Optimization of Neural Network for Identifying Scene Images

[1]Sambhrama KM, [1]Shivaprasad M, [1] Sreeram Bhat, [1] Thejus S K [1] Mrs. Sumathi K
[1] *Department of Electronics and Communication Engineering*
JNN College of Engineering,
Shivamogga, Karnataka, India
sumathik@jnnce.ac.in

Abstract— The fast progress of deep learning has greatly enhanced the accuracy of image classification tasks, especially in scene image recognition. Nevertheless, the challenge of high accuracy coupled with computational efficiency still exists. The optimization of neural network structures is targeted with the aim of boosting the recognition of scene images. Different optimizing methods, such as hyperparameter adjustment, architectural alterations, and regularization schemes, are investigated with the objective of enhancing accuracy as well as speed. Pre-trained convolutional neural networks (CNNs) like VGG, ResNet, and EfficientNet are tested and fine-tuned through transfer learning on scene specific datasets. Model pruning and quantization are also applied to decrease model complexity without impacting performance. Experimental results show that optimized models deliver better accuracy with decreased training time and lower resource utilization. This work demonstrates the efficacy of focused neural network optimization methods to create robust and effective systems for real-world scene image identification tasks.

Keywords— Neural Networks, PCA, Scene Classification, Indoor/Outdoor Detection, Sparse Representation Classifier.

I. Introduction

Over the last few years, the fast development of computer vision and deep learning has enormously enhanced the capacity of machines to analyze and categorize images. Scene image identification is one of the most difficult and influential topics in this area, where the objective is to precisely identify and classify the entire setting or environment represented in an image like a beach, forest, city scene, or indoor room. This activity is important in many applications of real world scenarios such as autonomous driving, surveillance, robotics, and content-based image retrieval.

In order to accurately carry out scene recognition, neural networks, especially convolutional neural networks (CNNs), have proven to be formidable tools since they can learn hierarchies of visual representations. The challenge of creating a highly accurate yet computationally friendly neural network is still a big challenge. Designing against such factors as overfitting, large model sizes, and slow inference times can be a setback, particularly when using models on resource-limited devices.

This research is concentrated on optimizing neural networks for scene image recognition with the goal of improving their accuracy, speed, and generalizability. Different optimization methods, including architectural changes, hyper parameter optimization, data augmentation, regularization, and transfer learning, are investigated to optimize model efficiency and resiliency. Through careful testing and iteration of neural network models, this research adds to more efficient and scalable solutions for automatic scene understanding.

1. Problem statement:

Conventional scene-classification techniques struggle with fluctuating conditions such as poor lighting, random noise, or changes in image scale. High-capacity neural networks achieve superior accuracy but demand substantial computation resources.Thus, the problem addressed in this study is to optimize the neural-network model for scene identification such that accuracy is preserved while reducing computational load, enabling deployment on real-world low-power devices.

II. LITERATURE SURVEY

In the study, A. Orazaev, P. Lyakhov, V. Brizitsky, and D. Korobeynikov present a neural network based system designed to detect and remove random valued impulse noise before classifying images. Their method integrates a dedicated noisy-pixel detector with an adaptive median-filtering mechanism, allowing the system to restore heavily corrupted images more effectively than traditional filtering techniques. By using pixel level brightness and geometric distance measures, the authors propose a more precise noise estimation strategy, which significantly improves the performance of a VGG16 classifier, showing accuracy gains of up to 14.95% over existing approaches. Their findings highlight how preprocessing through improved noise removal can substantially enhance recognition reliability in challenging environments.

A. Convolutional Neural Networks in Image Recognition CNNs have been widely adopted due to their ability to extract hierarchical representations through convolution, pooling, and nonlinear activation layers. Studies such as Ge et al. demonstrated improved classification accuracy by integrating multi layer CNNs with SoftMax classification for hyper spectral images. Similarly, Bai et al. Introduced a dual discrimination auto encoder to enhance zero-shot recognition performance through integrated feature encoding and classification layers.

B. Heterogeneous Multi Column CNN Architectures Single path CNNs often struggle to capture the full range of multi-scale visual information, prompting researchers to adopt multi column designs. Li et al. proposed a heterogeneous multi-column CNN framework in which each column varies in kernel configuration and depth, promoting complementary feature extraction. Sliding Window Fusion (SWF) was introduced to refine prediction reliability by selecting the optimal subset of sorted prediction probabilities for aggregation, outperforming traditional fusion rules.

