A Comparative Analysis of Transfer Learning derived CNN model and Scratch Convolutional Neural Network (CNN) on CIFAR-10 Dataset

Gaurav Kumar (student Id- 21061492)	ColabLink of the code 1. RESNET50 2. CNN From Scratch

University of Hertfordshire UH

Introduction

This Report is about a comprehensive exploration of transfer learning and its application to image classification using Keras, with a specific focus on the **ResNet50 model.** In this research report, we delve into the critical evaluation of the effectiveness of transfer learning in comparison to training a model from scratch, specifically in the context of the **CIFAR-10 dataset** employing **Convolutional Neural Network (CNN).** Our study encompasses key aspects such as understanding the principles of transfer learning, implementing them with Keras, carefully selecting an appropriate dataset, and conducting a thorough analysis of the results.

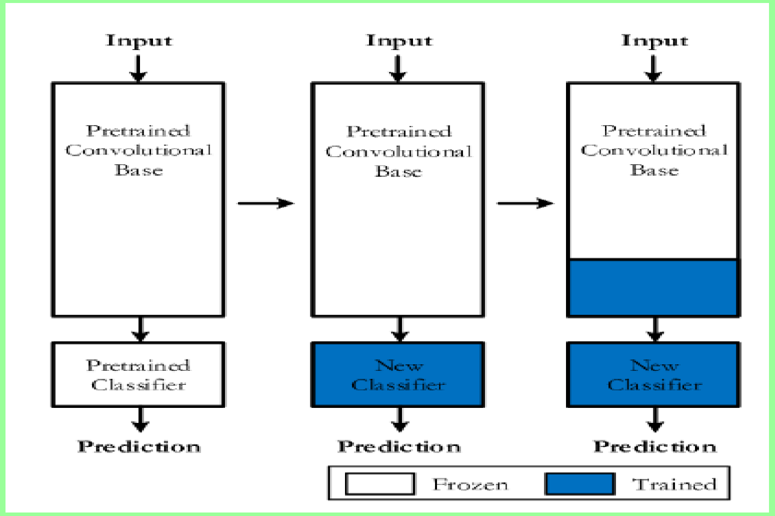
Throughout this report, we aim to shed light on the benefits and limitations of employing these two approaches in image recognition within **CNNs.** Notably, transfer learning involves building a **CNN** on top of a pre-defined and pre-trained CNN architecture like **VGG16, ResNet50, Inception and others**. The focal point of our investigation is the **ResNet50 model**, and we will unveil the insights gained from applying transfer learning techniques to the **CIFAR-10 dataset.**

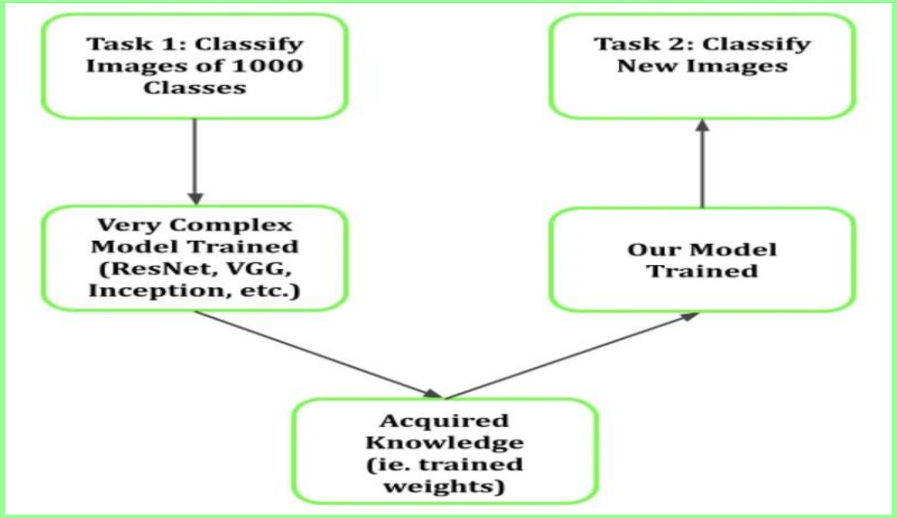
Moreover, we present a holistic view of the entire image processing pipeline within CNNs, providing a detailed exploration of the nuances between training models from scratch and leveraging pre-trained architectures. By the conclusion of this report, we aim to offer valuable insights into the strengths and potential areas of improvement for both transfer learning and conventional CNN training methods in the realm of image classification.

