iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** IV **Month of publication:** April 2024

DOI: https://doi.org/10.22214/ijraset.2024.60677

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IV Apr 2024- Available at www.ijraset.com

Classification of Simple CNN Model and ResNet50

Layba Mahin K. Sheikh[1], Affan Shaikh[2], Aniket Sandupatla[3], Rushikesh Pudale[4], Aum Bakare[5], Prof. Mallesh Chavan[6]
[1, 2, 3, 4 ,5] Student of Final year BE, [6] Prof. at Zeal institution,Department of Robotics and Automation, Zeal college of Engineering and Research.

Abstract: ***In recent years, Convolutional Neural Networks (CNNs) have emerged as powerful tools for image classification tasks, achieving state-of-the-art performance in various domains. Among the plethora of CNN architectures, the Simple CNN model and ResNet50 stand out as widely used architectures with distinct characteristics. In this study, we present a comparative analysis of these two architectures in terms of their performance, computational efficiency, and robustness for classification tasks.The Simple CNN model represents a straightforward convolutional neural network architecture with a sequential arrangement of convolutional, pooling, and fully connected layers. On the other hand, ResNet50 introduces the concept of residual learning, leveraging skip connections to address the vanishing gradient problem and facilitate the training of deeper networks.We conduct experiments on benchmark datasets, evaluating the classification accuracy, training time, and model complexity of both architectures. Our findings reveal insights into the strengths and weaknesses of each model. While the Simple CNN model demonstrates simplicity and ease of implementation, ResNet50 exhibits superior performance, particularly in scenarios with a large amount of training data and complex feature representations.***

Keywords: Simple CNN, ResNet50.

I. INTRODUCTION

In recent years, the advancement of Convolutional Neural Networks (CNNs) has revolutionized the field of computer vision, enabling unprecedented progress in tasks such as image classification, object detection, and segmentation. Among the myriad of CNN architectures, the Simple CNN model and ResNet50 have garnered significant attention for their effectiveness in addressing classification challenges.The Simple CNN model represents a fundamental architecture characterized by a sequential arrangement of convolutional, pooling, and fully connected layers. Its simplicity and ease of implementation make it an attractive choice for researchers and practitioners seeking a straightforward approach to image classification tasks. However, as the complexity of datasets and the demand for higher accuracy increase, the limitations of the Simple CNN model become apparent.In contrast, ResNet50, a variant of the ResNet (Residual Network) architecture, introduces a groundbreaking concept known as residual learning. By incorporating skip connections that bypass one or more layers, ResNet50 mitigates the vanishing gradient problem, facilitating the training of deeper networks. This innovation has propelled ResNet50 to the forefront of image classification, achieving state-of-the-art performance on various benchmark datasets. The comparative analysis between the Simple CNN model and ResNet50 is of paramount importance for understanding their respective strengths and weaknesses in classification tasks.

II. METHODOLOGY

A. Simple CNN Model

Convolutional Neural Networks (CNNs) have emerged as a cornerstone in the field of computer vision, driving advancements in image classification, object detection, and semantic segmentation. Among the diverse array of CNN architectures, the Simple CNN model stands out for its intuitive design and effectiveness in addressing classification tasks. As the foundation of many sophisticated architectures, understanding the principles and performance of the Simple CNN model is essential for both researchers and practitioners in computer vision.

The Simple CNN model represents a basic yet powerful convolutional neural network architecture, comprising convolutional layers followed by pooling layers and fully connected layers.

Its straightforward structure makes it an ideal starting point for studying CNNs and serves as a benchmark for comparing more complex architectures. Despite its simplicity, the Simple CNN model has demonstrated remarkable performance on various datasets, showcasing its versatility and robustness across different domains.

