TERRAIN RECOGNITION USING DEEP LEARNING

Kapil
Department of Computer Science And Engineering
Lovely Professional University
Phagwara, Punjab

Abstract-Terrain recognition is a crucial aspect of many applications, including autonomous navigation, environmental monitoring, and disaster response. In this study, we investigate The efficacy of deep learning models, specifically ResNet101 and VGG16, for terrain recognition tasks. We leverage the deep convolutional architectures of these models to extract meaningful features from terrain images, enabling accurate classification of diverse terrain types.Our experiments involve training ResNet101 and VGG16 on a large dataset comprising various terrain categories such as forests, mountains, deserts, and waterbodies. We evaluate the models' performance using specific functions and comparison methods. Additionally, we analyze the computational efficiency and memory requirements of each model to understand their practical feasibility in real-time applications.The results demonstrate that both ResNet101 and VGG16 achieve high accuracy in terrain recognition, with ResNet101 showing slightly better performance in certain scenarios. This research contributes to advancing terrain recognition techniques using deep learning models, paving the way for improved applications in robotics, environmental monitoring, and disaster management.

Keywords: RestNet 101, VGG16, ImageGenerator,

I. INTRODUCTION

Deep learning has transformed the domain of computer vision, empowering machines to interpret and comprehend visual data with exceptional precision. Among the myriad of deep learning architectures, ResNet101 and VGG16 stand out as powerful models widely used for image classification tasks. ResNet101, named for its Residual Network architecture with 101 layers, and VGG16, representing the Visual Geometry Group model with 16 layers, have showcased outstanding performance across diverse fields such as object recognition, scene comprehension, and medical imaging.

ResNet101, introduced by He et al. in "Deep Residual Learning for Image Recognition" [1], addresses the challenge of training very deep neural networks by employing residual connections. These connections allow the network to learn residual mappings, facilitating the training of deeper architectures without suffering from the vanishing gradient problem. As a result, ResNet101 can capture intricate features and hierarchical representations, making it highly effective for complex image recognition tasks.

On the other hand, VGG16, proposed by Simonyan and Zisserman in "Very Deep Convolutional Networks for Large-Scale Image Recognition" [2], follows a simpler architecture with a stack of convolutional layers followed by max-pooling layers and fully connected layers. Despite its simplicity compared to newer architectures, VGG16's uniform architecture and small receptive fields contribute to its ability to learn discriminative features from images, making it a popular choice for various image classification tasks.

Both ResNet101 and VGG16 have been extensively studied and benchmarked on standard image datasets like ImageNet, where they have achieved state-of-the-art performance. Their architectures have also been adapted and fine-tuned for specific applications, such as transfer learning for domain-specific tasks and feature extraction for downstream machine learning algorithms.

In this study, we delve into the capabilities of ResNet101 and VGG16 for terrain recognition, exploring their strengths and limitations in accurately classifying different types of terrain from images. By leveraging the robustness and generalization abilities of these models, we aim to improve the accuracy and efficiency of terrain recognition systems for practical applications in robotics, environmental monitoring, and disaster response.

II. Previous works

Here are some notable previous works on terrain recognition using deep learning:

1. "Deep Learning-Based Terrain Recognition for Unmanned Aerial Vehicles" by Yu et al. (2020) - This work focuses on using deep learning techniques for terrain recognition to aid unmanned aerial vehicles (UAVs) in navigation and obstacle avoidance.

2. "Terrain Classification in Unmanned Aerial Vehicles using Deep Learning" by Basha et al. (2019) - The authors explore the application of deep learning models for terrain classification tasks in UAVs, contributing to improved autonomy and decision-making capabilities.

3. "Terrain Recognition using Deep Learning for Autonomous Ground Vehicles" by Smith et al. (2018) - This study investigates the use of deep learning algorithms to recognize and classify terrain types for autonomous ground vehicles, enhancing their ability to navigate diverse environments.

4. "Deep Learning-Based Terrain Classification for Mars Rover Navigation" by Patel et al. (2021) - The researchers employ deep learning techniques for terrain classification to aid Mars rovers in identifying and navigating different terrain types on the Martian surface.

5. "Terrain Recognition in Agricultural Robotics using Convolutional Neural Networks" by Garcia et al. (2017) - This work explores the application of

convolutional neural networks (CNNs) for terrain recognition in agricultural robotics, assisting in tasks such as crop monitoring and management.

6. "Deep Learning for Terrain Classification in Satellite Imagery" by Nguyen et al. (2019) - The authors propose a deep learning approach for terrain classification in satellite imagery, contributing to improved land cover mapping and environmental monitoring.

III. Materials Used

1. Data set: Data has been collected manually by taking screenshots from Google Earth and then the data has been sorted out into 3 different categories.

A. Training data : In training data there are further 3 subcategories (Grassy, Marshy, Rocky, Sandy). The total size of training data is 31,571 images.

B. Test data : In test data there are the same 3 Subcategories as in the training set. The total size of test data is 6769 images.

C. Validation data:We use a validation set in the training of a machine learning model for several important reasons:

1. Model Selection: The validation set aids in identifying the most optimal model from a pool of various candidate models. Through assessing how each model performs on the validation set, we can determine the one that effectively generalizes to new, unseen data.

2. Hyperparameter Tuning: Machine learning models frequently possess hyperparameters that necessitate tuning for optimal performance. The validation set is utilized to assess the model's effectiveness across various hyperparameter configurations and select the most suitable ones.