C. Noise-Resilient Recognition and Image Restoration Recognizing objects in noisy images poses challenges for deep learning models trained on clean data. Recent work integrates noise-detection modules with adaptive filters before classification, ensuring restoration of corrupted pixels prior to neural processing. This two stage strategy noise removal followed by CNN based recognition achieves superior classification accuracy on datasets degraded by random valued impulse noise.
D. Scene Recognition Through Potential Object Region Analysis: Scene understanding requires capturing global contextual information beyond object level cues. The potential object-region recognition model identifies semantic regions by computing object distribution density and applying sliding-window analysis. Multi-scale salient regions are extracted and processed through CNN based transfer learning to produce region-level embeddings. These feature vectors are fused and classified using a Multi Layer Perceptron(MLP), achieving improvements in accuracy and recognition speed.

III. METHODOLOGY

3.1 Dataset & Preprocessing

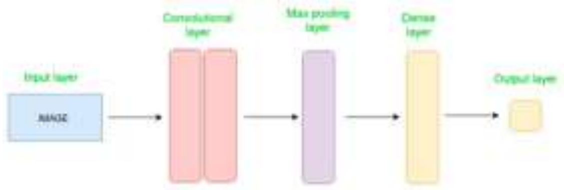


Figure 2.1: CNN model

The first step is to collect a large dataset of satellite images, some labeled as Mountain, Beach, Desert, Pond, Forest, Farmland for a project like Optimization Of Neural Network For Identifying Scene Images involves several steps, including acquiring the images, labelling them, and organizing them into appropriate directories.

Dataset: Total 2080 scene image dataset Categories, Select 6 classes as Mountain, Beach, Desert, Pond, Forest, Farmland.
Preprocessing consist of resizing standardise images to a fixed size(128x128).
Augmentation: Apply transformations (rotation, flipping, zooming) to reduce overfitting.

3.2 Model Architecture:

Design a CNN architecture suitable for image classification tasks. The architecture of the CNN model consists of several layers, including convolution layers, rectified linear unit (ReLU) layers, response normalization layers, and pooling layers.The input images are down-sampled to reduce the computational complexity of the deep learning architecture.

3.3 Training the Mode:

The CNN model is trained using the preprocessed images and their corresponding labels. During training, the model learns to extract relevant features from the images and map them to their corresponding labels. The training process also involves optimizing the model's parameters to minimize the loss function.

3.4 Model Evaluation:

After training, the model's performance is evaluated using a separate validation dataset to ensure it generalizes well to unseen data. Additionally, it is tested on another set of unseen data to measure its performance metrics such as accuracy, precision.The model can be further fine-tuned by adjusting the hyper parameters or adding regularization techniques to improve its performance.

3.5 Prediction:

After careful training and optimization, the neural network shows impressive results in recognizing and classifying different scene images. By using a diverse collection of images ranging from city scapes and countryside to indoor and outdoor settings the model learns to accurately tell scenes apart. Thanks to smart preprocessing steps like image cleanup, contrast improvement, and data enhancement, the network becomes better at spotting important details and telling similar-looking scenes apart. With tweaks to its internal settings and structure, along with the help of techniques like transfer learning, the model becomes faster, more accurate, and more reliable. Overall, it performs well not just in testing but also holds strong potential for real world uses like self driving vehicles, security systems, and apps that understand locations based on images.

IV. DESIGN AND IMPLEMENTATION

4.1 Introduction

The design and implementation of a detection and classification system for Identifying Scene Images detection using Convolutional Neural Networks (CNN) involves several steps.

A CNN is a type of deep neural network that is designed to process and analyze the data that has a grid like topology, such as an image . As shown in the Figure 3.1, the network consists of a series of convolutional layers that apply filters to the input data followed by pooling layers that down sample the output which completes feature extraction. The final layers of the network typically include one or more fully connected layers that perform classification.
Input layer: This layer takes in the raw image data as input and prepares it for processing by the rest of the network, that is the input layer would take in an Identifying Scene Images. The input image can be represented as a 3D tensor shape.
Convolutional 2D layer: This layer applies a set of 32/64/128 filters of size 3*3 to the dataset, which helps to extract features from the image. The resulting feature

maps high-light important patterns and edges in the image that can be used for classification.The output of a convolutional layer can be computed.