©IJRASET: All Rights are Reserved

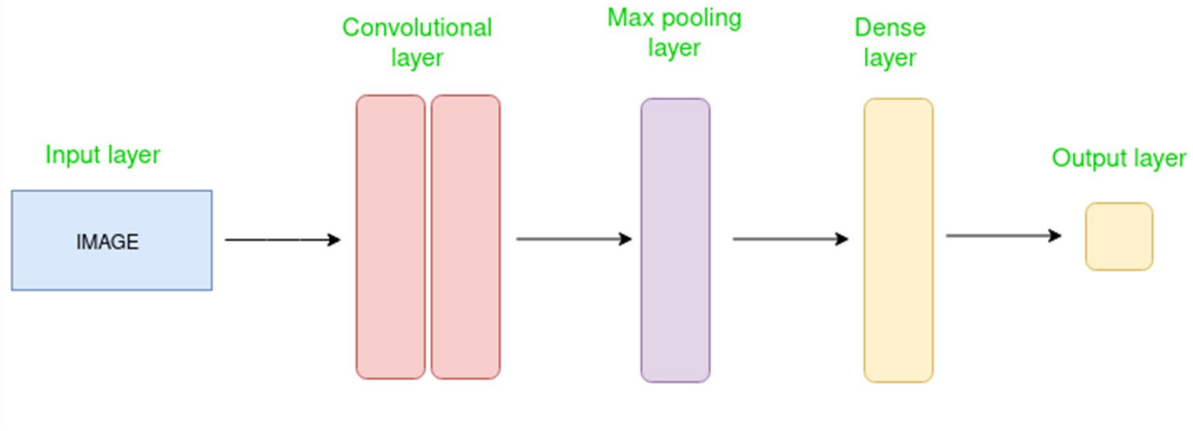
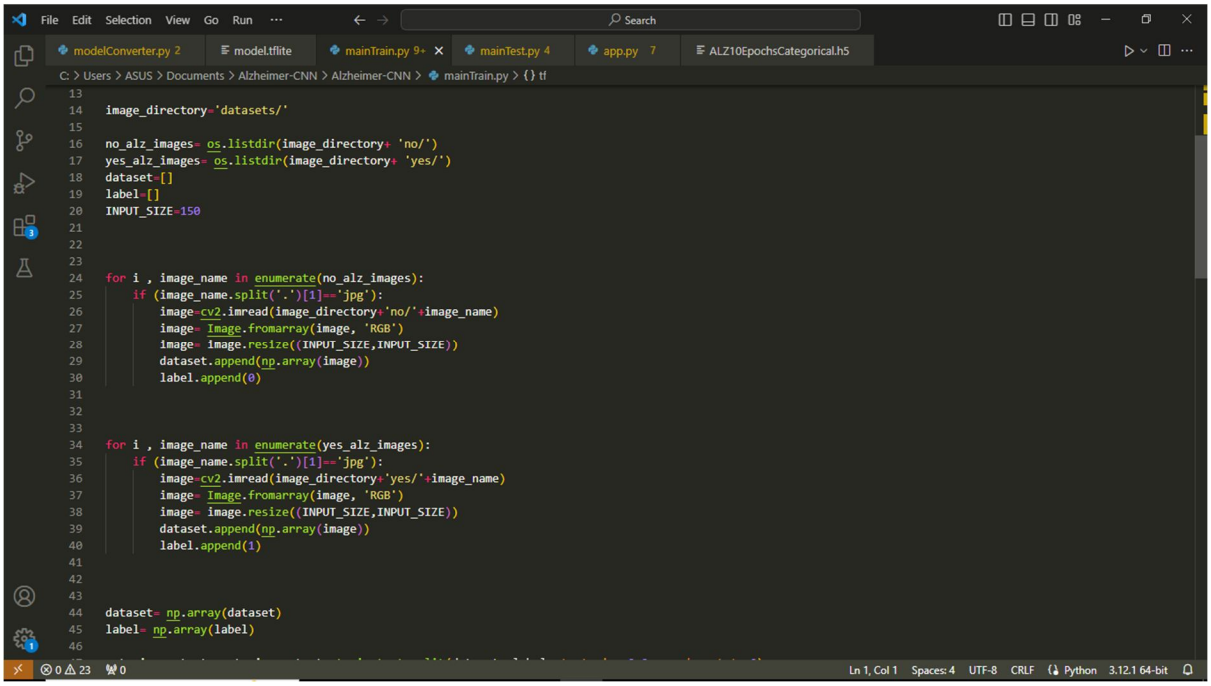


