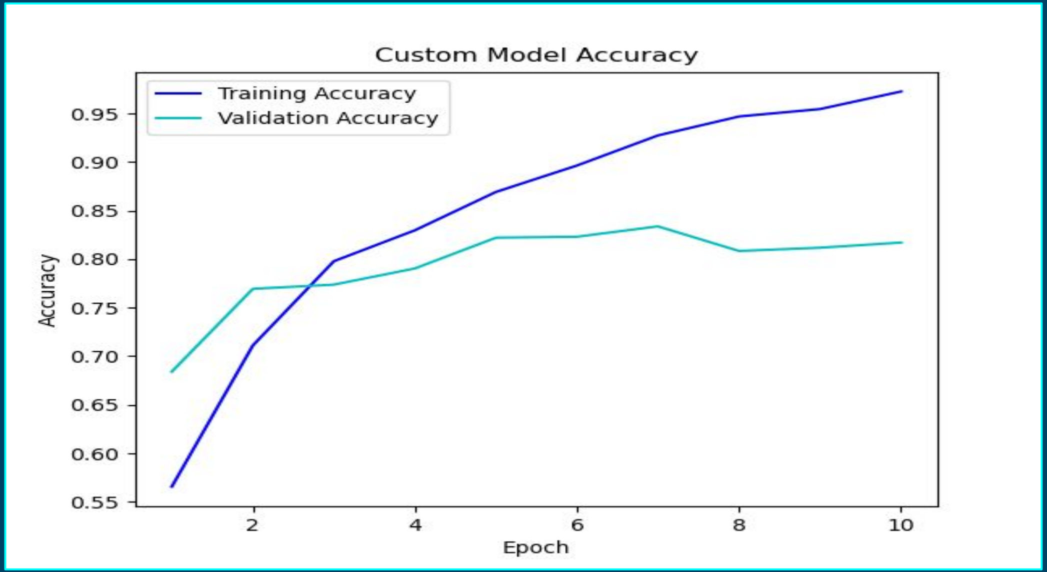
- The training is in 10 epochs and with batch_size=128. The training takes relatively constant ETA (13 min approximately) in each step.
- Training accuracy is increasing from 0.67 to 0.93 and Validation accuracy is increasing from 0.8687 to 0.888.
- Training loss is decreasing from 0.9875 to 0.187 and validation loss from 0.3664 to 0.329.

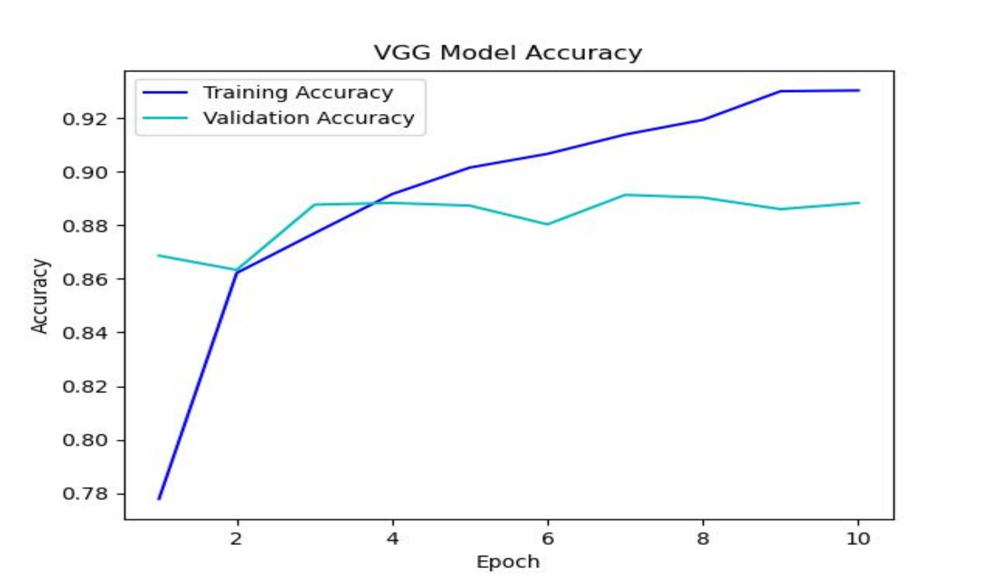
Result Comparison: CNN from Scratch Custom Model

```
Epoch 1/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 242s 2s/step - accuracy: 0.4590 - loss: 1.3357 - val_accuracy: 0.6840 - val_loss: 0.8296
Epoch 2/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 62s 569ms/step - accuracy: 0.6857 - loss: 0.8097 - val_accuracy: 0.7693 - val_loss: 0.6611
Epoch 3/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 79s 717ms/step - accuracy: 0.7999 - loss: 0.5557 - val_accuracy: 0.7737 - val_loss: 0.6328
Epoch 4/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 93s 842ms/step - accuracy: 0.8237 - loss: 0.4829 - val_accuracy: 0.7903 - val_loss: 0.5826
Epoch 5/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 89s 809ms/step - accuracy: 0.8627 - loss: 0.3812 - val_accuracy: 0.8220 - val_loss: 0.5172
Epoch 6/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 80s 721ms/step - accuracy: 0.8976 - loss: 0.2929 - val_accuracy: 0.8230 - val_loss: 0.5453
Epoch 7/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 96s 874ms/step - accuracy: 0.9264 - loss: 0.2218 - val_accuracy: 0.8337 - val_loss: 0.5750
Epoch 8/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 79s 719ms/step - accuracy: 0.9542 - loss: 0.1397 - val_accuracy: 0.8083 - val_loss: 0.6834
Epoch 9/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 86s 783ms/step - accuracy: 0.9565 - loss: 0.1291 - val_accuracy: 0.8117 - val_loss: 0.7019
Epoch 10/10
110/110 ━━━━━━━━━━━━━━━━━━━━ 90s 820ms/step - accuracy: 0.9779 - loss: 0.0773 - val_accuracy: 0.8170 - val_loss: 0.7641
```

