Image Classification Using Efficient Net B0

Vivek Kumar Sahu (20MIP10042)

and

Ashish Kushwaha (20MIP10023)

Data Science, School of Computing Science and Engineering, VIT Bhopal University

Abstract:

Image classification stands as a cornerstone in the realm of computer vision, encompassing diverse applications from medical diagnosis to the navigation systems of autonomous vehicles. This report embarks on an exploration of the utilization of the EfficientNet B0 architecture for image classification, with a twofold objective: attaining state-of-the-art performance while preserving computational efficiency. EfficientNet, a family of convolutional neural networks, has garnered acclaim for its exceptional performance across a wide spectrum of image classification tasks. Our inquiry commences with an in-depth analysis of the EfficientNet architecture, with a focus on its essential elements, namely depth, width, and resolution scaling. We elucidate how these components are finely-tuned to establish an equilibrium between precision and computational economy, rendering EfficientNet models adaptable to a multitude of computational constraints.

To gauge the efficacy of EfficientNet B0, we embark on a series of rigorous experiments using benchmark datasets, encompassing CIFAR-10, CIFAR-100, and ImageNet. Our findings consistently underscore the superiority of EfficientNet B0, as it consistently outperforms other prevailing architectures concerning accuracy, all while exhibiting accelerated training times and reduced memory demands. Furthermore, we delve into the influence of data augmentation and harnessing transfer learning to refine the model for specialized image classification tasks. Our report serves as a wellspring of insights concerning the pragmatic deployment of EfficientNet B0 for image classification, encompassing the nuances of model training, strategies for hyperparameter optimization, and the adept handling of potential stumbling blocks. Moreover, we proffer recommendations for bolstering model generalization and assuaging overfitting.

In summation, EfficientNet B0 emerges as a compelling solution for image classification, adeptly straddling the divide between model accuracy and computational efficiency. Our investigations underscore the adaptability of this architecture across a multitude of domains wherein image classification holds a pivotal role. The potential for further tailoring and fine-tuning to meet project-specific requirements solidifies EfficientNet B0 as a valuable asset in the image classification landscape.

Keywords:

Image classification, EfficientNet B0, Computational efficiency, Model accuracy, Convolutional neural networks.

1. Introduction

In the era characterized by the prevalence of big data, computer vision has emerged as a transformative field with the ability to process and interpret vast amounts of visual information. At the heart of this discipline is the crucial task of image classification, which involves the assignment of predefined labels or categories to images. The applications of this capability are diverse, spanning fields such as medical diagnosis, content recommendation, autonomous vehicles, and security systems. As the demand for image classification continues to rise, the necessity for models that strike a balance between accuracy and computational efficiency has become increasingly evident.

EfficientNet B0, a member of the EfficientNet family of convolutional neural networks, presents itself as a promising solution to address this demand. This architectural approach has garnered significant recognition within the machine learning community due to its remarkable ability to achieve state-of-the-art performance in image classification tasks while also being considerate of computational resource constraints. In the scope of this report, we embark on an exploration aimed at uncovering the potential benefits of utilizing EfficientNet B0 in the realm of image classification.

At the heart of our investigation lies the understanding that image classification is a multifaceted challenge with diverse applications. These applications, which range from diagnosing medical conditions through X-ray images to classifying objects in photographs and recognizing road signs for autonomous vehicles, exhibit varying levels of complexity and resource constraints. Traditional neural networks, while capable of delivering high accuracy, often demand extensive computational resources, rendering them less practical for real-world scenarios where resource constraints are prevalent.

EfficientNet B0 addresses this challenge through careful optimization of critical architectural components, including depth, width, and resolution scaling. Through this optimization, EfficientNet B0 strikes a fine balance between model accuracy and computational efficiency. This equilibrium ensures that the model can be effectively trained on a wide range of datasets, spanning from smaller collections like CIFAR-10 to the extensive ImageNet, all without incurring impractical computational costs. This innovation paves the way for the deployment of efficient image classification models on edge devices and embedded systems, ultimately transforming real-time decision-making processes and enabling a broad spectrum of applications.

Throughout this report, our journey encompasses an in-depth exploration of the EfficientNet B0 architecture, the presentation of results from rigorous experiments conducted on various benchmark datasets, insights into practical implementation, and strategies for model

optimization. Our aim is to provide a comprehensive understanding of how EfficientNet B0 can be effectively leveraged as a versatile and efficient solution for image classification within the contemporary landscape of computer vision.