2D layer (Max Pooling): This layer reduces the dimensionality of the feature maps with 2*2 or 3*3 by down sampling using maximum.

Fully connected layer: This layer takes the flattened feature maps from the previous layers and applies a set of weights and biases to produce an output which has been passed through one or more fully connected layers that is hidden layers such as dense layer for classification.

Output layer: This layer produces the final output of the network, which is a probability distribution over the different disease classes. The SoftMax function is applied to the output to convert the probabilities into a probability distribution that sums to one. The predicted class is the class with the highest probability in the output of the model. This is the class that the model predicts the input image that it belongs to. After designing the model, the dataset is divided into training, validation, and testing sets.The training set is used to train the model, the validation set is used to tune the model's hyper-parameters, and the testing set is used to evaluate the model's performance. The model is then trained using a suitable loss function, such as categorical cross-entropy, and an optimization function, such as Adam. The model is trained for a certain number of epochs, and the model's performance is monitored during training to detect overfitting or under fitting. After training the model, the model's performance is evaluated using the testing set.

4.2 System Design/Block Diagram

The block diagram outlines the structured workflow of an image detection system using deep learning. It begins with an image dataset, which undergoes data annotation to label objects or features within the images. This is followed by data augmentation, where the dataset is expanded through transformations such as rotation, scaling, and flipping to improve model robustness and prevent overfitting. The resulting annotated and augmented data is used to train the network, allowing the model to learn patterns and features necessary for detection. Once trained, the network is tested based on the trained data to evaluate its accuracy and effectiveness.
It then makes predictions on new test data to simulate real-world application scenarios and check how well it generalizes to unseen images.

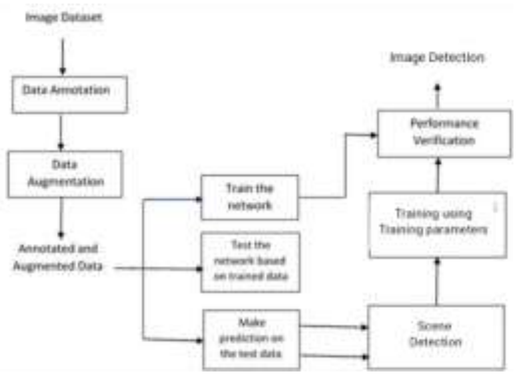


Image Dataset
Data Annotation
Data Augmentation
Annotated and Augmented Data
Train the network
Test the network based on trained data
Make prediction on the test data
Image Detection
Performance Verification
Training using Training parameters
Scene Detection

