Precision Livestock

Journal Homepage: sciencesforce.com/pl

Precision Liv. Vol. 2 (2025) 1–10

Paper Type: Original Article

Optimizing Animal Classification through Convolutional Neural Networks

Abdulrahman Aladdin Sobhi [1], Abdulrahman Mohamed Abbas [1], Tareef S. Alkellezli [2], and Mohamed Eassa [1,3,*]

[1] Faculty of Information Systems and Computer Science, October 6th University, Giza, 12585, Egypt.
Emails: 212102823@o6u.edu.eg; 212103294@o6u.edu.eg; mohamed.eassa.cs@o6u.edu.eg.
[2] College of Engineering, Ashur University, Baghdad, Iraq; tareef.alkellezli@au.edu.iq.
[3] Applied Science Research Center. Applied Science Private University, Amman.

Received: 23 Aug 2024 **Revised:** 16 Dec 2024 **Accepted:** 06 Jan 2025 **Published:** 08 Jan 2025

Abstract

Efficient and accurate wildlife monitoring in their natural habitats is crucial for conservation efforts. This project introduces an algorithm to identify wildlife species using image recognition, facilitating streamlined monitoring processes. Manual identification proves challenging with a diverse range of species, making automated classification essential. The application of robust deep learning algorithms enables effective wildlife recognition and classification, aiding in the prevention of wildlife-vehicle collisions, tracking of animals, and mitigation of poaching activities. By leveraging a dataset of wildlife images, the algorithm learns to categorize animals accurately, contributing to the preservation of biodiversity. Moreover, the proposed approach demonstrates promising results in classification accuracy, suggesting its potential to enhance existing wildlife conservation practices. The integration of such technological advancements into conservation strategies offers a scalable and efficient solution to address challenges in wildlife management. This research underscores the importance of employing innovative technologies to safeguard vulnerable species and their habitats in the face of increasing human-wildlife interactions and environmental threats.

Keywords: Animal Classes; CNN; Deep Learning; Decision Making; Image Processing; Resnet50; Transfer Learning.

1 | Introduction

In recent times, amidst the rapid proliferation of digital content, the automated categorization of images has emerged as a formidable challenge within the realm of visual data organization and retrieval systems. At the heart of this endeavor lies computer vision, an interdisciplinary subfield of artificial intelligence (AI) dedicated to endowing machines with human-like capabilities to interpret and understand visual information. Despite concerted research efforts to tackle these challenges, traditional methods have often fixated on the superficial aspects of image constituents, neglecting the broader context necessary for effective image processing. This myopic focus on low-level features has proven insufficient for grappling with the complexity inherent in image classification tasks.

Corresponding Author: mohamed.eassa.cs@o6u.edu.eg

https://doi.org/10.61356/j.pl.2025.2458

Indeed, image classification has long remained a thorny issue in the field of computer vision. While humans effortlessly decipher and categorize images, the computational demands posed by image data are formidable. Each image comprises a multitude of pixels, each imbued with distinct attributes, necessitating vast storage capacities and computational resources for processing. The sheer volume of computations required for image classification poses a formidable barrier to real-time decision-making, underscoring the need for more efficient algorithms and computational architectures.

Deep learning algorithms arrive, signaling the beginning of a new age in computer vision. Convolutional neural networks (CNNs) are the most advanced of these; their capacity to recognize complex patterns and characteristics in images has revolutionized image categorization tasks. By utilizing CNN's deep learning capabilities, scientists have made impressive progress in a variety of applications, ranging from determining the caliber of wood panels to recognizing livestock in pictures. CNN's capacity to autonomously learn and adapt from vast datasets offers a significant advantage, eliminating the need for human intervention in feature extraction.