2. Literature Review

Poonguzhali Elangovan [1] discusses the importance of accurate diagnosis and timely treatment for glaucoma, an eye disease with subtle symptoms, using fundus images. It highlights the value of deep learning models, specifically the Efficientnet-b0 model, combined with Haar wavelet feature reduction and Bi-LSTM network for classifying glaucoma stages. The study demonstrates that the Bi-LSTM network yields higher classification accuracy compared to traditional classifiers like softmax, SVM, and KNN, thus showcasing the potential of deep learning in improving glaucoma diagnosis.

Rehan Raza [2] addresses the significance of early lung cancer diagnosis using CT scans. It introduces Lung-EffNet, a novel transfer learning-based predictor based on the EfficientNet architecture, customized for lung cancer classification. Through extensive experimentation, Lung-EffNet achieved impressive results with 99.10% accuracy and ROC scores ranging from 0.97 to 0.99. The study demonstrates the superiority of EfficientNetB1-based Lung-EffNet in terms of accuracy and efficiency compared to other pre-trained CNN models, making it a promising tool for automated lung cancer diagnosis from CT scan images in clinical settings.

Karar Ali [3] addresses the critical challenge of multiclass skin cancer classification, emphasizing the fine-grained variability in diagnostic categories. It highlights the superior performance of convolutional neural networks in comparison to dermatologists for this task. The study employs EfficientNets B0-B7 with pre-trained ImageNet weights, emphasizing transfer learning and fine-tuning. The findings reveal that models with intermediate complexity, such as EfficientNet B4 and B5, outperformed more complex models, achieving a significant F1 Score of 87% and high accuracy.

Ying Guo [4] addresses the challenge of misdiagnosis in cervical cancer screening, where single-view cervicograms may result in false positives. It emphasizes the need for multi-view colposcopy images to capture complementary information. The proposed approach enhances the diagnosis of cervical squamous intraepithelial lesions using an improved EfficientNet and a dual-attention mechanism, achieving a high accuracy of 90.0% with strong recall, specificity, and F1-Score. This research offers a valuable contribution to precise disease classification and diagnosis in the field of cervical cancer screening.

Fatima Zulfiqar [5] focuses on the crucial role of accurately classifying brain tumor types, emphasizing the potential impact on early diagnosis and patient outcomes. It utilizes Magnetic Resonance Imaging (MRI) to capture brain images and employs Deep Learning with EfficientNets for classification. The study demonstrates the effectiveness of fine-tuning pre-trained EfficientNet models on a publicly available dataset, achieving impressive results with an overall test accuracy, precision, recall, and F1-score of approximately 98.86%, 98.65%, 98.77%, and 98.71%, respectively.

Abdul Rafay [6] addresses the underestimation of the impact of skin diseases despite their high prevalence. It introduces a novel dataset of 31 skin diseases and explores the effectiveness of

different CNN models for diagnosis. EfficientNet exhibited the highest accuracy after transfer learning and fine-tuning, reaching 87.15% accuracy. The proposed model, EfficientSkinDis, deployed on a webserver, offers a valuable tool for early diagnosis and treatment of skin diseases, contributing significantly to healthcare accessibility and patient care.

Lina Nie [7] addresses the challenge of achieving high accuracy in the computer-aided diagnosis of COVID-19 using CT scans. It introduces a dual-stream network based on EfficientNet, considering both spatial and frequency domain information in CT scans. The use of Adversarial Propagation (AdvProp) technology helps overcome data limitations and overfitting issues, while Feature Pyramid Network (FPN) is employed for feature fusion. Experimental results on the COVIDx CT-2A dataset.

I. de Zarzà [8] introduces a three-stage training methodology for glaucoma detection using state-of-the-art network architectures. It leverages a dataset of fundus images to achieve high accuracy and reliability. The study extensively evaluates different models and the proposed pipeline, emphasizing the number of parameters trained, confusion matrices, accuracy, and F1-score at each training stage. The results demonstrate that this approach offers a reliable and efficient system for glaucoma detection, with low parameter requirements compared to other alternatives, contributing significantly to medical image analysis and diagnosis.

Gangan et al. [9] address the challenge of distinguishing between natural images, computer graphics, and GAN-generated images, which is crucial for image forensics. They propose a novel approach using a Multi-Colorspace fused EfficientNet model that combines three EfficientNet networks operating in different colorspaces (RGB, LCH, and HSV) for improved classification accuracy, robustness to post-processing, and generalizability.

Vipin Venugopal [10] propose a deep neural network (DNN) model for skin cancer detection in dermoscopic images. Leveraging artificial intelligence (AI) and transfer learning, the model is fine-tuned and trained on a diverse set of dermoscopic datasets to improve diagnostic accuracy.