Transfer Learning Description

Transfer learning is a foundational concept in machine learning, especially potent in deep learning. It involves leveraging insights gained from solving one problem to address a different but related problem. In deep learning, this method proves highly effective, particularly when utilizing pre-trained models. The rationale is that models trained on extensive datasets, such as ImageNet for image classification, have acquired generic features transferable to new tasks. This hinges on the belief that fundamental features like edges and textures are universally valuable in visual recognition tasks. By initiating with a pre-trained model, significant computational resources and training time are saved, enabling meaningful results even with limited labeled data for the specific task. Numerous scientific studies support transfer learning's efficacy, demonstrating its ability to enhance model performance, expedite convergence, and capture intricate patterns in datasets with limited samples. This approach repurposes representations learned by deep neural networks across tasks, solidifying transfer learning as a valuable paradigm in contemporary machine learning research. **The diagram below shows the process how we can use the pre-trained weights for new model and classification.**

Diagrams showing process for Transfer learning




Task 1: Classify Images of 1000 Classes
Very Complex Model Trained (ResNet, VGG, Inception, etc.)
Acquired Knowledge (ie. trained weights)
Our Model Trained
Task 2: Classify New Images

Advantages of using Transfer Learning for CIFAR Dataset

Reduced Training Time : We can use pre-trained models for efficient learning, saving computational resources during training on the CIFAR dataset.
Improved Generalization: we can enhance CIFAR performance by transferring knowledge from a pre-trained model like RESNET50, promoting better generalization to a new data.
Overcoming Data Scarcity :If there is limitation of data pre-trained models can play a crucial role in image classification, particularly beneficial when task-specific data
Feature Extraction : Employ pre-trained CNNs to extract generic features, serving as a robust foundation for CIFAR-specific feature learning.
Fine-Tuning : Adapt pre-trained models to CIFAR nuances through fine-tuning, customizing them for task-specific features and improved performance.
Increased Robustness : Boost model resilience to variations in CIFAR data by leveraging the diversity experienced during pre-training on different datasets.
Broader Applicability : Transfer learning method can be applied on various other tasks apart from classification.

About RESNET50

In **2015, Microsoft Research introduced ResNet**, a groundbreaking advancement in deep neural networks that effectively tackled the vanishing gradient problem. ResNet's pivotal contribution lies in its introduction of residual learning, a technique employing skip connections to facilitate the direct flow of information across network layers. This innovation addresses the challenges posed by training very deep networks, overcoming the limitations of conventional architectures. Originally conceived for image classification, **ResNet's deep structures empower** it to capture intricate hierarchical features, proving highly adept in a multitude of computer vision tasks. Its inaugural triumph in the **2015 ImageNet Large Scale Visual Recognition Challenge** underscored ResNet's exceptional performance in image classification, establishing its superiority over shallower architectures..

About CIFAR10 Dataset

The CIFAR-10 dataset serves as a widely adopted resource for tasks related to image classification, standing as a prevalent benchmark for assessing the efficacy of deep learning models. Comprising a total of 60,000 32×32 color images distributed across 10 distinct classes, each class encompasses 6,000 images. The classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is partitioned into 50,000 images for training and 10,000 images for testing. A representative sample of the dataset, featuring images from all 10 classes, is provided below.

EDA performed on Data

Displayed the data and verified whether data class and the label are matching , and there are in total 10 class for both training and test set as shown in below diagram.

Fine-Tuning Process Overview for RESNET50

1. Feature Extraction Setup:

Leveraged a pre-trained ResNet50 model with ImageNet weights for feature extraction.
Adapted the architecture to accommodate an input size of (224 x 224).

2. Classifier Design:

Implemented a classification head consisting of global average pooling, dense layers (1024, 512), and a softmax layer with 10 output classes.

3. Up sampling for Input Transformation:

Resized input images from (32 x 32) to (224 x 224) using a (7x7) up-sampling layer.

4. Model Integration:

Connected the feature extraction and classifier layers to create the final model architecture.

5. Optimizer and Compilation:

Utilized Stochastic Gradient Descent (SGD) with a learning rate of 0.01.
Employed Sparse Categorical Crossentropy as the loss function for single-label classification.
Monitored accuracy as the evaluation metric.