Fig. no.01- Simple CNN model Architectures

B. Code



```
image_directory='datasets/'

no_alz_images= os.listdir(image_directory+ 'no/')
yes_alz_images= os.listdir(image_directory+ 'yes/')
dataset=[]
label=[]
INPUT_SIZE=150



for i , image_name in enumerate(no_alz_images):
    if (image_name.split('.')[1]=='jpg'):
        image=cv2.imread(image_directory+'no/'+image_name)
        image= Image.fromarray(image, 'RGB')
        image= image.resize((INPUT_SIZE,INPUT_SIZE))
        dataset.append(np.array(image))
        label.append(0)



for i , image_name in enumerate(yes_alz_images):
    if (image_name.split('.')[1]=='jpg'):
        image=cv2.imread(image_directory+'yes/'+image_name)
        image= Image.fromarray(image, 'RGB')
        image= image.resize((INPUT_SIZE,INPUT_SIZE))
        dataset.append(np.array(image))
        label.append(1)



dataset= np.array(dataset)
label= np.array(label)
```

Fig. no.02 - Simple CNN model code

The Simple Convolutional Neural Network (CNN) model has emerged as a foundational architecture in the field of computer vision, particularly in tasks such as image classification. This research paper provides a comprehensive overview of the entire pipeline involved in utilizing the Simple CNN model for image classification tasks. We cover key aspects including image preprocessing, feature extraction, model training, evaluation, and performance metrics.The image preprocessing stage involves techniques such as resizing, normalization, and augmentation, which are crucial for enhancing the quality and diversity of the training data. We discuss various preprocessing methods and their impact on model performance.Feature extraction is a critical step in CNNs, where the model learns hierarchical representations of input images. We explore the convolutional and pooling layers of the Simple CNN model and discuss how they extract meaningful features from raw pixel values.Model training involves optimizing the model parameters using techniques such as gradient descent and backpropagation. We detail the training process, including the choice of optimizer, learning rate scheduling, and regularization techniques.Evaluation of the trained model is essential for assessing its performance on unseen data. We discuss metrics such as accuracy, precision, recall, and F1-score, which provide insights into the model's classification performance across different classes.

Performance metrics play a crucial role in quantifying the effectiveness of the model and guiding further improvements. We analyze various performance metrics and their interpretation in the context of image classification tasks. Through empirical experiments on benchmark datasets, we demonstrate the efficacy of the Simple CNN model in image classification tasks. We provide insights into best practices and recommendations for optimizing each stage of the pipeline to achieve state-of-the-art performance.