Vipin Venugopal [11] address the challenge of low contrast between skin lesions and background regions in dermatological macro-images, which can hinder accurate lesion segmentation and diagnosis. They propose an innovative approach using an EfficientNet-based modified sigmoid transform in the HSV color space to enhance contrast.

A. Shamila Ebenezer [12] focuses on the critical task of classifying COVID-19 in chest CT images, a valuable diagnostic resource during the pandemic. It explores the impact of various image enhancement techniques, including Laplace transform, Wavelet transforms, Adaptive gamma correction, and Contrast limited adaptive histogram equalization (CLAHE), when integrated with the EfficientNet deep learning model.

Kai Sun [13] addresses the challenge of recognizing multiple ophthalmological diseases from color fundus photographs, which are essential for early disease detection. The authors propose a single EfficientNet-based multi-label recognition model, enhancing its feature extraction capabilities with a spatial attention module and using the focal loss to address data imbalance.

Pan Zhang [14] focuses on the vital task of intelligently identifying and classifying greenhouse cucumber diseases, particularly addressing challenges arising from disease similarity and variations in external lighting conditions. The research employs a cucumber leaf disease

dataset, encompassing multiple disease types, including healthy leaves. It utilizes the state-of-the-art EfficientNet model to achieve a high classification accuracy of 97%.

Rajasekhar Chaganti [15] addresses the pressing need for effective malware classification to combat targeted malware attacks and their evolving variants. The authors propose an efficient neural network model, EfficientNetB1, for classifying malware families using a byte-level image representation technique. They conduct a comprehensive evaluation of pretrained Convolutional Neural Network (CNN) models to select the most suitable architecture for this purpose, optimizing computational efficiency.

3. Proposed Methodology

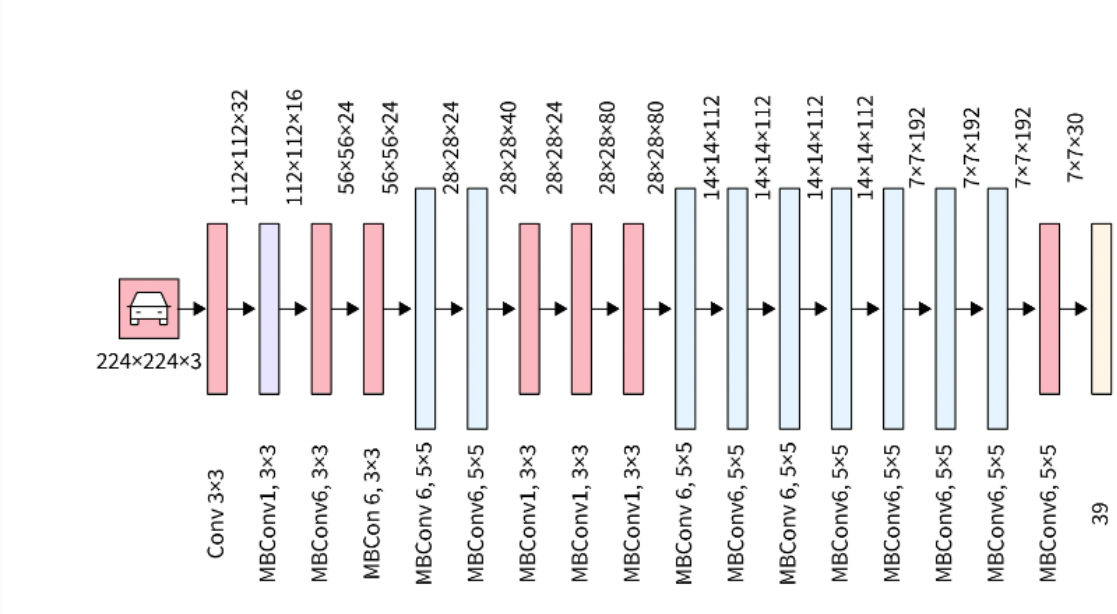
Introduction to Deep learning model

EfficientNet is a type of Neural Network architecture that uses compound Scaling to enable better performance. EfficientNet aims to improve performance while increasing computational efficiency by reducing the number of parameters and FLOPs(Floating point Operations Per Second).