Remarkably, CNN's revolutionary influence goes beyond scholarly investigations and penetrates practical uses like Google's autonomous vehicles and Facebook's facial recognition technologies. These innovative initiatives show the significant influence of deep learning in influencing the direction of computer vision. They represent the merging of state-of-the-art AI technologies with useful problem-solving techniques. The possibilities for applications in domains ranging from industrial quality control to wildlife conservation are endless as academics continue to push the limits of AI-driven image processing, signaling the start of a new era of innovation and discovery.

2 | Literature Review

2.1 | Convolutional Neural Networks

This paper's authors [3] present an automated underwater fish species classification method. Traditional approaches focus on classifying fishes in non-submerged environments, as underwater classification presents several challenges, including background smoothness, distortion, object segmentation, image quality, and obstacles. The proposed method suggests preprocessing the dataset to remove smoothness, utilizing image processing techniques to eliminate underwater obstacles, debris, and non-fish entities from the images. Subsequently, a Deep Learning approach is employed, leveraging Convolutional Neural Networks (CNNs) for fish species classification. The authors conducted a comparison of activation functions including ReLU, SoftMax, and tanh, with the ReLU activation function being identified as highly accurate.

2.2 | VGG16

The paper[4] delves into the application of VGG16 for Plant Classification, coupled with techniques such as Data Augmentation and Transfer Learning. Leveraging transfer learning and Convolutional Neural Networks (CNNs), the study aims to classify various plant species using leaf images, which are rich in low-level features such as shape and color, commonly utilized in plant recognition models. However, relying solely on leaf images poses a significant challenge in accurately classifying different plant species based on a single feature or criterion.

The work uses Data Augmentation, dropout, and transfer learning strategies to lessen this difficulty. These strategies successfully tackle overfitting in short datasets, one of the most computationally demanding problems with CNNs. The work makes use of the abundance of information included in a pre-trained VGG net model that was trained on the ImageNet dataset. The generation of extra training examples through data augmentation is essential for enhancing the model's capacity to generalize over a variety of datasets.

Moreover, Data Augmentation ensures that the model encounters a diverse range of images during training, reducing the risk of overfitting and enhancing the model's ability to generalize to unseen data. These enhancements in training methodology contribute to the overall robustness and generalization capability of the model, ultimately improving its performance in plant species classification tasks.

2.3 | Resnet50

The deep convolutional neural network architecture known as ResNet50 has advanced computer vision significantly. Developed by Microsoft Research, ResNet50 is renowned for its exceptional performance in image recognition tasks, particularly in large-scale visual recognition challenges such as ImageNet. The 50-layer deep structure of the architecture and its creative use of residual connections, which enable the network to efficiently propagate gradients during training and mitigate the vanishing gradient issue, are its defining features. These residual connections enable the network to learn more complex features and deeper representations, leading to improved accuracy and generalization capabilities. ResNet50 is still used as a standard for evaluating deep learning performance and has grown to be a mainstay in several applications, such as semantic segmentation, object detection, and picture classification. Because of its cutting-edge performance, scalability, and resilience, ResNet50 is a popular option for handling a variety of computer vision applications.

2.4 | Transfer Learning

Transfer learning is a machine learning technique where a model learned on one task is used or modified to serve as the basis for a new, related activity. Transfer learning is the process of applying information from one problem to another that is related, rather than beginning the learning process from scratch.

The pre-trained model, which was frequently created on a sizable dataset for a particular task, is utilized in transfer learning as a feature extractor or as a jumping-off point for training on a different dataset or task. If it starts with the features it has learned from the first challenge, it will learn more quickly and effectively. Because the model would have already learned general qualities that would probably be helpful in the second job, this might also assist prevent overfitting.

3 | Proposed System

Introducing a novel system tailored to organize images of wildlife, offering indispensable assistance to biologists and researchers as they delve deeper into the analysis and refinement of habitat, ecological, and conservation strategies. Embedded within the conceptual framework, as depicted in Figure 1, lies the innovative architecture for Fauna Image Classification, leveraging the power of Convolutional Neural Networks (CNNs).

