Exploring New Convolutional Neural Network Architectures for Satellite Image Categorization: A Comparative Study

Aryan Jain*, Debjani Ghosh†, Prashant Kumar‡,Ashish Soni §, Aryamaan Chaudhary¶
School of Computer Science Engineering and Technology
Bennett University, Greater Noida, India
*E21CSEU0220@bennett.edu.in, †debjani.ghosh@bennett.edu.in, ‡prashant.mnnit10@gmail.com,
§E21CSEU0228@bennett.edu.in, ¶E21CSEU0214@bennett.edu.in

***Abstract*—With deep learning, satellite images can accurately identify land cover elements such as annual crops, forests, herbaceous vegetation, pastures, permanent crops, and rivers. Deep learning has become a valuable tool in this regard. In this field, convolutional neural networks (CNNs) have shown great promise, with pre-trained models such as DenseNet121, EfficientNetB0, VGG16, and InceptionV3 showing encouraging outcomes. To outperform earlier CNN models, this research explores the possibilities of new CNN designs for satellite image categorization. We suggest a comparative study where these more recent architectures are refined using a dataset of satellite images. The results are shown together with a description of how well the above-mentioned CNN models work for this task. The performance is assessed based on accuracy and a discussion of the model's suitability for this task is included with the result.**

***Index Terms*—Satellite image classification, deep learning, CNN, DenseNet121, EfficientNetB0, VGG16, InceptionV3, accuracy**

I. Introduction

Classifying satellite pictures is important in application fields, such as land usage monitoring, environmental analysis, as well as accurate agro-based integration. Traditional ML methods, including SVMs, were used to classify spy satellites for relatively mediocre achievement. In recent times, convolutional neural network have surfaced as an effective weapon regarding tackling classifier problems, showing efficiency. This detailed investigation uses CNN models for characterizing satellite pictures into other six primary landform categories: annual crop, forest, herbaceous vegetation, pasture, permanent crop, and river. We assess the feasibility of 4 pre-trained CNN types, Densenet121, EfficientnetB0, VGG16, InceptionV3, on a dataset of 16,000 satellite photos. We used accuracy as a performance metric to compare the efficiency of these CNN models.

The said proposal adds toward the practice area like satellite image categorization in the following manner:

- Efficiency: We applied pre-trained models, attempting to reduce practice time and computation time if compared to training models from scratch.
- Simplicity and repeatability: Our research methods utilises well-defined data pre-processing and augmentation methods, making it repeatable for future studies.
- Comparison with existing work: We provide a comprehensive comparison with related articles, emphasizing breakthroughs attained by our project.

The next sections of the paper have been organized in the following approach: Section II includes an in depth summary of the applicable tasks as well as a comparison to previous studies. Section III informs about its methods, such as the data - set, data pre-processing techniques, data augmentation methods, and the four CNN models applied. Section IV elaborates on the results and discussion, including performance analysis on the basis of accuracy and comparison across all trained models. Section V presents the conclusion of the paper.

II. Related Work

Several studies [1]–[3] have explored CNNs for satellite image classification tasks. The authors of [1] have designed a CNN architecture for classifying high-resolution satellite images into 15 land cover classes, achieving an overall accuracy of 87%. [2] have researched transfer learning with CNNs for land cover classification using a data-set of 30,000 labelled high-resolution satellite images, achieving an overall accuracy of 90.2%. The authors of [3] applied deep CNN architecture for feature extraction of very high-resolution satellite images, achieving an accuracy of 89.% for 13 land cover classes. The result of our research outperforms the accuracy achieved

in [1] for a comparable number of land cover classes, showing the effectiveness of EfficientnetB0 in obtaining relevant features from satellite images. While [3] got similar levels of accuracy for a wider range of land cover classes, our project focuses on model complexity in order to possibly decrease training times, thus highlighting the trade-off between model complexity, accuracy, and efficiency. Our approach achieves comparable accuracy to [2] while using pre-trained models, which may result in reduced training resource needs.

III. Methodology

This section gives an in-depth description of the dataset, the methodologies used to pre-process the data, the techniques used to perform data augmentation and the four CNN models that are used in the project.

A. Dataset

Our research used a dataset [4] which includes 16,000 satellite images taken from an authorized source. The images are categorised into six different land cover classes: annual crop, forest, herbaceous vegetation, pasture, permanent crop, and river. It is essential to keep a fair division of classes for training CNN models to achieve the best results. Table I shows the distribution of classes in the dataset, showing that each class has a relatively similar amount of images for training purposes.

