**Image Classification using 3 models ResNet , Xception & DenseNet**

In this documentation, we will explore the fascinating world of deep learning architectures used for classification tasks. From the basics of each architecture to their applications, challenges, and performance comparison, we will dive into the key components and processes that make them powerful for image classification. Get ready to uncover the strengths and potential of each model—ResNet, Xception, and DenseNet—in the context of this specific dataset and task.

|  |  |
| --- | --- |
| **ID** | **Name** |
| **20210065** | **Ahmed Tarek Abdelwali Ahmed** |
| **20210080** | **Ahmed Alaa Abassi** |
| **20210058** | **Ahmed Said Mohamed Said** |
| **20210044** | **Ahmed Khaled Amin Kadi** |
| **20210119** | **Ahmed Medhat Abobakr** |

***Academic publications:***

**First Paper :**

**Deep Residual Learning for Image Recognition**

* ***Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun*;**

**Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers---8x deeper than VGG nets but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers. The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions, where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.**

**Link :** [**CVPR 2016 Open Access Repository**](https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html)

**Second Paper :**

**Xception: Deep Learning With Depthwise Separable Convolutions**

* ***Francois Chollet*;**

**We present an interpretation of Inception modules in convolutional neural networks as being an intermediate step in-between regular convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution). In this light, a depthwise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation leads us to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depthwise separable convolutions. We show that this architecture, dubbed Xception, slightly outperforms Inception V3 on the ImageNet dataset (which Inception V3 was designed for), and significantly outperforms Inception V3 on a larger image classification dataset comprising 350 million images and 17,000 classes. Since the Xception architecture has the same number of parameters as Inception V3, the performance gains are not due to increased capacity but rather to a more efficient use of model parameters.**

**Link :** [**CVPR 2017 Open Access Repository**](https://openaccess.thecvf.com/content_cvpr_2017/html/Chollet_Xception_Deep_Learning_CVPR_2017_paper.html)

**Third Paper :**

**Densely Connected Convolutional Networks**

* ***Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger***

**Recent work has shown that convolutional networks can be substantially deeper, more accurate, and efficient to train if they contain shorter connections between layers close to the input and those close to the output. In this paper, we embrace this observation and introduce the Dense Convolutional Network (DenseNet), which connects each layer to every other layer in a feed-forward fashion. Whereas traditional convolutional networks with L layers have L connections--one between each layer and its subsequent layer--our network has L(L+1)/2 direct connections. For each layer, the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers. DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters. We evaluate our proposed architecture on four highly competitive object recognition benchmark tasks (CIFAR- 10, CIFAR-100, SVHN, and ImageNet). DenseNets obtain significant improvements over the state-of-the-art on most of them, whilst requiring less memory and computation to achieve high performance.**

**Link :**[**CVPR 2017 Open Access Repository**](https://openaccess.thecvf.com/content_cvpr_2017/html/Huang_Densely_Connected_Convolutional_CVPR_2017_paper.html)

**Resource :**

**DenseNet-201 and Xception Pre-Trained Deep Learning Models for Fruit Recognition**

With the dramatic increase of the global population and with food insecurity increasing, it has become a major concern for both individuals and governments to fulfill the need for foods such as vegetables and fruits. Moreover, the desire for the consumption of healthy food, including fruit, has increased the need for applications in the field of agriculture that help to achieve better methods for fruit sorting and fruit disease prediction and classification. Automated fruit recognition is a potential solution to reduce the time and labor required to identify different fruits in situations such as retail stores during checkout, fruit processing centers during sorting, and orchards during harvest. Automating these processes reduces the need for human intervention, making them cheaper, faster, and immune to human error and biases. Past research in the field has focused mainly on the size, shape, and color features of fruits or employed convolutional neural networks (CNNs) for their classification. This study investigates the effectiveness of pre-trained deep learning models for fruit classification using two distinct datasets: Fruits-360 and the Fruit Recognition dataset. Four pre-trained models, DenseNet-201, Xception, MobileNetV3-Small, and ResNet-50, were chosen for the experiments based on their architecture and features. The results show that all models achieved almost 99% accuracy or higher with Fruits-360. With the Fruit Recognition dataset, DenseNet-201 and Xception achieved accuracies of around 98%. The good results exhibited by DenseNet-201 and Xception on both the datasets are remarkable, with DenseNet-201 attaining accuracies of 99.87% and 98.94%, and Xception attaining 99.13% and 97.73% accuracy, respectively, on Fruits-360 and the Fruit Recognition dataset.