In our endeavor, we leverage the Animal-10 dataset [2] obtained from Kaggle [6] to fuel our model's training process. Our approach hinges on a Convolutional Neural Network, meticulously crafted with the Rectified Linear Unit (ReLU) activation function and inspired by the ResNet50.

The journey unfolds with an initial phase dedicated to feature extraction, employing the prowess of the pre-trained ResNet50 model. This stage is augmented by the integration of Image Processing techniques, strategically interwoven throughout dataset handling - from loading to testing, training, and validation. These techniques play a pivotal role in mitigating noise, refining artifacts, combating blurriness, and eliminating dust imperfections inherent within the images.

With the groundwork laid and the data refined, the Convolutional Neural Network, bolstered by the Leaky ReLU activation function, embarks on its mission to train the model. Its objective: is to masterfully and definitively discern and classify various animal classes with unparalleled accuracy and precision.

DataSet source [2]: Animals-10 (kaggle.com).

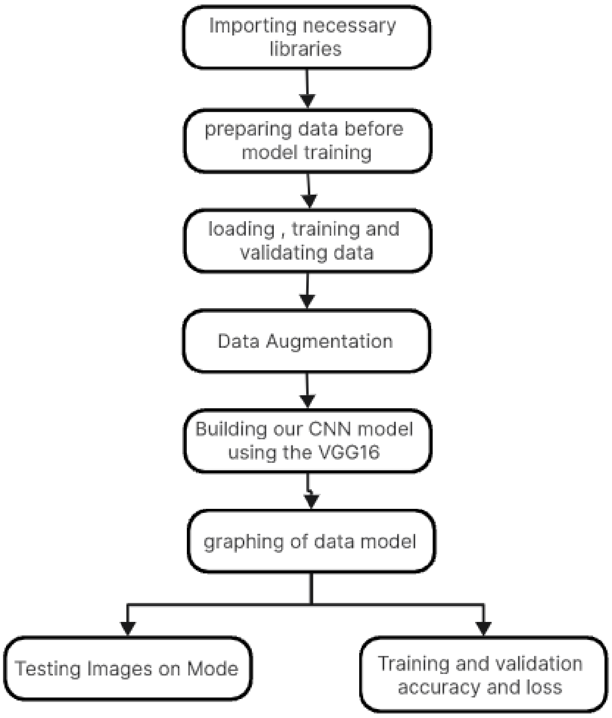

Figure 1. Design of proposed neural network.

3.1 | Importing Necessary Libraries

In the fields of data science and machine learning, libraries are vital because they offer the tools and features that make developing and implementing algorithms and models easier. The importance of various key libraries that are frequently used in data science and machine learning—such as pandas, numpy, matplotlib, os, keras, sklearn, and tensorflow—will be discussed in this article.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
import tensorflow as tf
from keras.preprocessing.image import load_img
from sklearn.model_selection import train_test_split
from keras.preprocessing.image import ImageDataGenerator
from keras import Sequential
from keras.layers import  Dense, Dropout
from keras.optimizers import Adam
from keras.layers import Input, Dense
from keras.models import load_model
from keras.preprocessing import image
from keras.applications import ResNet50
from keras.applications.resnet50 import preprocess_input
```

Figure 2. Importing necessary libraries.

3.2 | Preparing Data before Model Training

The importance of data organization cannot be emphasized in the fields of machine learning and data science, where data is the foundation of models and algorithms. One crucial aspect of data organization involves looping through the folders of a dataset, extracting relevant information, and structuring it appropriately. This seemingly mundane task holds significant importance and impacts various aspects of data-driven endeavors.