TABLE I
Class Distribution of the Satellite Image Dataset

Class	Number of Images
Annual Crop	3000
Forest	3000
Herbaceous Vegetation	3000
Pasture	3000
Permanent Crop	3000
River	2500

B. Data Pre-processing

It includes the preparation of data for analysis which includes cleaning, conversion, and rearranging it into an appropriate form for further analysis. Before inputting them into the CNN models, the satellite images required to undergo pre-processing. The pre-processing methods include:

- **Resizing** All images are equally adjusted to have a resolution of 150 x 150 pixels. The computational needs of training the CNN models decrease.

  The formula used to calculate the computational complexity of a the convolutional layer can be stated in Equation 1.

$$Complexity = (NumberofInputChannels) \times (KernelSize)^2 \times (NumberofOutputChannels) \times (OutputFeatureMapSize)^2 \quad (1)$$

  When the photos are resized, the "Output Feature Map Size" in this equation decreases, resulting in less computational complexity.
- **Normalisation**
  The pixel values of the images are scaled to a range lies between 0 and 1. Normalisation improves the convergence of the training process by verifying that all feature values have been normalized to a similar scale.
  The normalizing formula can be mathematically written as shown in Equation 2

$$NormalizedValue = \frac{(ActualValue - MinimumValue)}{(MaximumValue - MinimumValue)} \quad (2)$$

C. Data Augmentation

These methods are used to manually increase the dimension of the training dataset to improve the model's ability to generalise. The following data augmentation methods are applied:

1. **Random Rotation:** Performs a random rotation (θ) to the images to improve their ability to handle variations in image orientation.
2. **Width Shift:** This operation randomly adjusts the image horizontally by a particular number of pixels (ΔX) to reflect variations in the position of components. One can do this task using image manipulation libraries.
3. **Height Shift:** Offers a random shift in the image's position along the y-axis by a particular number of pixels (ΔY) to account for possible changes in the position of an object. Like the width shift, this can be achieved by using image manipulation libraries.
4. **Shear:** Uses a shearing transformation to deform the image, hence enhancing the model's capacity to adapt to geometrical variations. Shear transformations include adjusting the spatial coordinates of pixels according to a predetermined set of parameters.
5. **Zoom:** Models different distances by randomly adjusting the scale factor to zoom in or out on the image. Zooming can be achieved by using scaling methods offered by image manipulation libraries.

D. Convolutional Neural Network Models

The following subsection discusses the four pre-trained CNN models applied in the project: DenseNet121, EfficientNetB0, VGG16, and InceptionV3. We will look at the architecture and fundamental principles of these models, with a focus on the mathematical calculations performed in their convolutional layers.

1) DenseNet121: DenseNet121 is an integral part of the Dense Convolutional Networks family, which is a family of CNN architectures. These networks handle the challenge of the vanishing gradient problem, which is a difficulty experienced in deep neural networks. The issue occurs when gradients get extremely small during the back propagation process, resulting in slowing the training process. DenseNets achieves this by creating links between each layer and all succeeding levels in the structure of the network, encouraging the spread of features and the flow of information across the model.

The DenseNet121 architecture includes four dense convolutional blocks, each having a series of convolutional layers with batch normalisation and ReLU (Rectified Linear Unit) activation functions. The blocks are organized in an alternating pattern with transition layers, which help in minimizing the size of the feature map so as to improve the computation. Next, the neural network uses global average pooling, a fully-connected layer, then a softmax layer for classification tasks.

The basic method of CNNs is the convolution, where a kernel gets applied to the input feature map so as to extract characteristics. The mathematical representation of the convolution operation for one channel is shown in Equation 3.

$$Output[x, y] = \sum (Input[p, q] \times Kernel[x - p, y - q]) + Bias \quad (3)$$

where:

- Output[x, y] denotes the component at position (x, y) in the output feature map.
- Input[p, q] represents the component at position (p, q) in the input feature map.
- Kernel[x - p, y - q] represents the component at position (x - p, y - q) in the filter (kernel).
- Bias is a term added to the output.
- The summation iterates over all elements of the kernel that overlap with the current position in the input feature map.

The architecture of DenseNet121 is shown in the Fig.1 [5].