6. Learning Rate Schedule:

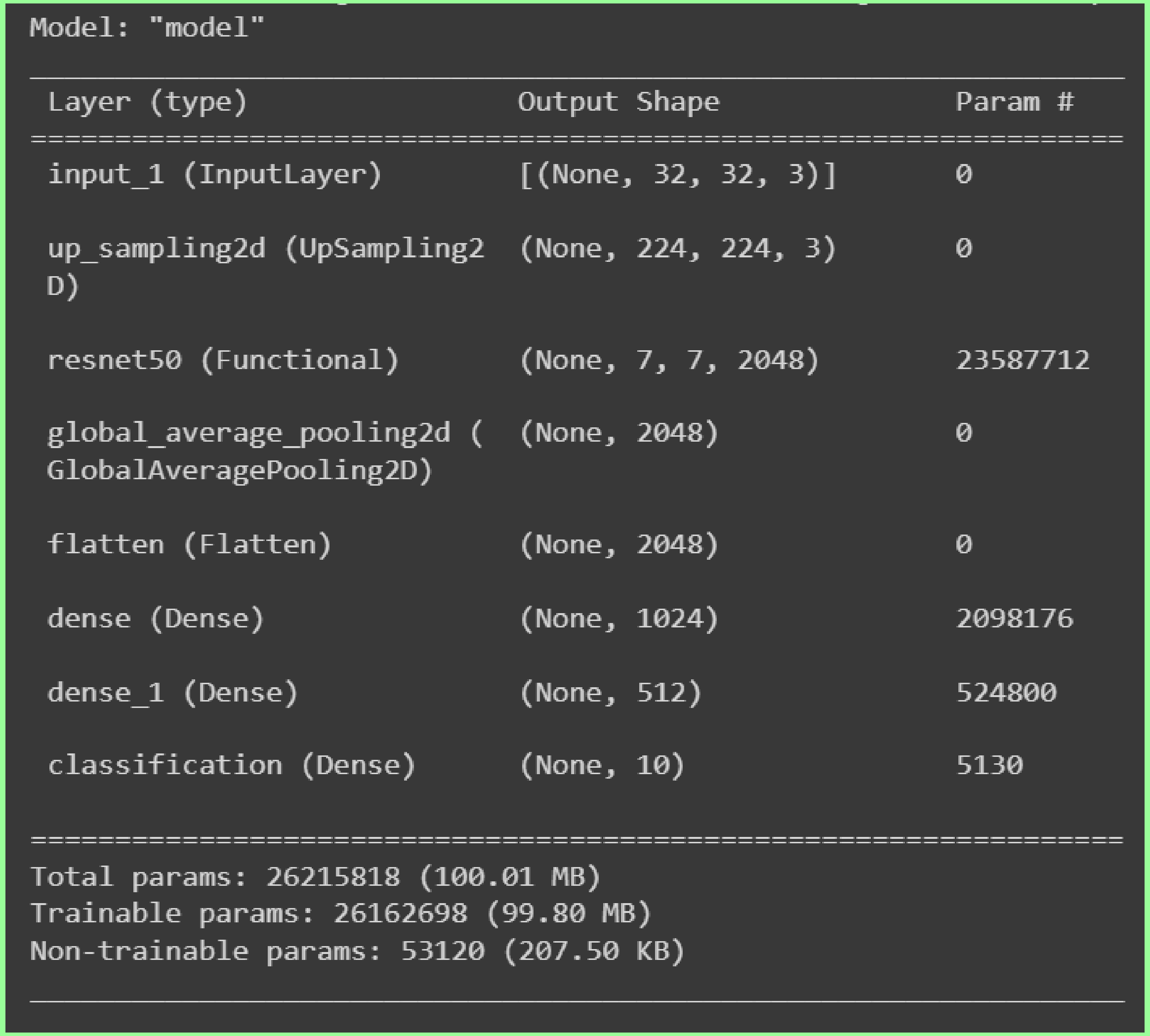
Designed a custom learning rate schedule, keeping a constant rate for the initial epochs and applying exponential decay thereafter.

7. Implementation Summary:

The fine-tuned model is set up for training on the target task, with a focus on efficient adaptation and optimal performance.