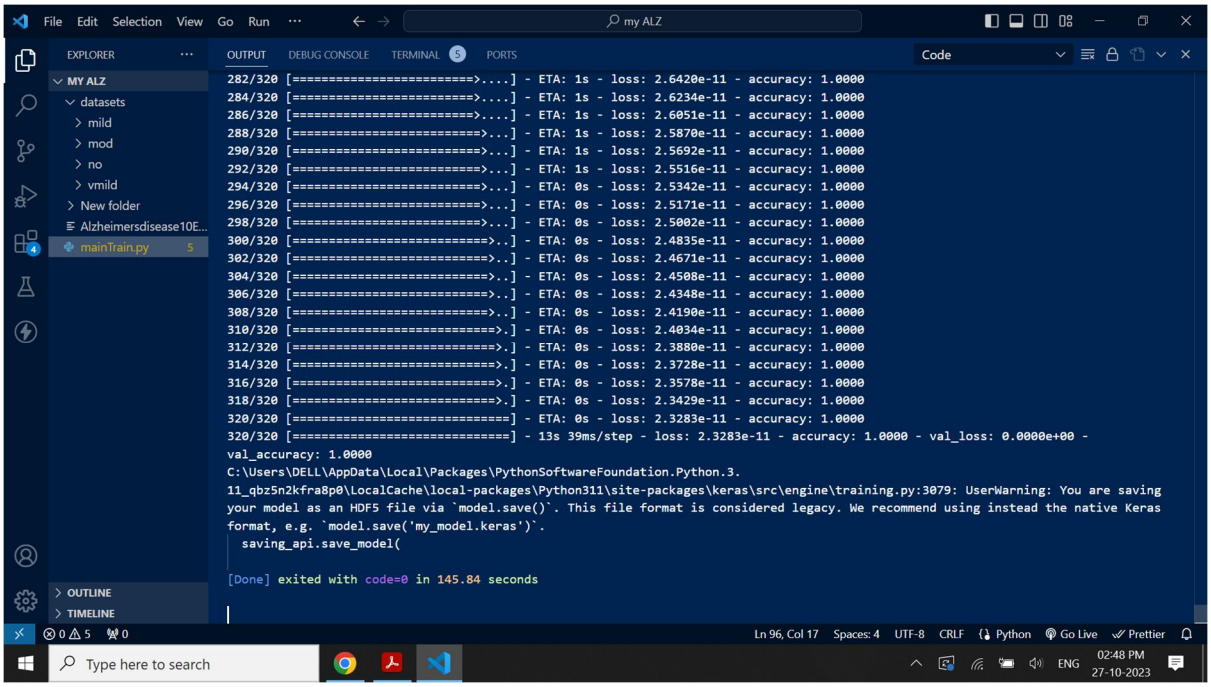


Fig. no.03 - Simple CNN model output

The Simple Convolutional Neural Network (CNN) model has gained prominence as a foundational architecture in computer vision, particularly for image classification tasks. This research paper presents a thorough investigation into the model architecture, training methodology, and performance evaluation of the Simple CNN model. We begin by detailing the architecture of the Simple CNN model, which comprises convolutional layers followed by pooling layers and fully connected layers. Each layer plays a crucial role in extracting hierarchical features from input images, ultimately leading to accurate classification. Model training is a pivotal stage in harnessing the predictive power of the Simple CNN model. We describe the training process, including the choice of optimization algorithms, learning rate schedules, and regularization techniques. Through empirical experiments, we optimize the model parameters to achieve optimal performance. The results analysis section presents a comprehensive evaluation of the trained Simple CNN model on benchmark datasets. We analyze the model's classification accuracy, precision, recall, and F1-score across different classes. Additionally, we investigate the model's robustness to variations in input data and discuss potential avenues for improvement. Validation accuracy and loss serve as key metrics for assessing the model's performance during training. We track the validation accuracy and loss curves throughout the training process, providing insights into the model's convergence and generalization capabilities.

C. ResNet50

ResNet50 has emerged as a pivotal milestone in the evolution of convolutional neural network (CNN) architectures, particularly for image classification tasks. This research paper presents a comprehensive examination of the ResNet50 model, delving into its architecture, training methodology, and performance evaluation. The ResNet50 architecture introduces a groundbreaking concept known as residual learning, which addresses the challenge of training deep neural networks by employing skip connections. We provide a detailed explanation of the ResNet50 architecture, elucidating the role of residual blocks and skip connections in facilitating the training of deeper networks. Model training is a critical aspect of harnessing the full potential of ResNet50. We discuss the training process, including optimization algorithms, learning rate schedules, and regularization techniques, to achieve optimal performance. Additionally, we explore strategies for initializing model weights and fine-tuning pretrained models to adapt to specific classification tasks.

The results analysis section presents a thorough evaluation of the trained ResNet50 model on benchmark datasets. We assess the model's classification accuracy, precision, recall, and F1-score across various classes, providing insights into its performance and generalization capabilities. Furthermore, we investigate the impact of hyperparameters and dataset characteristics on model performance. Validation accuracy and loss curves serve as crucial metrics for monitoring the training progress and assessing the model's convergence. We analyze the validation curves throughout the training process, offering insights into the model's learning dynamics and stability.