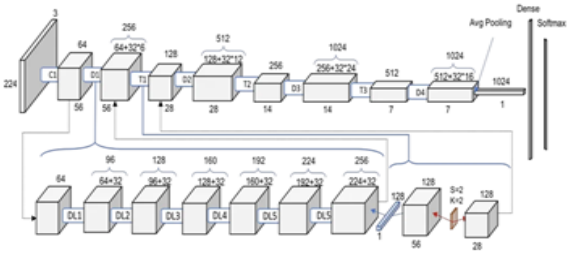

Fig. 1. DenseNet121 architecture

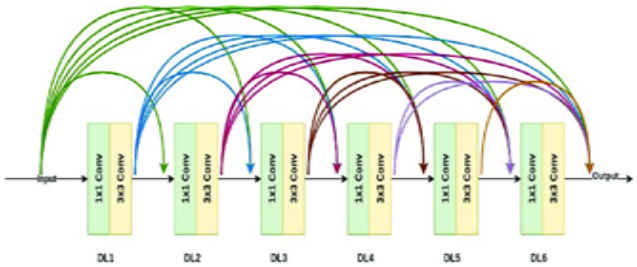

Fig. 2. EfficientNetB0 architecture

2) EfficientNetB0: EfficientNetB0 uses a compound scaling approach to preserve a uniform baseline network structure. This architecture depends on MobileNet inverted bottleneck blocks, that are efficient building blocks that are ideal for mobile and embedded devices. The compound scaling approach uses a scaling coefficient to alter the depth (number of layers), width (number of channels in each layer), and resolution of the input picture in a network. This allows for an organized review of the connection between accuracy and efficiency.

The mathematical complexities of depth, width, and resolution scaling in EfficientNet is not addressed in this research because to its complex nature. However, the basic approach focuses on changing the number of convolutional layers, the number of filters within each layer, and the size of the input picture according to the chosen scaling coefficient.

The architecture of EfficientNetB0 is shown in the Fig.2 [6]. VGG16 achieved outstanding results on the ImageNet classification task. The structure of VGG16 is defined by the layering of many convolutional layers, each using tiny (3x3) filters. The convolutional layers are alternated with pooling layers to reduce the dimensionality. After that, the network uses fully-connected layers with dropout for regularisation so as to reduce overfitting, and concludes with a softmax layer for classification.

The mathematical operations executed in the convolu-

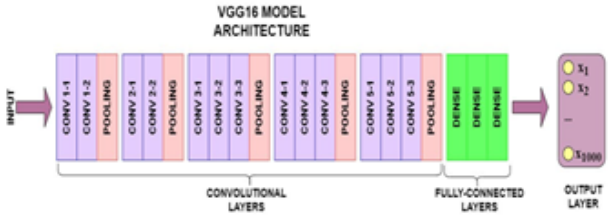

Fig. 3. VGG16 architecture

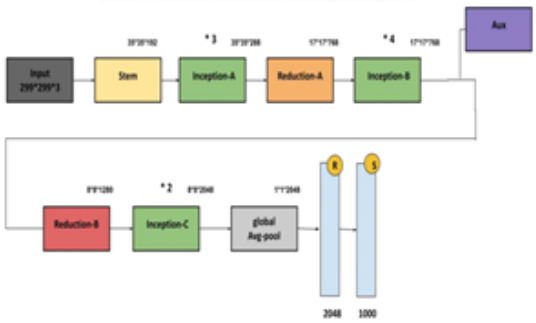

Fig. 4. InceptionV3 architecture

tional layers of VGG16 are similar to those described for DenseNet121, as stated in the equation for the Convolutional Layer. The primary difference between VGG16 and EfficientNetB0 is in their architectural design. VGG16 utilizes a series of stacked convolutional layers with smaller filters, which can result in a larger number of parameters compared to the approach used by EfficientNetB0.

The architecture of VGG16 shown in Fig.3 [7] is defined by its simplicity as well as its efficiency.

The network starts off with an input layer that receives an image of a specific dimension (e.g., 224x224x3 for RGB images).

The basic elements of VGG16 consist of five convolutional blocks. Each block consists of a set of convolutional layers (generally 2 or 3), with similar filter size (3x3) and an equal number of filters inside the block. These blocks gradually extract elements of greater complexity.

- Pooling Layers: Following each convolutional block, a max pooling layer is employed in order to reduce the dimension of the feature maps.
  After the convolutional blocks, the network proceeds to fully-connected layers. The intent of these layers is to convert the feature maps into vectors and carry out classifications using a substantial number of neurons.
- Dropout layers are intentionally placed between fully-connected layers to mitigate overfitting. During the training process, a random subset of neurons is excluded, which compels the network to acquire resilient characteristics that do not excessively depend on any individual neuron.
- The last layer is a softmax layer that produces a probability distribution for each class, showing the chance of a picture belonging to a specific category.