```python
InputPath = []
label = []
sourceFileDir = os.path.dirname(os.path.abspath("__file__"))
print(sourceFileDir)
i = 0
Animals = {}
for Class in os.listdir(os.path.join(sourceFileDir,"Animals")):
    print(Class)
    for path in os.listdir(os.path.join(sourceFileDir,"Animals",Class)):
        label.append(i)
        InputPath.append(os.path.join("Animals", Class, path))
    Animals.update({i:Class})
    i+=1
print(label)
```

[28] Python

```
/home/abdulrahman/Projects/VGG16
elephant
cow
horse
sheep
cat
```

```python
df = pd.DataFrame()
df['images'] = InputPath
df['label'] = label
df = df.sample(frac=1).reset_index(drop=True)
df.head()
```

[3]

	images	label
0	Animals/sheep/OIP-_3bv1qt1iLMDK_zNqRHn7AHaE8.jpeg	3
1	Animals/cow/OIP-dU_0Z7ZmMFTbwM6vHqmm7AHaFj.jpeg	1
2	Animals/sheep/eb32b30d2ef0093ed1584d05fb1d4e9f...	3
3	Animals/cow/OIP-SSu1VqLjOa0O_rlVubCgIQHaE8.jpeg	1
4	Animals/dog/OIP-rQC_I5phMQ0YTDnXwuYP6AHaFj.jpeg	5

Figure 3. Preparing data before model training.

3.3 | Loading, Training, and Validating Data

Loading, training, and verifying data is a common operation in the field of deep learning, but it's also a crucial step that can have a big impact on the dependability and performance of deep learning models.

Splitting the data 70% for training and 30% for testing

```python
train, test = train_test_split(df, test_size=0.3, random_state=1)
```

Python

Figure 4. Training and validating data.

3.4 | Data Augmentation

- In the realm of deep learning, where large amounts of labeled data are often required to train complex neural networks, data augmentation emerges as a powerful technique to address data scarcity, enhance model generalization, and improve overall performance.

- Using ImageDataGenerator to read the data from the data frame to receive new variations of the images at each epoch by doing various operations such as rescaling, rotating, shearing, zooming, horizontal and vertical flipping.

```
TrainGenerator = ImageDataGenerator(dtype = 'float32',
        preprocessing_function=preprocess_input,
        rescale = 1./255,
        rotation_range = 40,
        shear_range = 0.2,zoom_range = 0.2,
        horizontal_flip = True,vertical_flip=True,
        fill_mode = 'nearest')
ValGenerator = ImageDataGenerator(dtype = 'float32',
            preprocessing_function=preprocess_input,rescale = 1./255)
TrainIterator = TrainGenerator.flow_from_dataframe(df,
        x_col='images', y_col='label',
        target_size=(256,256),
        batch_size=8,
        class_mode='categorical')
ValIterator = ValGenerator.flow_from_dataframe(test,
        x_col='images', y_col='label',
        target_size=(256,256),
        batch_size=8,
        class_mode='categorical')