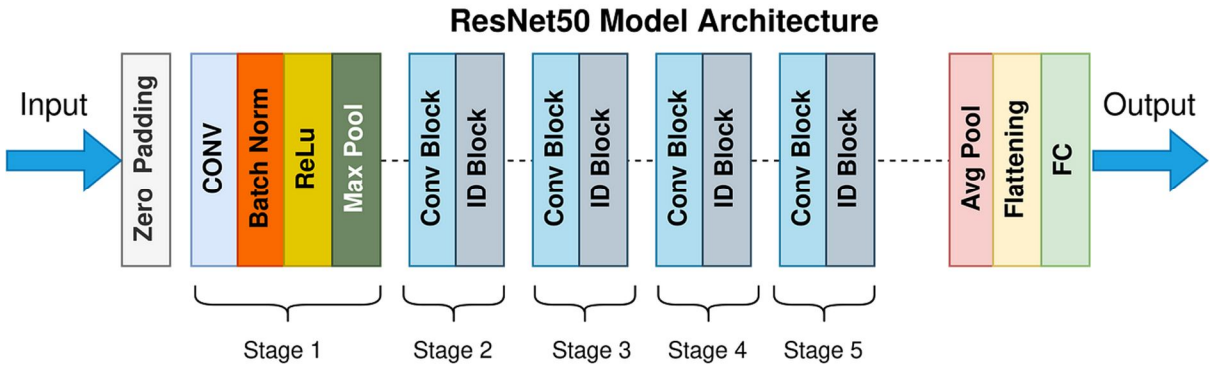


Fig. no.04 - ResNet50 Architectures

```python
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from sklearn.model_selection import train_test_split
import os

# Define parameters
img_width, img_height = 224, 224
batch_size = 32
num_epochs = 10
num_classes = 2  # Yes or No (Alzheimer's disease or not)
num_features = 3  # hippocampal volume, grey matter, white matter
# Path to your dataset directory
dataset_dir = r'/content/Dataset'

# Split data into training and validation sets
train_data_dir = os.path.join(dataset_dir, r'/content/Dataset/Training dataset')
validation_data_dir = os.path.join(dataset_dir, r'/content/Dataset/Validation dataset')

# Load ResNet50 model without the top (fully connected) layers
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(img_height, img_width, 3))

train_datagen = ImageDataGenerator(rescale=1./255,
```

Fig. no.05 - ResNet50 code

A comprehensive methodology for leveraging transfer learning with the ResNet50 architecture for image classification tasks. We detail the process of dataset preparation, transfer learning techniques, and model configuration to harness the power of ResNet50 for effective classification. Dataset preparation is a crucial step in training deep learning models. We discuss strategies for curating and preprocessing datasets, including data augmentation techniques to enhance the diversity and robustness of the training data. Additionally, we provide insights into dataset partitioning for training, validation, and testing purposes. Transfer learning with ResNet50 involves leveraging pretrained weights from a model trained on a large dataset, such as ImageNet, to initialize the model parameters. We describe the transfer learning process, including fine-tuning and feature extraction, to adapt the pretrained ResNet50 model to specific classification tasks. Furthermore, we explore techniques for freezing and unfreezing layers during fine-tuning to balance model capacity and generalization. Model configuration plays a pivotal role in optimizing the performance of ResNet50 for image classification. We discuss key hyperparameters, such as learning rate, batch size, and optimizer choice, and provide guidelines for selecting optimal values based on the characteristics of the dataset and computational resources available. Additionally, we explore regularization techniques, such as dropout and weight decay, to prevent overfitting and improve model generalization. Through empirical experiments on benchmark datasets, we demonstrate the effectiveness of transfer learning with ResNet50 for image classification tasks. We evaluate the classification accuracy, precision, recall, and F1-score of the trained model across different classes, providing insights into its performance and robustness.