3) InceptionV3: InceptionV3 is a complex convolutional neural network structure that falls under the Inception family. Inception networks incorporate inception modules, which integrate convolutional and pooling layers into a single entity. The utilization of these inception modules allows effective investigation of various filter sizes within the network, which may result in improved feature extraction capabilities. InceptionV3 uses a hierarchical arrangement of inception modules with varying filter widths. The purpose of this architecture is to extract a wider range of spatial and spectral data from the input images.

The mathematical procedures carried out in the convolutional layers of InceptionV3 are similar to those outlined for DenseNet121, as stated in the equation for the Convolutional Layer. The primary difference is in the use of inception modules, which combine various convolutional and pooling processes into a single unit. This approach allows the exploration of different filter sizes within a single module, which may result in enhanced feature extraction compared to using separate convolutional and pooling layers.

The architecture of InceptionV3 is shown in Fig.4 [8].

4) Model Training: This section includes an in-depth examination of the learning procedure and the outcomes obtained, providing valuable information on the efficiency of each CNN model in detecting satellite images. The Adam optimizer is widely used for training deep neural networks because of its ability to adaptively adjust the learning rate. It calculates a learning rate for each parameter separately, modifying it according to previous gradients. This frequently results in quicker convergence in comparison to conventional gradient descent methods that employ a constant learning rate for all parameters.

The Categorical Cross-Entropy Loss is a mathematical

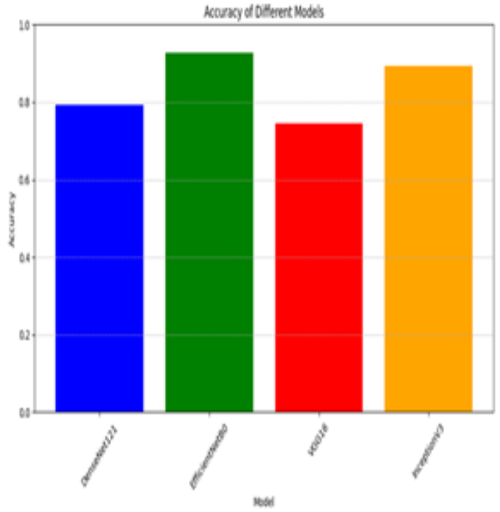

Fig. 5. Comparison of the models based on the accuracy

function used to measure the difference between predicted and actual categorical values. This loss function is particularly suitable for multi-class classification tasks, such as land cover classification. The metric quantifies the disparity between the probability distribution projected by the model for each category and the factual distribution of categories (represented as one-hot encoding) for a picture. Throughout the training process, the model's aim is to minimise the loss function, allowing it to change its parameters so as to generate accurate class probabilities.

The epoch frequency is the total number of times the model went through training utilizing the entire data set. Starting with five epochs is a typical choice to start assessing the performance of the model. An epoch represents a single iteration through the training data. While additional research could reveal an appropriate number of epochs for each model, using five epochs as a starting point is a fair foundation for initial assessment.

IV. Results and Discussion

This section focuses on the results that were obtained from training four CNN models: DenseN24et121, EfficientNetB0, VGG16, and Inception V3. Fig.5 shows an analysis of the four models based on their accuracy.

The test set accuracy of each model gives helpful insights illustrated in Table II

- The Dominance of EfficientNetB0: EfficientNetB0 shows exceptional performance with an impressive accuracy rate of 92.87%, making it the clear frontrunner. The outstanding performance can be ascribed to two main factors:

TABLE II
Comparision of the models

Model	Accuracy
DenseNet121	0.7926
EfficientNetB0	0.9287
VGG16	0.7458
InceptionV3	0.8945