Figure 2.2: Block diagram

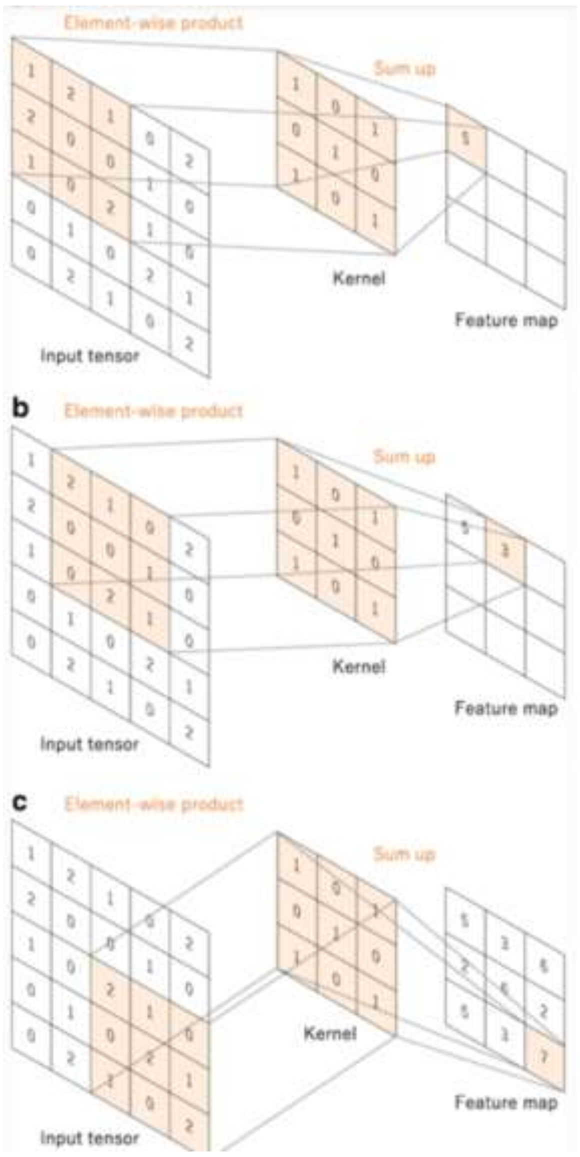


Figure 2.2: Diagram A to C is an example of convolution operation with a kernel size of 3x3.

The predictions are fed into the scene detection module, where the model identifies and classifies various elements within an image. Based on this, further training using optimized parameters is conducted to refine the model's performance. After optimization, performance verification is carried out to assess the reliability and accuracy of the system. If the model meets the performance criteria, it is considered ready for image detection tasks in practical applications. This two-stage process from data preparation to system verification. ensures a reliable and accurate deep learning model for image detection.

4.3 Kernel function over Feature extraction

A kernel is applied across the input tensor, and an element wise product between each element of the kernel and the input tensor is calculated at each location and summed to obtain the output value in the corresponding position of

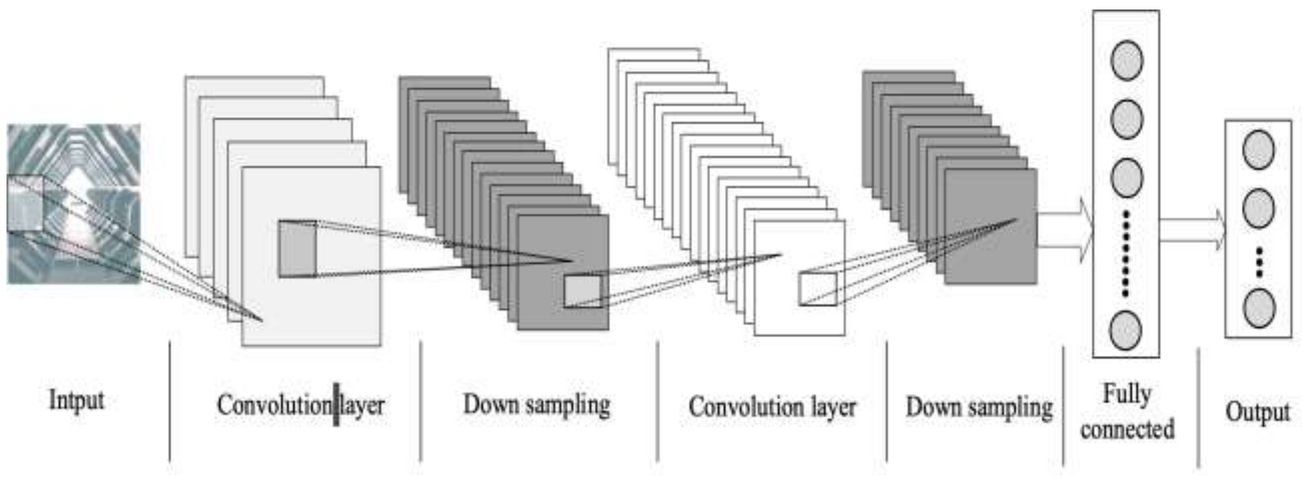


Figure 4.3:

the output tensor, called a feature map.the process of training a CNN model with regard to the convolution layer is to identify the kernels that work best for a given task based on a given training dataset. Kernels are the only parameters automatically learned during the training process in the convolution layer; on the other hand, the size of the kernels, number of kernels, padding and stride are hyper parameters that need to be set before the training process starts. A set of 32/64/128 filters of size 3*3 to the dataset, which helps to extract features from the image. The resulting feature maps highlight important patterns and edges in the image that can be used for classification.