Found 16615 validated image filenames belonging to 9 classes.
Found 4985 validated image filenames belonging to 9 classes.
```

Figure 5. Data augmentation.

3.5 | Building our CNN using Transfer Learning with ResNet50 Architecture

In the dynamic field of deep learning and convolutional neural networks (CNNs), the model's performance, efficiency, and adaptability can be greatly impacted by the choice of architecture. ResNet50 is one such design that has attracted a lot of attention and adoption.

A well-known convolutional neural network architecture with 50 layers is called ResNet50. It is an adaptation of Microsoft Research's ResNet (Residual Network) design. An outline of the ResNet50 architecture is provided below:

- Input Layer
    - Accepts input images of fixed size.
- Convolutional Layers
    - The initial convolutional layer performs convolution on the input image to extract low-level features.
    - Adjacent to a ReLU activation function and a batch normalization layer.
- Residual Blocks
    - ResNet50 is made up of multiple residual blocks. There are several convolutional layers and skip connections in every residual block. (shortcuts).
    - The skip connection allows the gradient to flow directly through the block without passing through the layers, addressing the vanishing gradient problem.
    - These residual blocks enable the network to learn more complex features while mitigating the degradation problem (where adding more layers leads to worse performance).
- Identity Block

- The identity block is a basic building block of ResNet. It contains three convolutional layers:
  - Convolutional layer with 1x1 filters to reduce the dimensionality.
  - Convolutional layer with 3x3 filters to extract features.
  - Convolutional layer with 1x1 filters to increase dimensionality back to the original size.
- A skip connection is used to append the input to the third convolutional layer's output.

- Convolutional Blocks
  - ResNet50 has several convolutional blocks containing multiple identity blocks followed by convolutional layers.
  - These blocks increase the network's depth and allow it to learn more abstract features.
- Global Average Pooling Layer
  - After the convolutional layers, ResNet50 typically includes a global average pooling layer.
  - This layer provides input to the fully linked layers by reducing the spatial dimensions of the feature maps to a vector of average values.
- Fully Connected Layers
  - Finally, ResNet50 includes one or more fully connected layers followed by a softmax activation function.
  - These layers perform classification based on the features learned by the convolutional layers.
- Output Layer
  - The ultimate categorization probabilities are generated by the output layer for the input image, typically representing the likelihood of each class.

ResNet50's architecture allows it to obtain cutting-edge results on a variety of picture classification tasks, such as the identification of sceneries, objects, and animals. Its ability to strike a balance between computational efficiency and model depth qualifies it for practical use.

```
model = Sequential()

model.add(ResNet50(
    include_top = False,
    pooling='avg',
    weights='imagenet',
))
model.add(Dense(512,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(9,activation='softmax'))
model.layers
model.layers[0].layers
model.layers[0].trainable = True
model.summary()
```

```
Layer (type)                 Output Shape              Param #
================================================================
resnet50 (Functional)        (None, 2048)              23587712

dense (Dense)                (None, 512)               1049088

dropout (Dropout)            (None, 512)               0

dense_1 (Dense)              (None, 9)                 4617

================================================================
```

Figure 6. Building our CNN model using the VGG16 architecture.

3.6 | Training and Validation Accuracy and Loss

In the realm of deep learning, where the ultimate goal is to build models that can generalize well to unseen data and make accurate predictions, monitoring training and validation accuracy and loss hold paramount importance. These metrics serve as vital indicators of a model's performance, generalization ability, and training progress.

```
history = model.fit(TrainIterator, epochs=40, validation_data=ValIterator)
```