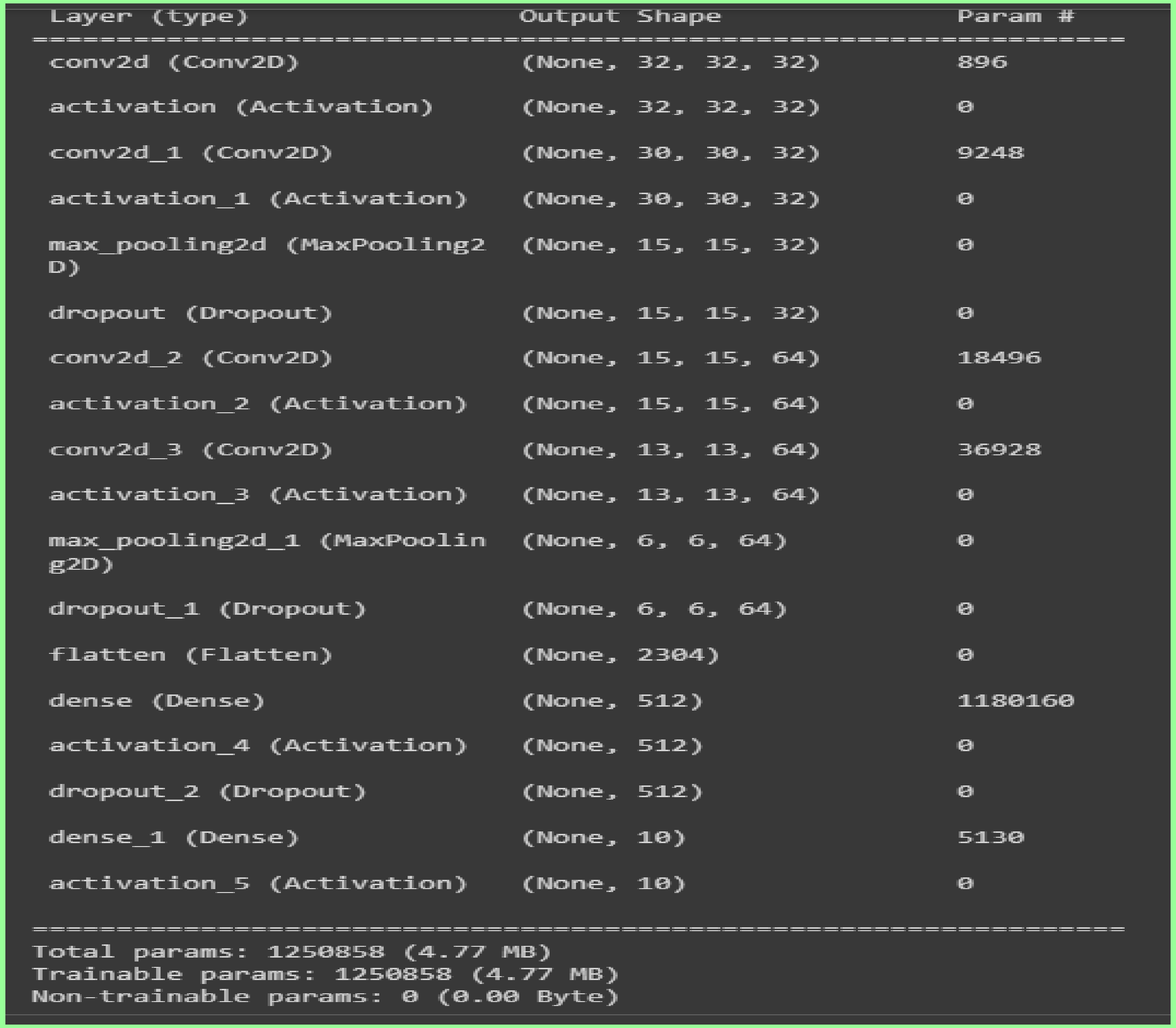
8. **Parameters for RESNET50:** We have included **include_top=False**, to indicate that we want the full convolutional base of ResNET50 without the final classification layer. To leverage ImageNet feature extraction on CIFAR10.

The comparison of model for RESNET50 and Scratch CNN model architecture and Trainable Parameters



```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 32, 32, 3)]       0

 up_sampling2d (UpSampling2  (None, 224, 224, 3)       0
 D)

 resnet50 (Functional)       (None, 7, 7, 2048)        23587712

 global_average_pooling2d (  (None, 2048)              0
 GlobalAveragePooling2D)

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 1024)              2098176

 dense_1 (Dense)             (None, 512)               524800

 classification (Dense)      (None, 10)                5130

=================================================================
Total params: 26215818 (100.01 MB)
Trainable params: 26162698 (99.80 MB)
Non-trainable params: 53120 (207.50 KB)
_________________________________________________________________
```