Link : [DenseNet-201 and Xception Pre-Trained Deep Learning Models for Fruit Recognition](https://www.mdpi.com/2079-9292/12/14/3132)

**Resource :**

**The Application of Improved DenseNet Algorithm in Accurate Image Classification**

Image recognition technology belongs to an important research field of artificial intelligence. In order to enhance the application value of image recognition technology in the field of computer vision and improve the technical dilemma of image recognition, the research improves the feature reuse method of dense convolutional network. Based on gradient quantization, traditional parallel algorithms have been improved. This improvement allows for independent parameter updates layer by layer, reducing communication time and data volume. The introduction of quantization error reduces the impact of gradient loss on model convergence. The test results show that the improvement strategy designed by the research improves the model parameter efficiency while ensuring the recognition effect. Narrowing the learning rate is conducive to refining the updating granularity of model parameters, and deepening the number of network layers can effectively improve the final recognition accuracy and convergence effect of the model. It is better than the existing state-of-the-art image recognition models, visual geometry group and EfficientNet. The parallel acceleration algorithm, which is improved by the gradient quantization, performs better than the traditional synchronous data parallel algorithm, and the improvement of the acceleration ratio is obvious. Compared with the traditional synchronous data parallel algorithm and stale synchronous parallel algorithm, the optimized parallel acceleration algorithm of the study ensures the image data training speed and solves the bottleneck problem of communication data. The model designed by the research improves the accuracy and training speed of image recognition technology and expands the use of image recognition technology in the field of computer vision.Please confirm the affiliation details of [1] is correct.The relevant detailed information in reference [1] has been confirmed to be correct.

Link :[The application of improved densenet algorithm in accurate image recognition | Scientific Reports](https://www.nature.com/articles/s41598-024-58421-z)

**Resnet**

**Input Layer:**

The image data is passed through initial convolutional layers with filters to extract low-level features like edges and textures.

**Residual Blocks:**

* The core idea of ResNet is the **residual connection**:
  1. A shortcut connection bypasses one or more convolutional layers.
  2. The output of these layers is added to the input (skip connection).
* This "shortcut" allows the model to learn the residual (difference) instead of the full transformation, making it easier to train deeper networks.

**Layer Composition:**

* Each block has :
  1. **Convolution Layers**: Extract features.
  2. **Batch Normalization**: Normalize the activations to speed up training.
  3. **Activation Function (ReLU)**: Introduces non-linearity.
* Skip connections combine the input with the output of these layers.

**Stacking Residual Blocks:**

* Multiple residual blocks are stacked to form the ResNet architecture.
* The number of blocks depends on the variant (e.g., ResNet-18, ResNet-50).

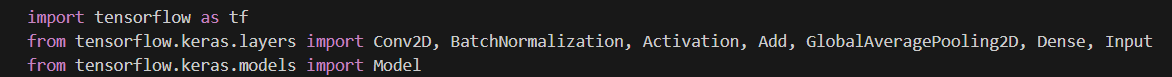
**Global Average Pooling:**

* After feature extraction, the output is averaged across spatial dimensions, reducing it to a vector.

**Fully Connected Layer:**

* A dense layer maps the features to the output classes using softmax for classification.

**Import Libraries**



**Basic Block**

**Implements the basic block for ResNet, the fundamental building unit which are :**

* First Convolutional Layer
* Second Convolutional Layer
* Skip Connection Logic

A screen shot of a computer program

Description automatically generated

**Resnet Layer**

A screen shot of a computer code

Description automatically generated

**ResNet Model Architecture**

Input Layer

* The input shape of the images.

Initial Convolutional Layer

* Applies a large 7x7 convolution for initial feature extraction.
* Downsamples the input with stride 2 and a max-pooling operation.

ResNet Layers

* Adds four layers of varying depth (blocks) and filter sizes.
* Gradually increases complexity and reduces spatial dimensions.

Global Average Pooling

* Reduces each feature map to a single value.

Output Layer

* Final dense layer with softmax activation for classification

A computer screen shot of a program

Description automatically generated

**Confusion Matrix**

A blue squares with white text

Description automatically generated

**Roc & Auc Curve**

A graph of a number of colored lines

Description automatically generated

# ***Xception***

**Input Layer:**

* The image data is passed through an initial convolution layer for basic feature extraction.

**Depthwise Separable Convolutions:**

* The main innovation in Xception:
  + **Depthwise Convolution**: Applies a single filter per input channel (lightweight spatial filtering).
  + **Pointwise Convolution**: Uses 1x1 convolutions to combine the filtered outputs and introduce cross-channel interactions.
* This two-step process reduces the computational cost while maintaining high accuracy.