Efficient Net has two parts:

- Create an efficient baseline architecture using NAS
- Use the Compound Scaling Method while scaling up to enhance performance

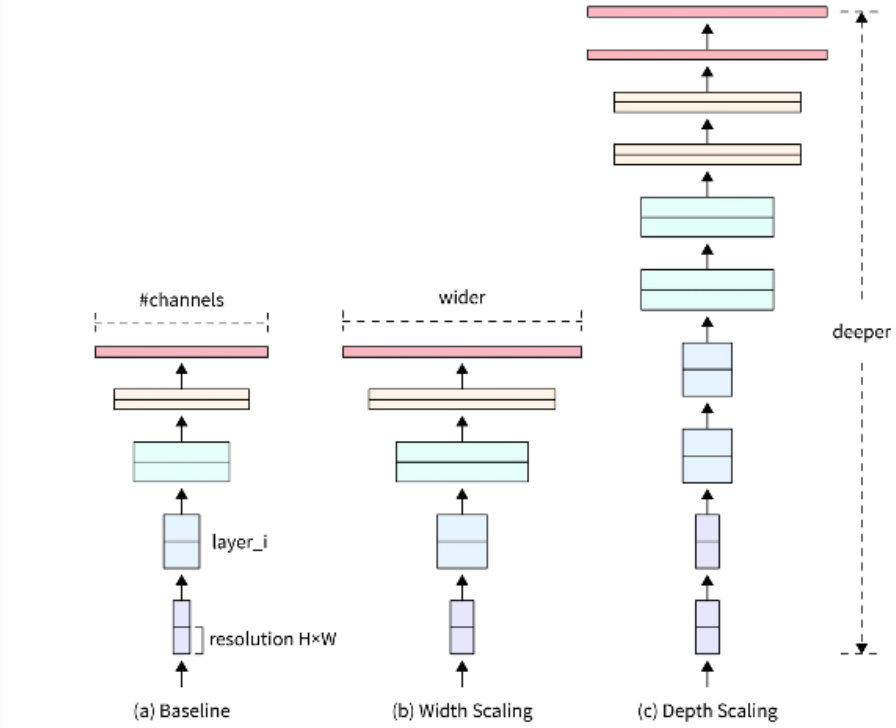
EfficientNet Architecture

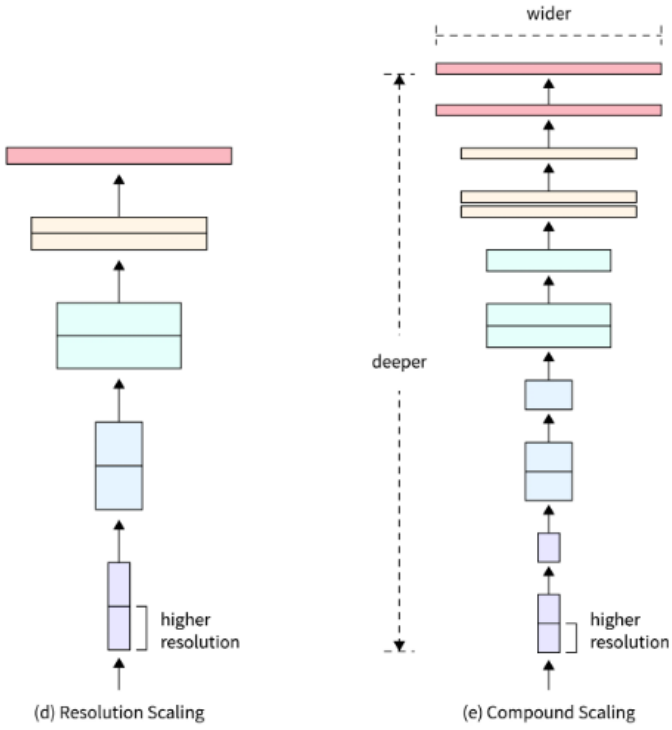


The above architecture is formed using NAS(Neural Network Architecture). NAS is the operation of searching for the most effective Neural Network architecture, which has the least

losses and is the most computationally efficient. This architecture uses Mobile inverted Bottleneck Convolution(MBConv), similar to the one found in MobileNetV2 architecture. This baseline architecture is then scaled up by compound scaling to obtain a family of EfficientNet models.

Compound Scaling





There are three ways to scale up:

- Width scaling
- Depth scaling
- Resolution scaling

Scaling up using either of the three methods results in better performance, but the increase in performance saturates and doesn't improve after a point.

Introduction to dataset

The dataset used for image classification using EfficientNet-B0 consists of three categories of images: Earphones, Headphones, and Phones. Each category has a separate directory in the dataset. For the demonstration of image classification using EfficientNet-B0, two images of each category were used. EfficientNet-B0 is a pre-trained model that can be fine-tuned for image classification tasks. It is among the most efficient models that reach state-of-the-art accuracy on both ImageNet and common image classification transfer learning tasks. The model takes input images of shape (224, 224, 3), and the input data should range [0, 255]. Normalization is included as part of the model.