RESNET50



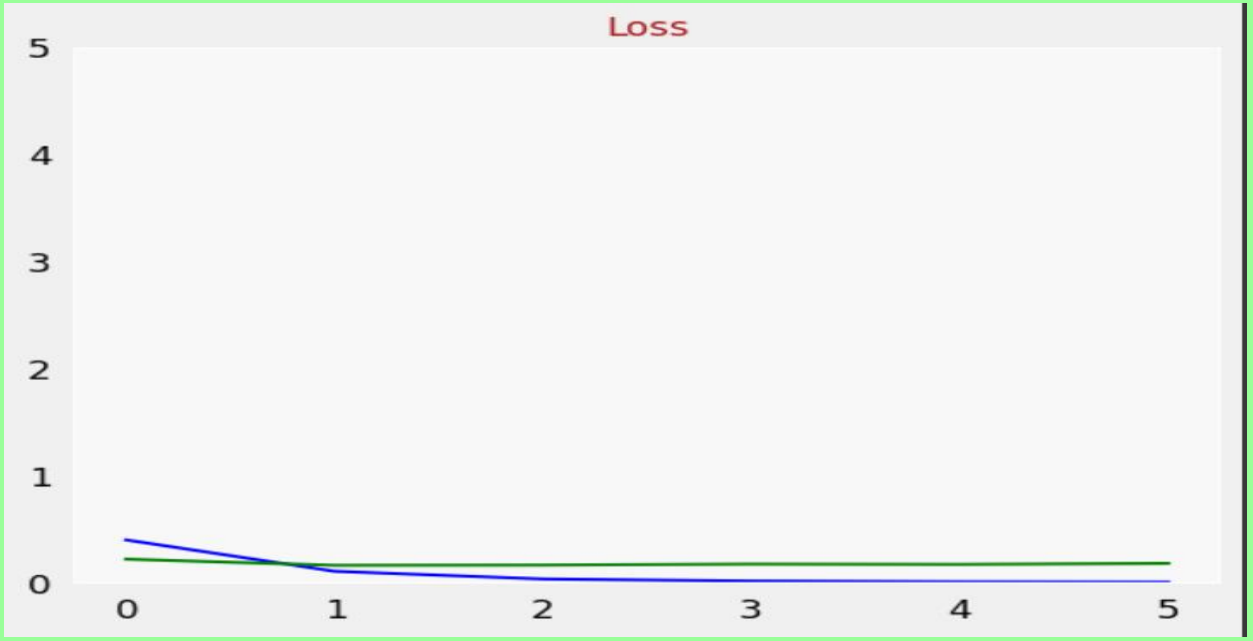
```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 32, 32, 32)        896

 activation (Activation)     (None, 32, 32, 32)        0

 conv2d_1 (Conv2D)           (None, 30, 30, 32)        9248

 activation_1 (Activation)   (None, 30, 30, 32)        0

 max_pooling2d (MaxPooling2  (None, 15, 15, 32)        0
 D)

 dropout (Dropout)           (None, 15, 15, 32)        0

 conv2d_2 (Conv2D)           (None, 15, 15, 64)        18496

 activation_2 (Activation)   (None, 15, 15, 64)        0

 conv2d_3 (Conv2D)           (None, 13, 13, 64)        36928

 activation_3 (Activation)   (None, 13, 13, 64)        0

 max_pooling2d_1 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 dropout_1 (Dropout)         (None, 6, 6, 64)          0

 flatten (Flatten)           (None, 2304)              0

 dense (Dense)               (None, 512)               1180160

 activation_4 (Activation)   (None, 512)               0

 dropout_2 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 10)                5130

 activation_5 (Activation)   (None, 10)                0

=================================================================
Total params: 1250858 (4.77 MB)
Trainable params: 1250858 (4.77 MB)
Non-trainable params: 0 (0.00 Byte)
```