**Stacking Separable Convolutions:**

* Similar to Inception, multiple convolutional modules are stacked, but Xception replaces the complex branching structure with separable convolutions.

**Residual Connections:**

* Like ResNet, Xception includes skip connections to help gradients flow during backpropagation and ease training of deep layers.

**Global Average Pooling:**

* Outputs from the last convolutional layers are averaged spatially to form a feature vector.

**Fully Connected Layer:**

* The dense layer maps the feature vector to class probabilities using softmax.

***Transfer Learning Model Function***

**Freeze Pre-trained Layers**

**Add Custom Layers**

* GlobalAveragePooling2D(): Reduces dimensions while retaining important features.
* Dropout(0.5): Helps prevent overfitting by deactivating neurons randomly.
* Dense(1024, activation='relu'): A fully connected layer to learn more task-specific patterns.
* Dense(len(le.classes\_), activation='softmax'): Outputs probabilities for each class.

**Compile the Model**

A computer code on a black background

Description automatically generated

***Initializing & Training the model***

**Base Model**

* weights="imagenet": Loads pre-trained weights.
* include\_top=False: Removes the default dense layers.
* input\_shape=(size, size, 3): Defines the input dimensions for the dataset.

**Transfer Learning**: Combines the frozen base model with the new custom layers.

A screenshot of a computer

Description automatically generated

***Fine Tuning***

**-Unfreeze the Last Few Layers**

* Allows the last 10 layers to learn task-specific features by unfreezing them.

**-Recompile the Model**

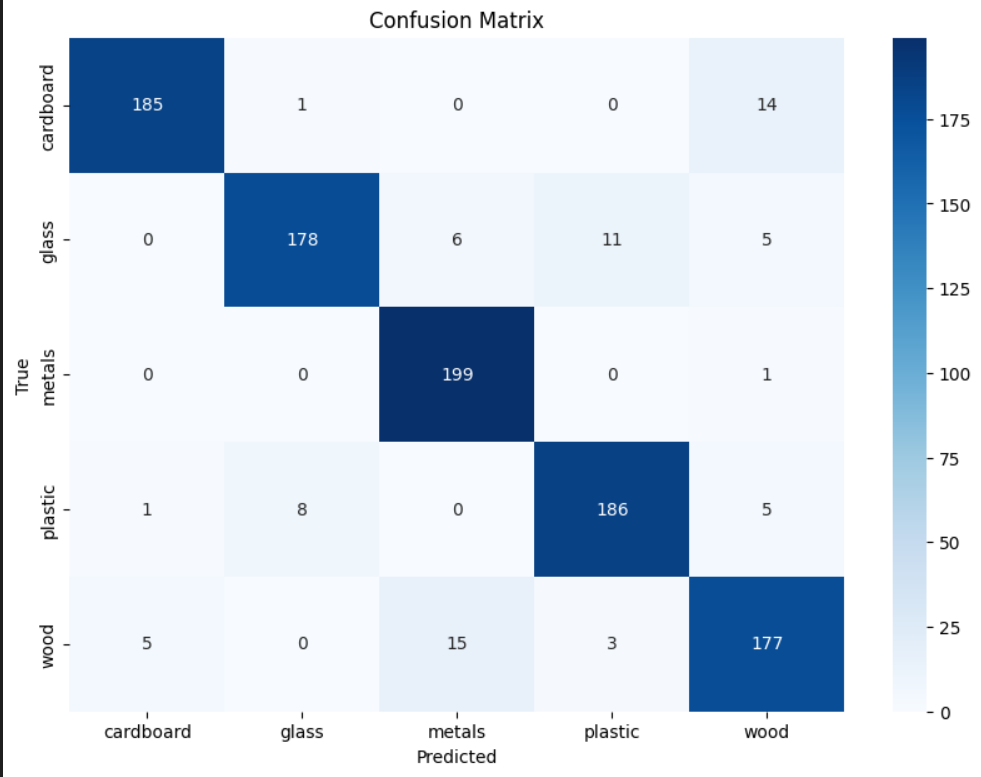
**-Fine-Tune the Model**

* Updates the unfrozen layers for better accuracy.
* Stops early if validation accuracy stagnates.

A black background with many colorful lights

Description automatically generated with medium confidence

**Confusion Matrix**



**Roc & AUC Curve**

A graph of a curve

Description automatically generated

# **Densenet**

**Input Layer:**

* The image data is passed through an initial convolution layer to extract basic features.