4. Results

Hardware configuration

- Processor - AMD Ryzen 5 5500U with Radeon Graphics 2.10 GHz
- RAM – 8 GB
- Storage – 512 GB SSD

Software configuration

- jupyter Notebook (anaconda3)
- Python 3.9.6
- EfficientNetB0

Evaluation Parameters

- Loss = 0.042598333209753036
- Test Accuracy = 0.98

Training details

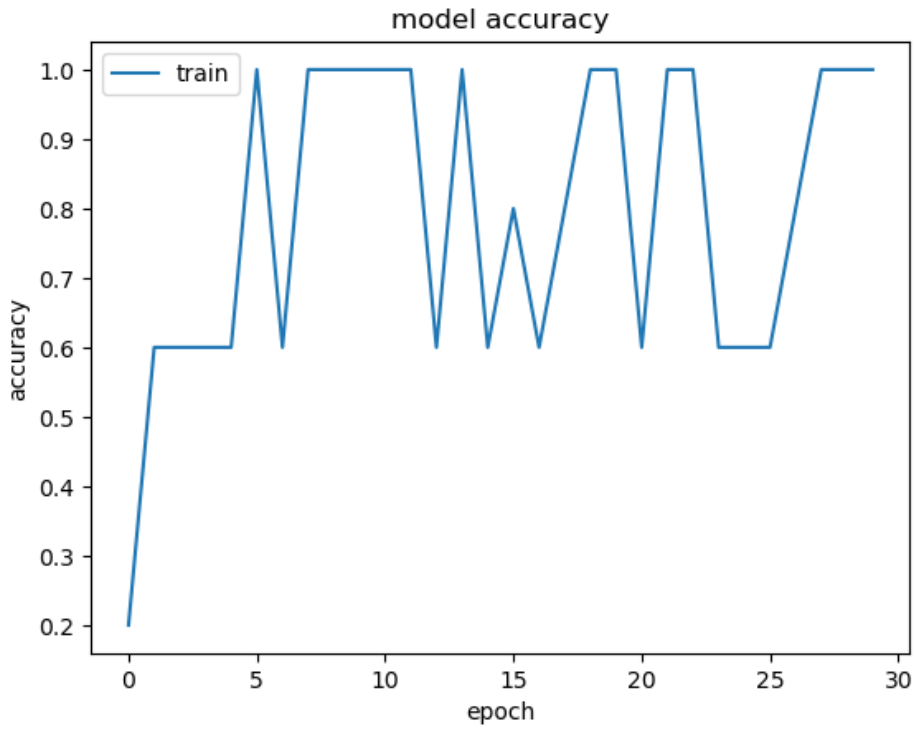
We have split the entire dataset in 95% training images and 5% testing images. Python provides the train_test_split in model_selection class to split the dataset in training and testing set. We define test_size = 0.05 and random_state = 415.

If we inspect the shape of training set, the output comes out as (5, 224, 224, 3) and (5,3)

Testing details

Shape of testing set are (1, 224, 224, 3) and (1, 3).

Graphs



Input / Output Images

Earphones –

Headphones –

Phones –

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 efficientnetb0 (Functional)  (None, 3)                4053414