This convolution operation allows the network to extract important features such as edges, textures, and patterns. By stacking multiple layers of convolutions, CNNs learn increasingly complex features, making them highly effective for image classification, detection, and segmentation tasks. The kernel essentially acts as a feature detector during his process.

3.2 Software Components:

1. cv2 (OpenCV): OpenCV(Open Source Computer Vision Library) is an open-source computer vision and machine learning software library widely used for image processing,computer vision tasks and real time video analysis. Developed originally by Intel and now maintained by the OpenCV community, OpenCV2 (often referred to as OpenCV) offers a comprehensive suite of functions and algorithms that enable developers to perform a wide range of tasks, including image capture, manipulation, feature detection, object recognition, and motion tracking.OpenCV provides efficient functions for reading CT scan images stored in various formats. It can also be used for essential preprocessing tasks like resizing, normalization, and potentially intensity standardization of the images for better model performance.
2. numpy (np):NumPy excels at handling multi-dimensional arrays, which is crucial for representing CT scan images. You can use NumPy for efficient image transformations, calculations, and data restructuring for model input.NumPy provides various mathematical functions that might be useful for calculations related to image processing or model training.NumPy plays a crucial role in organizing and processing the data used for training the machine learning model and managing the extracted features. Specifically, NumPy enables developers to store and manipulate the extracted features from captured images.
3. os:This library allows you to interact with the operating system for tasks like accessing and managing CT scan image files. You can use it to navigate directories containing the data and potentially define file paths for loading and saving images.os offers functionalities for checking file existence and handling potential errors during file access. This can be useful for ensuring the code can gracefully handle situations where data files are missing or inaccessible.While libraries like cv2 are better suited for image loading itself, os can indirectly aid in the process. By providing file paths and potentially listing files within a directory, os can help prepare the groundwork for loading images using OpenCV or other libraries.
4. random (shuffle):Shuffling the data before training helps prevent the model from learning biases based on the order of data presentation. This is particularly important for ensuring the model generalizes well to unseen data
5. tqdm: Training a deep learning model on medical images can take time. Tqdm provides a progress bar that displays the training progress, making it easier to track the ongoing process.
6. tensorflow.python.framework.ops(ops):TensorFlow is an open-source machine learning framework developed by Google, renowned for its flexibility, scalability, and ease of use in building and deploying deep learning models.This import statement suggests using TensorFlow as the deep learning framework for building your lung cancer detection model. TensorFlow provides the computational backend for defining and executing the model’s operations.This operation is essential for CNNs used in image recognition tasks like lung cancer detection. It performs the core convolution process described earlier, where filters (kernels) slide across the

input CT scan image, extracting features.tf.ops provides functions like tf.nn.conv2d to define convolutional layers in your model.

V. RESULTS AND DISCUSSION

5.1 Dataset & Preprocessing steps:

The dataset used in this project was sourced from Kaggle and contains a total of 1,600 images categorized into six distinct classes: Pond, Beach, Industry, Desert, Mountain, and Forest. These categories collectively represent a wide variety of natural and man made environments, enabling the model to learn diverse visual patterns. Each class contributes an equal number of images, ensuring balanced training and unbiased classification performance across all categories.
Preprocessing was performed to prepare the dataset for model training, beginning with image initialization at a resolution of 128×128, batch size set to 32, and training configured for 25 epochs across 65 batches. Normalization was applied to scale pixel values and ensure uniform input distribution for the neural network. To improve model robustness, several augmentation techniques including horizontal flips, vertical flips, and rotation were employed. These steps help reduce overfitting and enhance the model's ability to generalize across unseen images.