CNN_ Scratch

Accuracy and loss curve analysis for RESNET50

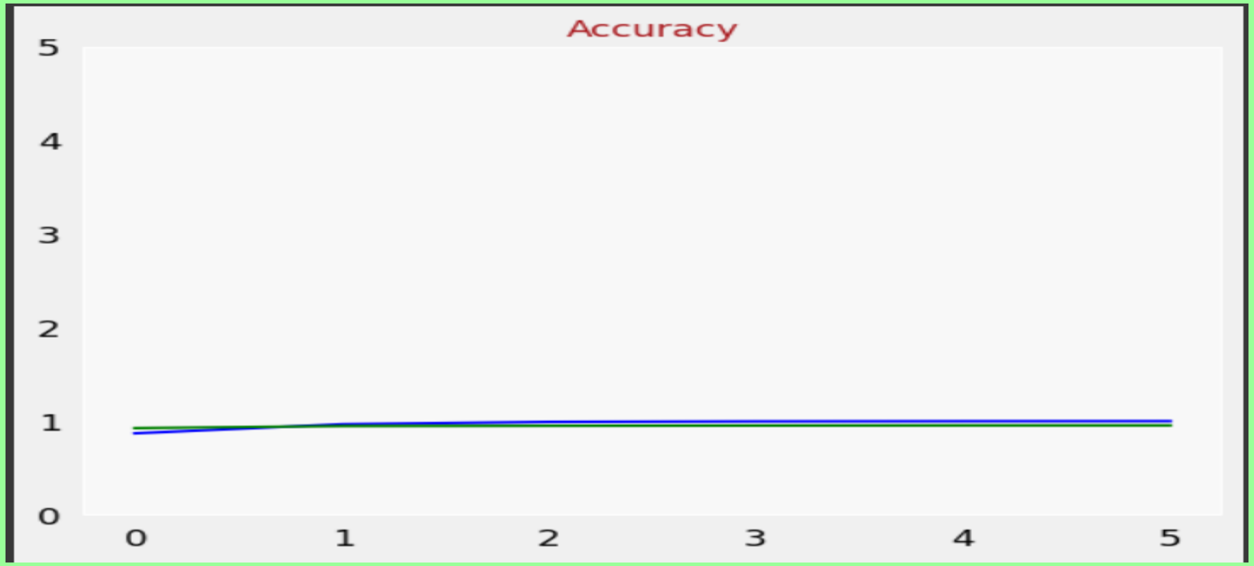
Loss curve analysis

1) From Loss curve we can see there are no sudden spikes or increasing validation, this indicates that there is "”no overfitting”.
2) Low loss value indicates accuracy of our predictions.
3) Loss curve show descending trend , means it is improving over epochs.
4) The loss curve also is indication a successful convergence as it is stable and smooth.



Accuracy curve analysis

1) From Accuracy curve we can see there are Parallel lines suggesting consistency and steady learning rate .
2) There is less or almost no noise or biases in the input data.
3) The output is positive as there is decreasing loss and steady accuracy.

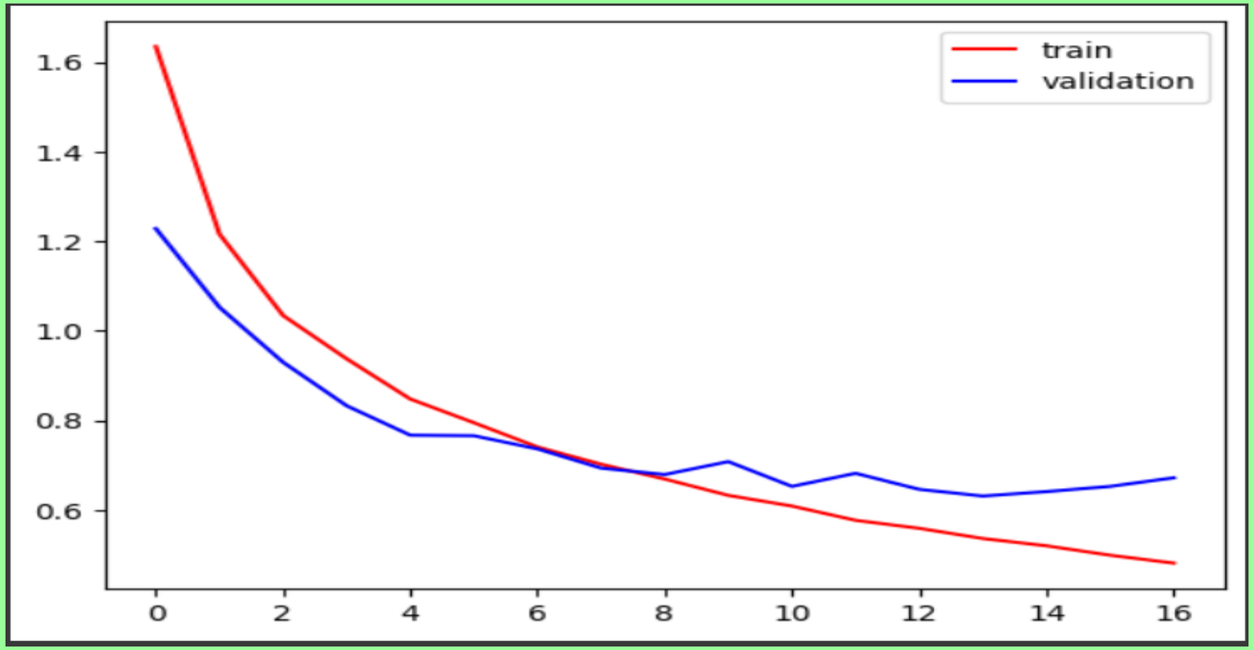


Accuracy and loss curve analysis for CNN Scratch

Loss curve analysis

The gradual decrease in both training and validation loss indicates that the model is converging and improving its ability to generalize to unseen data.

The training loss starts at 1.63 in the first epoch and gradually decreases over subsequent epochs, reaching around 0.48 by the 17th epoch. This decline indicates that the model is learning from the training data and improving its ability to make accurate predictions.