- The training is in 10 epochs and with batch_size=128. The training takes relatively constant ETA (2 min approximately) in each step.
- Training accuracy is increasing from 0.459 to 0.978 and Validation accuracy is increasing from 0.68 to 0.817
- Training loss is decreasing from 1.3357 to 0.077 and validation loss from 0.83 to 0.764.

RESULT COMPARISON:Training accuracy vs Validation accuracy

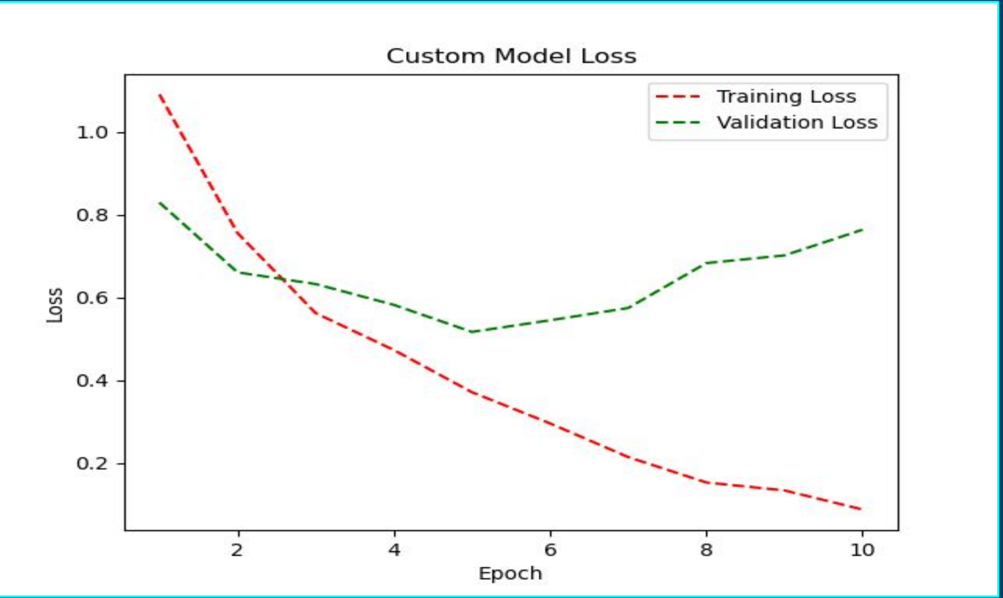


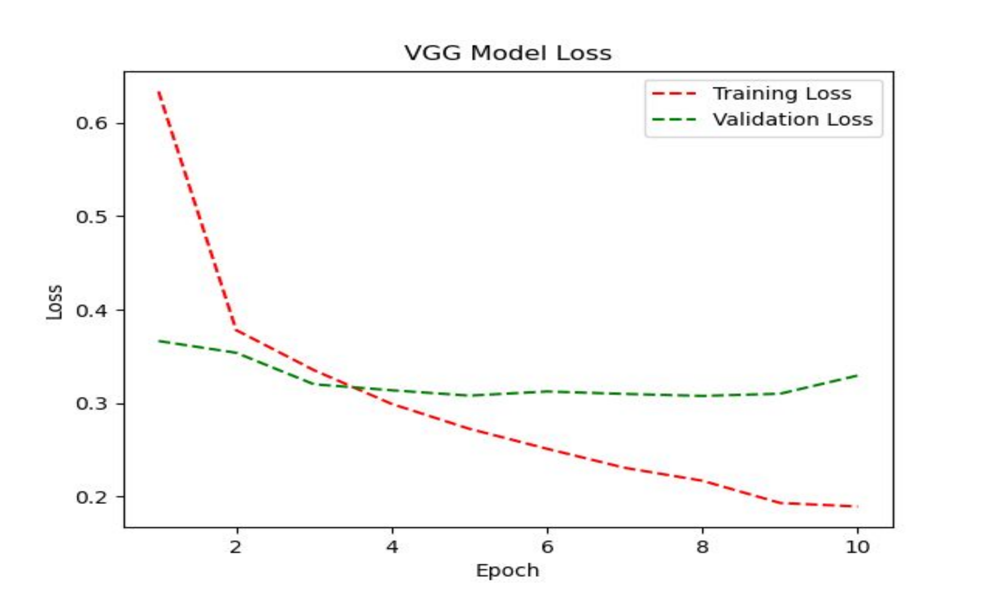


- both training and validation accuracy increasing over time suggest that the model is training well.
- the training accuracy is consistently higher than the validation. this may suggest overfit of the model.
- the training accuracy reaching out to the peak may suggest a case of overfitting.

- the training accuracy increases throughout the epochs suggest the model is learning from the training data.
- the validation accuracy also increases throughout but its fluctuates more than the training accuracy suggest it may overfit to the training data.

RESULT COMPARISON: Training Loss vs Validation Loss





- the training is decreasing over time, showing the model is learning the training data.
- the validation loss is decreasing but at a slower rate than training loss shows the model may start to overfit to the training data.
- the validation loss is much higher than the training loss which also suggests overfitting.

- training loss is generally decreasing over time, which suggest that the model is learning.
- validation loss is also decreasing which suggest that the model is generalizing well to the data.
- the validation loss is consistently lower than the training loss, which is a good sign and shows the model is not overfitting to training data.

Comparison of Both Models

- Both models exhibit good performance on the data, the pre-trained model (VGG-16) outperforms our custom or scratch model in terms of accuracy. The average accuracy metrics with average losses are detailed in the following table.

Model name	Training accuracy	Validation Accuracy	Training Loss	Validation Loss
VGG-16	0.91	0.88	0.23	0.32
From Scratch	0.847	0.79	0.414	0.649

- The training loss in the above table indicates how well the model performs on the training dataset, whereas the validation loss indicates how well the model performs on the dataset.
- The VGG - 16 model performs better than the other model with noticeably higher precision when comparing the two models' losses and accuracy. This is mostly due to the VGG-16 model's pre-training on comparable datasets using its weights. The weights of the layers that are frozen during the first training phase stay the same throughout the training procedure.