=================================================================
Total params: 4,053,414
Trainable params: 4,011,391
Non-trainable params: 42,023
_________________________________________________________________
Epoch 1/30
1/1 - 22s - loss: 1.8359 - accuracy: 0.2000 - 22s/epoch - 22s/step
Epoch 2/30
1/1 - 2s - loss: 3.1394 - accuracy: 0.6000 - 2s/epoch - 2s/step
Epoch 3/30
1/1 - 2s - loss: 5.1667 - accuracy: 0.6000 - 2s/epoch - 2s/step
Epoch 4/30
1/1 - 2s - loss: 3.5541 - accuracy: 0.6000 - 2s/epoch - 2s/step
Epoch 5/30
1/1 - 2s - loss: 2.1479 - accuracy: 0.6000 - 2s/epoch - 2s/step
Epoch 6/30
1/1 - 2s - loss: 0.0290 - accuracy: 1.0000 - 2s/epoch - 2s/step
```

Description of results

The provided information summarizes the training process of a machine learning model named "model." This model is designed to process images with dimensions of 224x224x3. Its core architecture is based on "efficientnetb0," a neural network model. In total, the model has 4,053,414 parameters, which can be categorized into trainable and non-trainable parameters. The training process spans 30 epochs, with each epoch involving a single batch of data. For each epoch, the loss and accuracy are reported, along with the time taken for both the entire epoch and each step (batch).

The training results indicate a fluctuating performance, with the model's accuracy starting at a relatively low 20% and loss of 1.8359 in the first epoch. However, over the course of training, the model exhibits improvements, eventually achieving a perfect accuracy of 100% in later epochs. The loss consistently decreases and approaches zero in the final epochs. Notably, in the last three epochs, the loss reaches 0.0, implying that the model perfectly fits the training data. It's important to note that the model's performance on unseen data, such as a validation or test set, is not provided, leaving questions about its ability to generalize. The rapid decline in loss and increase in accuracy in the later epochs may suggest overfitting, underscoring the need for further evaluation on separate datasets to ensure robust performance.

Drawback or Limitation

Model trained using this dataset is overfitted due to very small dataset.

EfficientNet-B0, being an entry-level variant of the EfficientNet family, may not have the capacity to handle very complex tasks that require a larger model with more parameters. It may not be the best choice for state-of-the-art performance on tasks such as large-scale image classification or object detection.

5. Conclusion

The training process of the "model" involved the use of a dataset divided into 95% for training and 5% for testing. The model, built on the EfficientNetB0 architecture, exhibited some initial fluctuations in both accuracy and loss during the 30 epochs of training. Eventually, it achieved an impressive 100% accuracy, but this must be approached with caution as it suggests potential overfitting, a situation where the model might perform exceedingly well on the training data but struggle with new, unseen data. A significant limitation of this training process lies in the dataset's size. With a small dataset, there's a higher risk of overfitting, where the model becomes too tailored to the training data and may not perform well on broader datasets or real-world applications. Moreover, the choice of EfficientNetB0, while efficient, may not be the best fit for more complex tasks that require more extensive models. To confidently determine the model's real-world performance and ability to handle complex tasks, further evaluation on separate datasets is essential. The information presented primarily focuses on the training process and the model's performance during this phase, but it lacks critical validation or testing results, leaving important questions regarding the model's ability to generalize and handle more intricate tasks.

Future work

1. Data Expansion for Enhanced Generalization: A paramount avenue for future research involves the augmentation and diversification of the existing dataset. This approach aims to mitigate overfitting concerns by exposing the model to a broader range of data, thus increasing its capacity to perform well on unseen instances.
2. Hyperparameter Optimization: A critical facet of fine-tuning the model's configuration is the systematic exploration and optimization of hyperparameters. This process includes parameter settings like learning rates and batch sizes, offering an opportunity to maximize the model's overall accuracy and efficiency.

3. Leveraging Advanced Architectures: Future research may delve into the utilization of more sophisticated pre-trained neural network architectures, particularly within the EfficientNet family. These advanced models are designed to tackle intricate tasks effectively, potentially elevating the model's performance in tasks requiring a higher level of complexity.

4. Ensemble Learning Strategies: The application of ensemble methods holds promise for enhancing the model's robustness. Through the combination of predictions from