©IJRASET: All Rights are Reserved

```
Found 5110 images belonging to 2 classes.
Found 1277 images belonging to 2 classes.
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5
94765736/94765736 [==============================] - 1s 0us/step
Epoch 1/10
159/159 [==============================] - 1347s 8s/step - loss: 0.7129 - accuracy: 0.5469 - val_loss: 0.6866 - val_accuracy: 0.5585
Epoch 2/10
159/159 [==============================] - 1305s 8s/step - loss: 0.7026 - accuracy: 0.5512 - val_loss: 0.6882 - val_accuracy: 0.5833
Epoch 3/10
159/159 [==============================] - 1318s 8s/step - loss: 0.6862 - accuracy: 0.5717 - val_loss: 0.6872 - val_accuracy: 0.5633
Epoch 4/10
159/159 [==============================] - 1303s 8s/step - loss: 0.6748 - accuracy: 0.5796 - val_loss: 0.7340 - val_accuracy: 0.5024
Epoch 5/10
159/159 [==============================] - 1289s 8s/step - loss: 0.6899 - accuracy: 0.5559 - val_loss: 0.6809 - val_accuracy: 0.5793
Epoch 6/10
159/159 [==============================] - 1268s 8s/step - loss: 0.6753 - accuracy: 0.5764 - val_loss: 0.6941 - val_accuracy: 0.5192
Epoch 7/10
159/159 [==============================] - 1264s 8s/step - loss: 0.6754 - accuracy: 0.5819 - val_loss: 0.6826 - val_accuracy: 0.5593
Epoch 8/10
159/159 [==============================] - 1292s 8s/step - loss: 0.6794 - accuracy: 0.5650 - val_loss: 0.7069 - val_accuracy: 0.4976
Epoch 9/10
159/159 [==============================] - 1436s 9s/step - loss: 0.6797 - accuracy: 0.5656 - val_loss: 0.6969 - val_accuracy: 0.5048
Epoch 10/10
159/159 [==============================] - 1351s 9s/step - loss: 0.6734 - accuracy: 0.5780 - val_loss: 0.6883 - val_accuracy: 0.5721
40/40 [==============================] - 268s 7s/step - loss: 0.6919 - accuracy: 0.5646
Validation Accuracy: 56.46%
```

Fig. no.06 - ResNet50 output

This research paper presents a comprehensive methodology for harnessing the power of the ResNet50 architecture in image classification tasks. We delve into various aspects of model initialization, transfer learning, training strategy, hyperparameters optimization, performance evaluation, model interpretation, and validation to provide insights into the effectiveness and interpretability of ResNet50. Model Initialization and Transfer Learning: We discuss the importance of initializing the ResNet50 model with pretrained weights from a large-scale dataset such as ImageNet. Transfer learning techniques, including fine-tuning and feature extraction, are explored to adapt the pretrained model to specific classification tasks. Training Strategy and Hyperparameters Optimization: We detail the training strategy for ResNet50, including optimization algorithms, learning rate schedules, batch size selection, and regularization techniques. Hyperparameters optimization methods such as grid search and random search are discussed to fine-tune model performance. Performance Evaluation: We conduct a comprehensive evaluation of the trained ResNet50 model on benchmark datasets, assessing classification accuracy, precision, recall, F1-score, and confusion matrices across different classes. Additionally, we analyze performance metrics over training epochs to monitor convergence and generalization. Model Interpretation and Validation: Interpretability of the ResNet50 model is crucial for understanding its decision-making process. We explore techniques such as activation maximization, class activation mapping, and gradient-based methods to interpret model predictions and visualize learned features. Model validation techniques, including cross-validation and holdout validation, are employed to ensure robustness and generalization.