- As we did in our model training, an experiment may be carried out to further improve accuracy by unfreezing any or all of the layers, enabling the model to update its weights throughout ensuing training rounds. This contributes to improving accuracy, as the VGG-16 table illustrates.

LIMITATIONS AND POTENTIAL IMPROVEMENTS

- For the image classification of the Natural Scenes around the world, we have used the VGG-16 model. There are other models such as RESNET, ALEXNET, UNET which could also have been used to perform the image classification

- There are different activation functions which can be used to increase the efficiency of the VGG-16 model which has been developed from scratch

- The Adam optimizer has been used as the optimizer function for building the VGG-16 model from scratch. There are numerous optimizer functions such as RMSProp, SGD which can be used to increased the efficiency of the VGG-16 model

- The learning rate of the VGG-16 model can be altered to increase the efficiency of the VGG-16 model which has been developed from scratch

- Limitations:-

1. Depth challenges:- Deep Network may face training difficulties due to vanishing or exploding gradient

2. Overfitting in Dataset:- Large capacity may lead to overfitting in smaller datasets.

3. Architecture Complexity:- The complex architecture might be less suitable for real-time or edge computing

4. Interpretability:- Deeper architecture may reduces interpretability of learned features.

5. Data Augmentation is not used since the dataset ois sufficiently large enough.

- Future Enhancement:-

1. Efficient architecture:- Develop more efficient variant to reduce computational demands. We can use ResNet 101,ResNet50V2 etc. versions of the ResNet to improve classification of images in real world scenario.

2. Architectural Modifications:- Investigate modifications for improved convergence in very deep networks.

3. Interpretability:- Advance methods for interpreting & explaining model decisions.

CONCLUSION

- The loss of the VGG-16 ML model which indicates wrong classification of the image label has decreased significantly for both the models developed from scratch and the transfer learning ML model with an higher epoch
- The accuracy of the VGG-16 ML model which indicates correct classification of the image label has enhanced significantly for both the models developed from scratch and the transfer learning ML model with an higher epoch
- The transfer learning ML model has a higher accuracy which indicates correct classification of the image label as compared to the ML model which has been developed from scratch
- The transfer learning ML model has a lower loss which indicates wrong classification of the image label as compared to the ML model which has been developed from scratch

THANK YOU