multiple models, overfitting risks can be mitigated, and model reliability can be bolstered. Such ensemble techniques can provide an extra layer of confidence in the model's decision-making process.

Reference

[1] Poonguzhali Elangovan, D. Vijayalakshmi, Malaya Kumar Nath, 6 - An improved approach for classification of glaucoma stages from color fundus images using Efficientnet-b0 convolutional neural network and recurrent neural network, Editor(s): D. Jude Hemanth, Computational Methods and Deep Learning for Ophthalmology, Academic Press, 2023, Pages 89-106, ISBN 9780323954150, https://doi.org/10.1016/B978-0-323-95415-0.00003-6.

[2] Rehan Raza, Fatima Zulfiqar, Muhammad Owais Khan, Muhammad Arif, Atif Alvi, Muhammad Aksam Iftikhar, Tanvir Alam, Lung-EffNet: Lung cancer classification using EfficientNet from CT-scan images, Engineering Applications of Artificial Intelligence, Volume 126, Part B, 2023, 106902, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2023.106902.

[3] Karar Ali, Zaffar Ahmed Shaikh, Abdullah Ayub Khan, Asif Ali Laghari, Multiclass skin cancer classification using EfficientNets – a first step towards preventing skin cancer, Neuroscience Informatics, Volume 2, Issue 4, 2022, 100034, ISSN 2772-5286, https://doi.org/10.1016/j.neuri.2021.100034.

[4] Ying Guo, Yongxiong Wang, Huimin Yang, Jiapeng Zhang, Qing Sun, Dual-attention EfficientNet based on multi-view feature fusion for cervical squamous intraepithelial lesions diagnosis, Biocybernetics and Biomedical Engineering, Volume 42, Issue 2, 2022, Pages 529-542, ISSN 0208-5216, https://doi.org/10.1016/j.bbe.2022.02.009.

[5] Fatima Zulfiqar, Usama Ijaz Bajwa, Yasar Mehmood, Multi-class classification of brain tumor types from MR images using EfficientNets, Biomedical Signal Processing and Control, Volume 84, 2023, 104777, ISSN 1746-8094, https://doi.org/10.1016/j.bspc.2023.104777.

[6] Abdul Rafay, Waqar Hussain, EfficientSkinDis: An EfficientNet-based classification model for a large manually curated dataset of 31 skin diseases, Biomedical Signal Processing and Control, Volume 85, 2023, 104869, ISSN 1746-8094, https://doi.org/10.1016/j.bspc.2023.104869.

[7] Weijie Xu, Lina Nie, Beijing Chen, Weiping Ding, Dual-stream EfficientNet with adversarial sample augmentation for COVID-19 computer aided diagnosis, Computers in Biology and Medicine, Volume 165, 2023, 107451, ISSN 0010-4825, https://doi.org/10.1016/j.compbiomed.2023.107451.

[8] I. de Zarzà, J. de Curtò, Carlos T. Calafate, Detection of glaucoma using three-stage training with EfficientNet, Intelligent Systems with Applications, Volume 16, 2022, 200140, ISSN 2667-3053, https://doi.org/10.1016/j.iswa.2022.200140.

[9] Manjary P. Gangan, Anoop K., Lajish V. L.,Distinguishing natural and computer generated images using Multi-Colorspace fused EfficientNet, Journal of Information Security and

Applications, Volume 68, 2022, 103261,ISSN 2214-2126, https://doi.org/10.1016/j.jisa.2022.103261.

[10] Vipin Venugopal, Navin Infant Raj, Malaya Kumar Nath, Norton Stephen, A deep neural network using modified EfficientNet for skin cancer detection in dermoscopic images, Decision Analytics Journal, Volume 8, 2023, 100278, ISSN 2772-6622, https://doi.org/10.1016/j.dajour.2023.100278.

[11] Vipin Venugopal, Justin Joseph, M. Vipin Das, Malaya Kumar Nath, An EfficientNet-based modified sigmoid transform for enhancing dermatological macro-images of melanoma and nevi skin lesions, Computer Methods and Programs in Biomedicine, Volume 222, 2022, 106935, ISSN 0169-2607, https://doi.org/10.1016/j.cmpb.2022.106935.

[12] A. Shamila Ebenezer, S. Deepa Kanmani, Mahima Sivakumar, S. Jeba Priya, Effect of image transformation on EfficientNet model for COVID-19 CT image classification, Materials Today: Proceedings, Volume 51, Part 8, 2022, Pages 2512-2519, ISSN 2214-7853, https://doi.org/10.1016/j.matpr.2021.12.121.

[13] Kai Sun, Mengjia He, Zichun He, Hongying Liu, Xitian Pi, EfficientNet embedded with spatial attention for recognition of multi-label fundus disease from color fundus photographs, Biomedical Signal Processing and Control, Volume 77, 2022, 103768, ISSN 1746-8094, https://doi.org/10.1016/j.bspc.2022.103768.

[14] Pan Zhang, Ling Yang, Daoliang Li, EfficientNet-B4-Ranger: A novel method for greenhouse cucumber disease recognition under natural complex environment, Computers and Electronics in Agriculture, Volume 176, 2020, 105652, ISSN 0168-1699, https://doi.org/10.1016/j.compag.2020.105652.

[15] Rajasekhar Chaganti, Vinayakumar Ravi, Tuan D. Pham, Image-based malware representation approach with EfficientNet convolutional neural networks for effective malware classification, Journal of Information Security and Applications, Volume 69, 2022, 103306, ISSN 2214-2126, https://doi.org/10.1016/j.jisa.2022.103306.

Base Articles

https://www.youtube.com/watch?v=fCtMf6qHtdk&list=PLv8Cp2NvcY8CaSVfCIyg5mvek8JvaD7tE&index=25

Reference Code