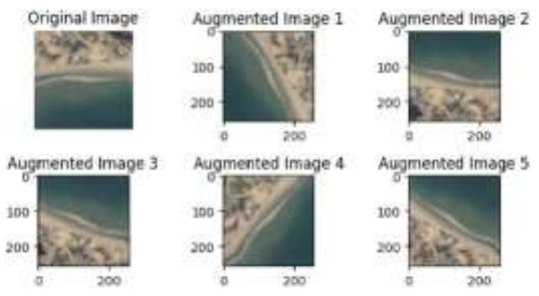


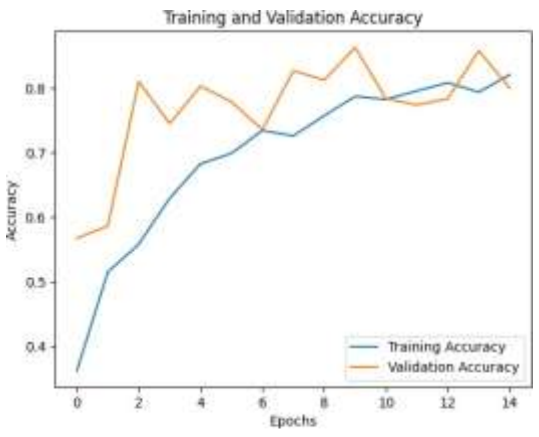
Figure 5.1: Augmented Results

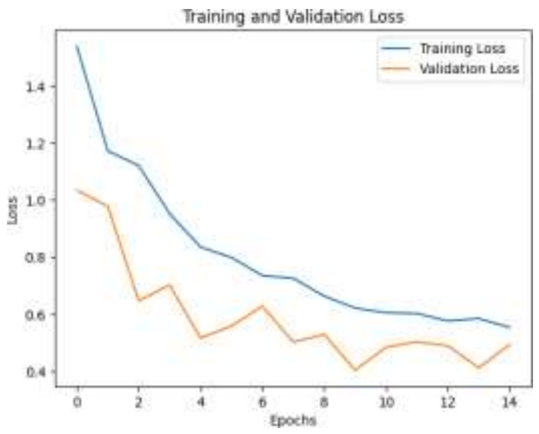
5.2 Training Data with augmented images:

During the training phase, augmentation parameters were carefully configured to enrich the dataset without distorting essential visual information. Pixel values were rescaled to the range [0, 1] for stable gradient-based optimization. A validation split of 20% was used to monitor performance during training. Additional transformations including random rotations up to 20 degrees, width and height shifts, zooming, and horizontal flips ensured that the model encountered a broad range of variations, significantly improving its resilience against orientation and positional changes.

5.1 Model Compilation and Optimization:

The model was compiled using the Adam optimizer, chosen for its adaptive learning rate capability that accelerates convergence and stabilizes training. A suitable loss function was employed to quantify prediction error by comparing model outputs with actual class labels. For evaluation, 1,680 images were allocated for training and 420 for validation, enabling a balanced assessment of learning performance and generalization capability. A summary table captures the key accuracy and loss metrics obtained after training.
Model evaluation revealed strong classification performance, achieving 89.16% training accuracy, 88% testing accuracy, and a loss rate of 29.16%.





5.1 Class-Wise Accuracy Report:

Class-wise accuracy results show performance variations across the six categories, with Forest achieving the highest accuracy at 92.22%, followed by Desert at 89.99%, Beach at 88.3%, and Pond at 85.88%. Meanwhile, Industry and Mountain recorded comparatively lower accuracies of 77.3% and 75.69%, respectively. These differences suggest that complex textures or visually similar backgrounds may make certain categories more challenging for the model to distinguish.

Sl. No.	Classes	Accuracy percentage
1.	Pond	85.88%
2.	Beach	88.3%
3.	Industry	77.3%
4.	Desert	89.99%
5.	Mountain	75.69%
6.	Forest	92.2223%