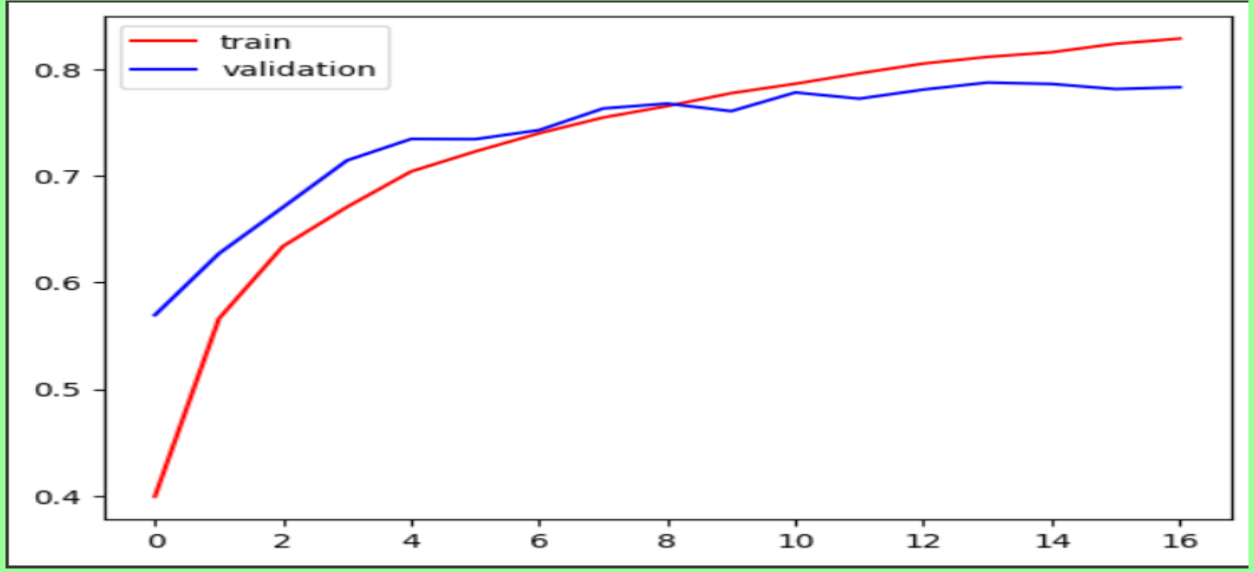


1) .

Accuracy curve analysis

1. Gradual increase in accuracy over epochs suggest good training process and gap between the 2 graphs indicate less overfitting.

2. Minor fluctuations in both training and validation accuracy are normal. Overall upward trends signify positive learning and continuous performance enhancements.



Comparison of results: Transfer learning vs. from scratch CNN Model

Parameter for best model

Model	EPOCS	Learning rate	Batch Size	Train Accuracy	Val Accuracy	Test Accuracy	Test Loss
RESNET50	6	0.01	64	92%	95%	95.33%	0.1792
Scratch CNN	50(early Stopping 17)	0.01	64	82%	78.3%	77%	0.6967

RESNET50: In summary, the training of the **ResNet50** model over **6 epochs** has resulted in an exceptionally high level of accuracy, particularly notable in the classification task. Commencing with an initial training accuracy of **86.91%** in the **1st** epoch, the model showcased rapid and robust convergence, achieving a staggering accuracy of **99.91%** by the final epoch. The validation accuracy has consistently reflected the model's prowess, peaking at **95.33%**. However, it is important to note a slight increase in validation loss during the final epoch, potentially indicating a mild case of overfitting. The learning rate remained constant at **0.01** throughout training. The comprehensive nature of the **ResNet50** architecture, with its deep and residual connections, evidently facilitated the efficient learning of intricate features from the dataset. As such, while the model's high accuracy is commendable, further investigation into potential overfitting and fine-tuning of hyper parameters may be beneficial to ensure optimal generalization to unseen data. Overall, the **ResNet50** model's training process underscores its remarkable capacity for accurate image classification, laying the foundation for future optimization and deployment in real-world scenarios.

Comparison of results: Transfer learning vs. from scratch CNN Model - Continued…

Scratch CNN : The training of the **Convolutional Neural Network (CNN)** from scratch for image classification has exhibited promising results over the course of **17 epochs(early stopping)**. The model demonstrates a clear trend of improvement in both training and validation accuracy, showcasing its ability to learn and generalize from the provided dataset. Starting with an initial accuracy of **39.9%** in the first epoch, the model's performance steadily increased, reaching a validation accuracy of **78.31%** in the final epoch. The training process was efficient, and the early stopping mechanism kicked in at the **17th epoch**, indicating a stable and well-generalizing model. The consistent rise in accuracy suggests that the model effectively learned intricate features from the images, contributing to its overall effectiveness in the classification task. Further analysis and fine-tuning could potentially lead to even better performance, and considerations for optimizing **hyper parameters** or exploring additional data augmentation techniques might be avenues for future improvements. Overall, the results indicate a successful initial training phase with the potential for further refinement and application in real-world scenarios.