```
Epoch 10/40
2077/2077 [==============================] - 351s 169ms/step - loss: 0.5206 - accuracy: 0.8289
Epoch 11/40
2077/2077 [==============================] - 337s 162ms/step - loss: 0.4992 - accuracy: 0.8377
Epoch 12/40
2077/2077 [==============================] - 334s 161ms/step - loss: 0.4687 - accuracy: 0.8463
Epoch 13/40
2077/2077 [==============================] - 338s 163ms/step - loss: 0.4506 - accuracy: 0.8556
...
Epoch 39/40
2077/2077 [==============================] - 328s 158ms/step - loss: 0.1984 - accuracy: 0.9349
Epoch 40/40
2077/2077 [==============================] - 328s 158ms/step - loss: 0.1957 - accuracy: 0.9360
```

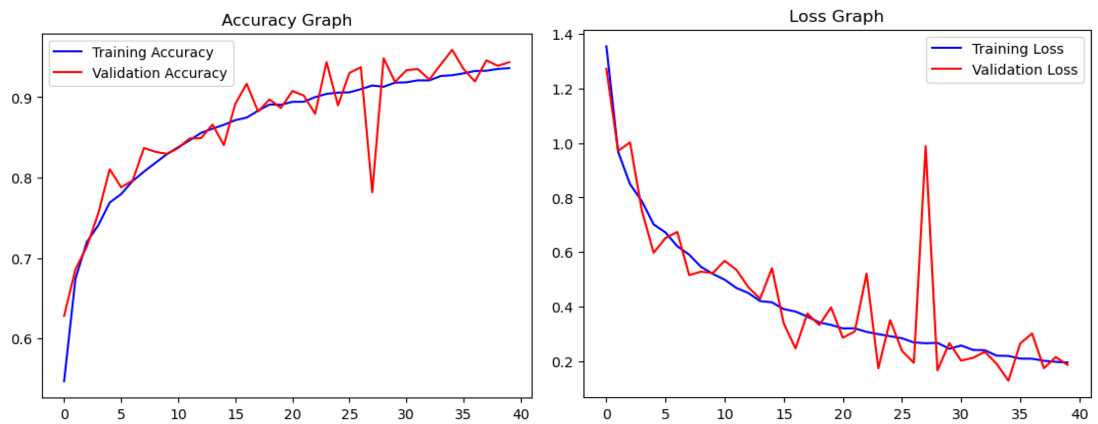

Figure 7. Training and validation accuracy and loss.

3.7 | Testing Images on Mode

Figure 8. Testing image.

4 | Result and Observation

The model under consideration was developed using the Python programming language and underwent testing within Visual Studio. It was evaluated on a dataset comprising 16615 images depicting 9 distinct animal species across various animal classes and kingdoms. Demonstrating notable performance, the model achieved an accuracy rate of 93.6% for the 9 animal classes. The neural network exhibited proficiency in accurately identifying and categorizing animal images into their respective classes. Notably, certain test images were classified with perfect accuracy, achieving a 100% success rate. Furthermore, the model successfully detected and classified 4985 test images from the 9 animal classes, with training conducted on 16615 images, and achieved an accuracy rate of 94.32%, all of which belonged to the same 9 animal classes.

Table 1. Number of images for the animal classes.

Animal	Number of Images
Butterfly	2112
Cat	1668
Chicken	1893
Cow	1866
Dog	2042
Elephant	1446
Horse	1906
Sheep	1820
Squirrel	1862
Total	16615

Table 2. Accuracy rate for training and validation.

Model	VGG16(old model) In[1]	ResNet50(new model)
Training Accuracy	92.3%	93.6%
Validation Accuracy	87.22%	94.32%

5 | Conclusion

The usage of transfer learning with ResNet50 architecture optimizes a powerful approach to image classification. By leveraging the depth and resilience of ResNet50's architecture alongside the knowledge encoded in pre-trained weights, this methodology enables efficient adaptation to large datasets. Not only does it reduce the computational burden and accelerate model convergence, but it also showcases remarkable generalization capabilities across diverse domains. Empirical validation underscores the efficacy of this approach, demonstrating impressive accuracy rates even with large and varied datasets. As we navigate further into this paradigm, the impact of utilizing pre-trained models for diverse applications is set to revolutionize the landscape of deep learning and artificial intelligence.

Acknowledgments

The author is grateful to the editorial and reviewers, as well as the correspondent author, who offered assistance in the form of advice, assessment, and checking during the study period.

Author Contribution

All authors contributed equally to this work.

Funding

This research has no funding source.

Data Availability

The datasets generated during and/or analyzed during the current study are not publicly available due to the privacy-preserving nature of the data but are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there is no conflict of interest in the research.

Ethical Approval

This article does not contain any studies with human participants or animals performed by any of the authors.

References

[1] Khan, S., Singh, S., Almas, S. & Razzaque, A. (2022). Animal Classification Using CNN with VGG-16 Architecture. IJARSCT.

[2] Kaggle. com,Animals10dataset[Online], Availableat : https://www.kaggle.com/alessiocorrado99/animals10.

[3] Rathi, D., Jain, S., & Indu, S. (2017). Underwater Fish Species Classification using Convolutional Neural Network and Deep Learning. International Conference of Advances in Pattern Recognition.

[4] Abas, M. A. H., Ismail, N., Yassin, A. I. M., & Taib, M. N. (2018). VGG16 for plant image classification with transfer learning and data augmentation. International Journal of Engineering &Technology.