5.1 Sample Prediction Results:

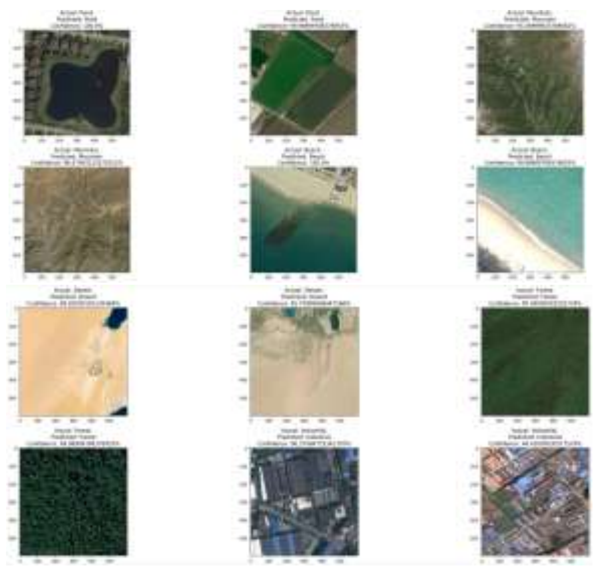


Figure : Obtained Results

IV. CONCLUSION

Optimization of the neural networks to identify scene images is crucial to the development of visual recognition automation systems. Application of a robust CNN design and appropriate preprocessing of data, augmentation techniques, and hyperparameters such as the learning rate, batch size, and number of epochs can help improve the performance of the model significantly. Methods like data normalization, image data augmentation, and dropout are utilized to prevent overfitting and improve generalizability across multiple environments. The application of the Adam optimizer with the categorical cross-entropy loss function, coupled with a well-selected activation strategy (e.g., ReLU and Softmax), enables effective training and correct multi-class classification. The accuracy metrics illustrate that with correct optimization, the model can effectively classify multiple scene classes (e.g., beach, mountain, forest) even under complicated visual scenarios. This is done through emphasizing the capability of deep learning to comprehend scenes at the same time as providing a firm groundwork for its application to autonomous systems, intelligent surveillance, and image retrieval based on content.

REFERENCES:

[01] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. (2015) "Going deeper with convolutions." In Proceedings of the IEEE conference on computer vision
and pattern recognition

[02] Gao, Jingyu, Jinfu Yang, Guanghui Wang, and Mingai Li. (2016) "A novel feature extraction method for scene recognition based on centred convolutional restricted Boltzmann machines." Neuro computing 214: 708-717.

[03] Masood, Sarfaraz, Tarun Luthra, Himanshu Sundriyal, and Mumtaz Ahmed. (2017)"Identification of diabetic retinopathy in eye images using transfer learning." In 2017International Conference on Computing, Communication and Automation (ICCCA), pp.1183-1187. IEEE.

[04] Kibria, Sakib B., and Mohammad S. Hasan. (2017) " An analysis of Feature extraction and Classification Algorithms for Dangerous Object Detection." In 2017 2nd International Conference on Electrical Electronic Engineering (ICEEE), pp. 1-4. IEEE.

[05] D. Jia, D. Wei, L.J. Li, L. Kai, and F.F. Li, ImageNet: A Large-scale Hierarchical
Image Database, IEEE Conference on Computer Vision and Pattern Recognition, 2009.

[06] Y. Jiang, Research on scene image content representation and classification, National
University of Defence Technology Doctoral Thesis, 2010

[07] Sajna Tasneem, S.; Shabeer, S. Optimized Image Restoration Based on Residual Cascade Convolution Neural Networks. In Proceedings of the International Conference on Intelligent Computing and Control Systems, Secunderabad, India, 15–17 May 2019; pp. 550–553.

[08] Yun, K.; Huyen, A.; Lu, T. Deep Neural Networks for Pattern Recognition. Adv. Pattern Recognit. Res. 2018, 49–79. [CrossRef

[09] Zhang, J.; Shao, K.; Luo, X. Small Sample Image Recognition Using Improved Convolutional Neural Network. J. Vis. Commun. Image Represent. 2018, 55, 640–647.

[10] Khaw, H.Y.; Soon, F.C.; Chuah, J.H.; Chow, C.O. Image Noise Types Recognition Using Convolutional Neural Network with Principal Components Analysis. IET Image Process. 2017, 11, 1238–1245.

[11] Yim, J.; Sohn, K.-A. Enhancing the Performance of Convolutional Neural Networks on Quality Degraded Datasets. In Proceedings of the International Conference on Digital Image Computing, Sydney, Australia, 29 November–1 December 2017.

[12] Momeny, M.; Latif, A.M.; Agha Sarram, M.; Sheikhpour, R.; Zhang, Y.D. A Noise Robust Convolutional Neural Network for Image Classification. Results Eng. 2021, 10, 100225.

[13] Aggarwal, C.C. Neural Networks and Deep Learning. In Neural Networks and Deep Learning; Springer: New York, NY, USA, 2018.

[14] Schaefferkoetter, J.; Yan, J.; Ortega, C.; Sertic, A.; Lechtman, E.; Eshet, Y.; Metser, U.; Veit-Haibach, P. Convolutional Neural
Networks for Improving Image Quality with Noisy PET Data. EJNMMI Res. 2020, 10, 105.