Conclusion

In summary, the choice between employing ResNet50 through transfer learning or training a CNN from the ground up hinges on factors like dataset size, resource availability, and the specific task requirements. While transfer learning offers efficiency and generalization advantages, its suitability depends on the unique characteristics of the use case. The key points can be succinctly captured as follows:

Performance Superiority: Transfer learning with ResNet50 generally outperforms a scratch CNN in terms of accuracy and convergence speed. The pre-trained weights on extensive datasets like ImageNet equip ResNet50 with valuable prior knowledge for learning intricate features.

Efficiency and Generalization: ResNet50 showcases superior efficiency and generalization by leveraging knowledge from diverse datasets. This results in quicker convergence and enhanced overall performance, especially beneficial for smaller target datasets like CIFAR-10.

Feature Extraction Advantage: ResNet50 excels in feature extraction, surpassing the capabilities of a scratch CNN. Its deep architecture and skip connections enable effective capture of complex hierarchical features, contributing to improved overall performance.

Overfitting Mitigation: Transfer learning aids in overfitting mitigation, as the pre-trained ResNet50 model has acquired robust representations from extensive data. In contrast, a scratch CNN may struggle to generalize, particularly with limited training data.

Resource Efficiency:

Training ResNet50 from scratch demands substantial computational resources and time. Transfer learning, by leveraging pre-existing knowledge, offers a resource-efficient training alternative, making it suitable for scenarios with constraints on resources.

Fine-Tuning Opportunities: ResNet50 provides opportunities for fine-tuning, allowing adaptation to specific nuances in the target dataset. Fine-tuning a pre-trained model often results in superior performance compared to training from scratch.

Limitations

Data Quality and Quantity: The model's performance may face limitations due to the scarcity of high-quality labeled data. Additionally, biased predictions, particularly in rare classes, can arise from imbalanced datasets.

Model Interpretability: Understanding the decision-making processes of complex models, like deep neural networks, can be challenging due to their inherent lack of interpretability.

Computational Resources: Requirement of high computational resources, this involves cost and thus limiting scalability.

Overfitting: Overfitting can occur on training data . This can lead to leading to poor generalization on new, and unseen data.

Potential Areas of Improvement

Enhancing Model Robustness and Generalization through Data Augmentation and Diversity: The incorporation of sophisticated data augmentation techniques, coupled with the collection of diverse data, serves to fortify the model against variations in the input, thereby elevating robustness and promoting better generalization.

Improving Model Interpretability with Explainability Techniques: The integration of explainability techniques, such as LIME or SHAP, contributes to enhancing the model's interpretability. This not only aids in understanding model predictions but also fosters trust among end-users by providing insights into the decision-making process.

Optimizing Model Efficiency through Model Optimization: Fine-tuning hyper parameters, exploring diverse architectures, or utilizing model compression techniques are effective strategies to optimize model efficiency. These actions result in improved performance while concurrently reducing computational requirements.

Mitigating Overfitting and Enhancing Generalization via Regularization Techniques: The application of regularization methods, including dropout or L1/L2 regularization, acts as a countermeasure against overfitting. This process bolsters the model's ability to generalize well to unseen data.

Boosting Performance with Transfer Learning: Leveraging other pre-trained models and incorporating transfer learning methodologies proves advantageous, particularly when confronted with limited labeled data. This approach enhances model performance by capitalizing on knowledge gained from broader datasets.

Enhancing Model Accuracy and Stability through Ensemble Methods: The implementation of ensemble methods, which involve combining predictions from multiple models, serves to elevate overall model accuracy and stability.

References

1. https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

2. https://www.tensorflow.org/guide/keras/transfer_learning/

3. https://keras.io/api/applications/

4. https://www.tensorflow.org/api_docs/python/tf/keras/datasets/

5. https://www.tensorflow.org/api_docs/python/tf/image/resize
6. https://www.researchgate.net/figure/TOP-LEVEL-DIAGRAM-OF-TRANSFER-LEARNING-FROM-A-PRE-TRAINED-CNN-MODEL_fig4_333882146
7. https://Github.com/
8. https://www.kaggle.com/
9. https://towardsdatascience.com/deep-learning-and-transfer-learning-31c6394b10c5