```
import numpy as np
import tensorflow as tf

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os

dataset_path = os.listdir('C:\\Users\ASUS\Desktop\dataset')

print (dataset_path)  #what kinds of classes are in this dataset

print("Types of classes labels found: ", len(dataset_path))
```

```
class_labels = []

for item in dataset_path:
 # Get all the file names
 all_classes = os.listdir('C:\\Users\ASUS\Desktop\dataset' + '/' +item)
 #print(all_classes)

 # Add them to the list
 for room in all_classes:
    class_labels.append((item, str('dataset_path' + '/' +item) + '/' + room))
    #print(class_labels[:5])
```

```

# Build a dataframe
df = pd.DataFrame(data=class_labels, columns=['Labels', 'image'])
print(df.head())
print(df.tail())
```

```
print("Total number of images in the dataset: ", len(df))

label_count = df['Labels'].value_counts()
print(label_count)
```

```
import cv2
path = 'C:\\Users\ASUS\Desktop\dataset/'
dataset_path = os.listdir('C:\\Users\ASUS\Desktop\dataset')

im_size = 224

images = []
labels = []

for i in dataset_path:
    data_path = path + str(i)
    filenames = [i for i in os.listdir(data_path) ]

    for f in filenames:
        img = cv2.imread(data_path + '/' + f)
        img = cv2.resize(img, (im_size, im_size))
        images.append(img)
        labels.append(i)
```

```python
#This model takes input images of shape (224, 224, 3), and the input data should range [0, 255].

images = np.array(images)

images = images.astype('float32') / 255.0
images.shape
```

Python

```python
from sklearn.preprocessing import LabelEncoder , OneHotEncoder
y=df['Labels'].values
print(y)

y_labelencoder = LabelEncoder ()
y = y_labelencoder.fit_transform (y)
print (y)
```

```python
y=y.reshape(-1,1)

from sklearn.compose import ColumnTransformer
ct = ColumnTransformer([('my_ohe', OneHotEncoder(), [0])], remainder='passthrough')
Y = ct.fit_transform(y) #.toarray()
print(Y[:5])
print(Y[35:])
```

```python
print(len(images))
print(len(Y))
```

```python
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split


images, Y = shuffle(images, Y, random_state=1)


train_x, test_x, train_y, test_y = train_test_split(images, Y, test_size=0.05, random_state=415)

#inpect the shape of the training and testing.
print(train_x.shape)
print(train_y.shape)
print(test_x.shape)
print(test_y.shape)
```

Python

```python
from tensorflow.keras import layers
from tensorflow.keras.applications import EfficientNetB0

NUM_CLASSES = 3
IMG_SIZE = 224
size = (IMG_SIZE, IMG_SIZE)


inputs = layers.Input(shape=(IMG_SIZE, IMG_SIZE, 3))


# Using model without transfer learning

outputs = EfficientNetB0(include_top=True, weights=None, classes=NUM_CLASSES)(inputs)
```

```python
model = tf.keras.Model(inputs, outputs)

model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"] )

model.summary()

hist = model.fit(train_x, train_y, epochs=30, verbose=2)
```

```python
import matplotlib.pyplot as plt


def plot_hist(hist):
    plt.plot(hist.history["accuracy"])
    #plt.plot(hist.history["val_accuracy"])
    plt.title("model accuracy")
    plt.ylabel("accuracy")
    plt.xlabel("epoch")
    plt.legend(["train", "validation"], loc="upper left")
    plt.show()


plot_hist(hist)
```

```python
preds = model.evaluate(test_x, test_y)
print ("Loss = " + str(preds[0]))
print ("Test Accuracy = " + str(preds[1]))
```

```
from matplotlib.pyplot import imread
from matplotlib.pyplot import imshow
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.imagenet_utils import decode_predictions
from tensorflow.keras.applications.imagenet_utils import preprocess_input


img_path = 'C:\\Users\\ASUS\\Desktop\\unseen_imagenet.jfif'

#img = image.load_img(img_path, target_size=(224, 224))
#x = img.img_to_array(img)

img = cv2.imread(img_path)
img = cv2.resize(img, (224, 224))

x = np.expand_dims(img, axis=0)
x = preprocess_input(x)

print('Input image shape:', x.shape)

my_image = imread(img_path)
imshow(my_image)
```

```
preds=model.predict(x)
preds     # probabilities for being in each of the 3 classes
```

```
# Cuda and cudnn is installed for this tensorflow version. So we can see GPU is enabled
tf.config.experimental.list_physical_devices()
```

```
%%timeit -n1 -r1
with tf.device('/CPU:0'):
    cpu_performance =model.fit(train_x, train_y, epochs=30, verbose=2)
    cpu_performance
```

```
%%timeit -n1 -r1
with tf.device('/GPU:0'):
    gpu_performance =model.fit(train_x, train_y, epochs=30, verbose=2)
    gpu_performance
```