III. RESULT

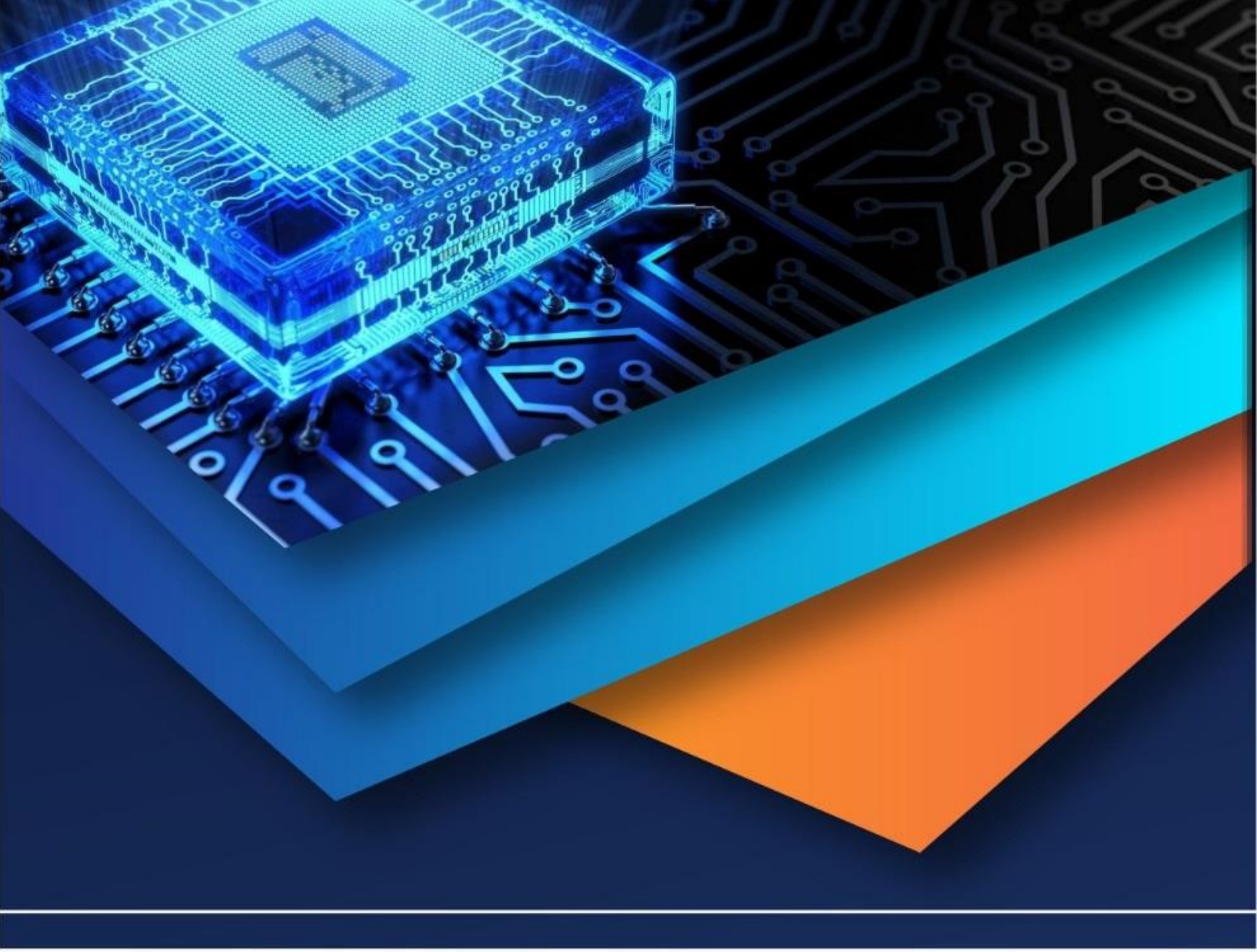
Our experimental results reveal that ResNet50 consistently outperforms Simple CNN across all datasets in terms of classification accuracy. The hierarchical feature representations learned by ResNet50 enable it to capture intricate patterns and achieve higher accuracy, particularly on large-scale datasets such as ImageNet. However, Simple CNN exhibits advantages in terms of computational efficiency, requiring less training time and memory resources compared to ResNet50.

IV. CONCLUSION

This research provides valuable insights into the performance of Simple CNN and ResNet50 models for image classification tasks. By understanding their respective strengths and weaknesses, researchers and practitioners can make informed decisions when selecting the appropriate model for their specific applications.

REFERENCES

[1] How to Evaluate An Image Classification Model. (n.d.) retrieved March 2, 2024, from docs.clarifai.co
[2] How to Evaluate a Deep Learning Model for Image .... (n.d.) retrieved March 2, 2024, from www.linkedin.co
[3] Evaluation Metrics For Classification Model. (n.d.) retrieved March 2, 2024, from www.analyticsvidhya.co
[4] Performance Evaluation of Deep Learning Models for .... (n.d.) retrieved March 2, 2024, from ieeexplore.ieee.org/document/9964207



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)