**Dense Blocks:**

* DenseNet's key feature:
  1. Every layer is **directly connected** to every other layer within the same block.
  2. This means each layer receives inputs from all preceding layers and passes its outputs to all subsequent layers.
* This ensures feature reuse and reduces redundant computations.

**Layer Composition:**

* Each dense block contains:
  1. **Batch Normalization**: Normalizes activations.
  2. **ReLU Activation**: Introduces non-linearity.
  3. **Convolution Layer**: Extracts features.
* The outputs are concatenated, not added, preserving all features.

**Transition Layers:**

* Between dense blocks, **transition layers** are added:
  + These include convolution and pooling layers to reduce the feature map size and prevent overfitting.

**Global Average Pooling:**

* As with the other architectures, feature maps are averaged spatially to create a compact feature vector.

**Fully Connected Layer:**

* Finally, a dense layer produces the class probabilities using softmax.

**Loading & Customizing Pre-trained DenseNet Model**

**-Load the pre-trained DenseNet121 model, excluding its top layers.**

* weights='imagenet': Uses pre-trained weights.
* include\_top=False: Excludes the fully connected layers at the top, as we will add our own for classification.
* input\_shape=(256, 256, 3): Specifies the input size for the images (256x256 pixels with 3 color channels).

**-Freezing the Pre-trained Base Model**

* We freeze the pre-trained base model by setting trainable = False.

**-Adding Custom Classification Layers**

* **Global Average Pooling**: Converts the 2D output of the base model into a 1D vector by averaging over the spatial dimensions.
* **Dropout**: Regularization technique to prevent overfitting by randomly setting 50% of the neurons to zero during training.
* **Dense Layer**: Fully connected layer with 1024 units and ReLU activation, followed by another Dense layer for final classification. The output layer uses softmax activation to handle multi-class classification.

A screen shot of a computer program

Description automatically generated

**Compiling & Training the Model**

We train the model for the first 10 epochs with early stopping enabled. Early stopping monitors the validation accuracy, and if it does not improve for 5 epochs, it will stop training to avoid overfitting.

A screen shot of a computer

Description automatically generated

**Fine Tuning**

After the initial training phase, we unfreeze the last 10 layers of the DenseNet model for fine-tuning.

Then We train the model again for 10 more epochs with early stopping enabled. Now, the weights of both the new layers and the fine-tuned layers will be updated.

A computer screen shot of text

Description automatically generated

**Confusion Matrix**

**A screenshot of a graph

Description automatically generated**

**Roc & AUC Curve**

A graph of a curve

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| Point of comparison | ResNet | Xception | DenseNet |
| Pros | * Strong in handling vanishing gradient problems through residual connections. * Very effective with deeper architectures (e.g., ResNet50, ResNet101) for high-level feature extraction. * Great for object detection in noisy or variable conditions | * Efficient for transfer learning with pre-trained weights (ImageNet). * Handles large variations in object shapes, making it suitable for detecting various raw materials. * Good performance in fine-tuning, allowing adaptation to specific recycling-related tasks. | * Dense connections improve the flow of information and gradients, leading to faster convergence. * Excellent for detecting subtle features in images, especially if fine details matter. * Less prone to overfitting compared to traditional architectures, even with fewer layers |
| Cons | * Requires longer training times for deeper models. * May overfit on small datasets unless augmented correctly. | * Computationally expensive due to its depth and complexity. * May require a large dataset for optimal performance. | * High memory consumption, especially for deeper networks. * Computational cost can be a bottleneck if you're working with large input sizes or complex tasks. |

|  |  |  |  |
| --- | --- | --- | --- |
| POC | ResNet | Xception |  |
| Accuracy |  |  |  |
| Loss Curve |  |  |  |

|  |  |
| --- | --- |
|  | DenseNet |
| Accuracy |  |
| Loss Curve |  |

So from the previous table we conclude that in relatively small data models like ResNet might struggle without fine-tuning or augmentations due to its complexity.

DenseNet could be more favorable since it uses dense connections that improve learning even with limited data. Which is proven in our project where DenseNet have accuracy precenatge of 91% . Due to its balance of performance and efficiency in recognizing subtle features in small datasets like ours.

While in xception it could excel in precision and recall metrics, which are especially crucial for tasks where you don’t want to misclassify (false positives/negatives).

However, both Xception and DenseNet demonstrate strong performance, with DenseNet leveraging its dense connections for efficient feature learning and Xception utilizing its depthwise separable convolutions for capturing complex patterns. This is evident in our project, where Xception achieves an accuracy percentage of 92%, making it nearly equal to DenseNet’s 91%. This balance highlights the capability of both models to effectively recognize subtle variations in small datasets like ours.