  - EfficientNetB0's compound scaling mechanism allows the creation of a well-balanced architecture that is highly effective. The network's depth, breadth, and resolution can be adjusted based on a single scaling coefficient. This has the potential to improve the allocation of resources, resulting in a model that is both accurate and efficient.
  - Task Suitability: The accurate design choices of EfficientNetB0 make it highly suitable for the task of categorising land cover categories in satellite pictures. The model's ability to extract relevant features from these images is likely a contributing factor to its exceptional success.
- DenseNet121 and InceptionV3 both obtained respectable accuracies, with values of 79.26% and 89.45% respectively. This underlines the overall efficacy of Convolutional Neural Networks for tasks including the classification of satellite images. However, their performance is not as good as EfficientNetB0, suggesting possible constraints.
- Architectural Complexity: The dense connectivity patterns of DenseNet121 might require a larger amount of training data or computer resources in order to fully utilise its capabilities, contrary to the efficient architecture of EfficientNetB0.
- Feature Extraction: Although InceptionV3's inception modules have their benefits, they may not be as efficient as the design choices made in EfficientNetB0 in capturing the exact features that are crucial to classifying land cover in satellite pictures.
- Evaluation of VGG16's Performance: The accuracy of VGG16 was the lowest, measuring at 74.58%. This can be ascribed to its comparatively less complex structure in comparison to the other models. VGG16 uses the method of merging many convolutional layers with small filters, resulting in a large number of parameters and a potential risk of overfitting. Additionally, its design might not have efficiency in extracting the essential features required for this specific purpose.

Overall, the results highlight the importance of investigating recent progress in CNN structures. The out-

standing efficiency of EfficientNetB0 underscores the effectiveness of designing efficient and task-oriented models for image classification tasks, such as land cover classification in satellite imagery.

V. Conclusion

Four pre-trained CNN models, especially DenseNet121, EfficientNetB0, VGG16, and InceptionV3, were trained using a dataset which includes 16,000 satellite images. These images were classified into six distinct land cover categories. The EfficientNetB0 model emerges as the winner, with an excellent accuracy percentage of 92.87There are supplementary options accessible for more examination and improvement.

- Hyperparameter Tuning: Using several hyperparameter optimisation techniques, like as grid search and random search, may improve the model's performance for the given dataset. Epoch Exploration involves conditioning the models for a prolonged number of epochs while closely verifying the validation accuracy. This approach allows the investigation of potential improvements in accuracy without the possibility of excessive fitting. Performing an architectural investigation to analyze several CNN designs, such as U-Net or DeepLabv3+, specifically built for satellite image classification applications, could result in further enhanced results.
- Techniques for improving data through augmentation: By using advanced data augmentation techniques, such as colour jittering or elastic transformations, it is possible to increase the variety of the training data.
- Gathering of information: Enhancing the model's capacity for generalisation can be achieved by increasing the dataset size through the integration of satellite pictures from differed geographical regions. Additionally, offering additional land cover categories may also contribute to this improvement.

This research contributes to the future study of deep learning methods in the classification of satellite photos. By utilising advancements in convolutional neural network (CNN) architectures, data augmentation techniques, and hyperparameter optimisation, it is possible to achieve higher levels of accuracy and generalisability.

References

[1] R. Naushad, T. Kaur, and E. Ghaderpour, "Deep transfer learning for land use and land cover classification: A comparative study," *Sensors*, vol. 21, no. 23, p. 8083, Dec. 2021. [Online]. Available: http://dx.doi.org/10.3390/s21238083

[2] R. Gaetano, D. Ienco, K. Ose, and R. Cresson, "A two-branch cnn architecture for land cover classification of pan and ms imagery," *Remote Sensing*, vol. 10, no. 11, 2018. [Online]. Available: https://www.mdpi.com/2072-4292/10/11/1746

[3] P. Zhang, Y. Ke, Z. Zhang, M. Wang, P. Li, and S. Zhang, "Urban land use and land cover classification using novel deep learning models based on high spatial resolution satellite imagery," *Sensors*, vol. 18, no. 11, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/11/3717

[4] P. Helber, B. Bischke, A. Dengel, and D. Borth, "Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 7, pp. 2217–2226, 2019.

[5] P. Ruizz. (2022) Understanding and visualizing densenets. Accessed: 13/05/2024. [Online]. Available: https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a

[6] H. Amin, A. Darwish, A. E. Hassanien, and M. Soliman, "End-to-end deep learning model for corn leaf disease classification," *IEEE Access*, vol. 10, pp. 31 103–31 115, 2022.

[7] J. McDermott. (2024) Hands-on transfer learning with keras and the vgg16 model. Accessed: 13/05/2024. [Online]. Available: https://www.learndatasci.com/tutorials/hands-on-transfer-learning-keras/

[8] A. Brital. (2024) Inception v3 cnn architecture explained. Accessed: 13/05/2024. [Online]. Available: https://medium.com/@AnasBrital98/inception-v3-cnn-architecture-explained-691cfb7bba08