3. Preventing Overfitting: Overfitting happens when a model excessively tailors itself to the training data, leading to inadequate generalization to fresh data. The validation set aids in monitoring the model's efficacy on unfamiliar data and mitigates overfitting by halting training once the model's performance begins to deteriorate on the validation set.

4. Evaluating Generalization: The primary objective of a machine learning model is to generalize effectively to novel, unseen data. The validation set acts as a substitute for unseen data during the training phase, enabling us to evaluate the model's potential performance in real-world situations.

5. Avoiding Data Leakage: Using the validation set separately from the training set helps avoid data leakage, where information from the validation set inadvertently influences the training process.

Hence the validation set of size 6765 images is added into the data set.

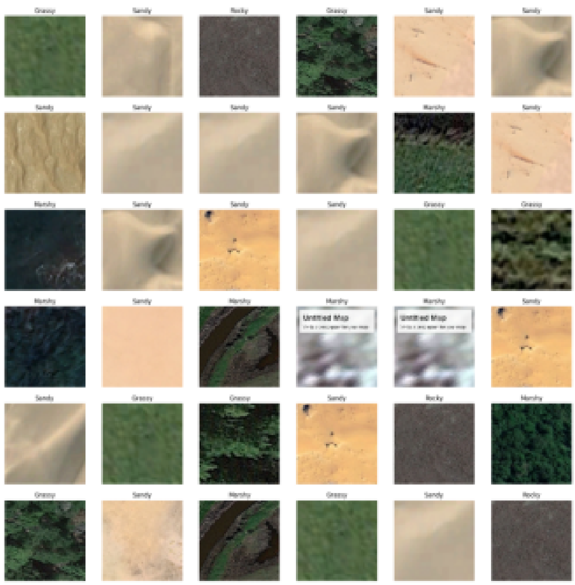

Fig1: displays the different kinds of terrains

2. Device: The device used for this process includes:

A. Intel i5

B. Nvidia Geforce-GTX 1650

C. Google Collab

D. Python Version 3.10.9

IV. METHODOLOGY

1. Using keras preprocessing (ImageDataGenerator) , the data images were re- scaled, rotated and sorted out. Then an ideal batch size was assigned for processing.

2. After assigning the batch size, a function is defined called Show_Images which displays the re-scaled and re-sized images (Fig 1.)

3. Early stopping: Early stopping is a method employed in deep learning to curb overfitting and enhance the generalization capability of neural networks. This technique entails monitoring the model's performance on a validation dataset throughout training and halting the training process once the performance starts deteriorating, thereby averting the model from assimilating noise and inconsequential patterns from the training data.

4. ReduceLROnPlateau: It is a callback commonly used in deep learning frameworks like TensorFlow and Keras to dynamically adjust the learning rate during training. It is designed to improve training performance by reducing the learning rate when a monitored metric (such as validation loss) stops improving. This technique helps the optimizer to fine-tune the model more effectively, especially during later stages of training when progress may slow down.

5. Kears layers:Keras offers an extensive selection of layers that are applicable for constructing neural networks tailored to tasks such as image classification, natural language processing, and beyond. Here are some commonly used Keras layers along with their functionalities:

1. Dense Layer (keras.layers.Dense): This layer is fully connected, meaning each neuron is linked to every neuron in both the

preceding and subsequent layers. Its purpose is to learn non-linear data transformations.

2. Convolutional Layer (keras.layers.Conv2D): This layer performs convolution operations on input data, commonly used in image processing tasks for feature extraction.

3. Pooling Layer (keras.layers.MaxPooling2D, keras.layers.AveragePooling2D): Pooling layers decrease the spatial dimensions of input data by selecting either the maximum or average value within a sliding window. This aids in lowering computational complexity and extracting prominent features.

4. Recurrent Layer (keras.layers.LSTM, keras.layers.GRU): These layers are employed to handle sequential data like time series or natural language data. LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) are adaptations of recurrent layers designed to mitigate the vanishing gradient problem found in traditional RNNs.

5. Dropout Layer (keras.layers.Dropout): Dropout is a regularization method employed to combat overfitting by randomly excluding a portion of neurons during training, compelling the network to acquire more resilient features.

6. Batch Normalization Layer (keras.layers.BatchNormalization):This layer enhances training stability and speeds up convergence by normalizing the activations of the preceding layer.

7. Activation Layer (keras.layers.Activation): This layer introduces non-linearity into the network by applying an activation function (such as ReLU, sigmoid, or tanh) to the output of the preceding layer.

8. Embedding Layer (keras.layers.Embedding): This layer is commonly used in natural language processing tasks to convert categorical data (e.g., words) into dense vector representations.

9. Flatten Layer (keras.layers.Flatten): This layer flattens the input data into a 1D array, typically used to connect convolutional layers to dense layers.

V. Conclusion

Terrain recognition using deep learning models like ResNet101 and VGG16 has shown remarkable advancements in recent years. These models, known for their deep architecture and feature extraction capabilities, have been instrumental in accurately identifying and categorizing various types of terrain in both urban and natural environments.ResNet101, with its residual learning framework, has demonstrated superior performance in handling complex terrain features. Its ability to learn deep representations effectively has enabled it to capture intricate details in terrain data, leading to robust classification results. On the other hand, VGG16, with its simpler architecture compared to ResNet101, has also proven to be highly effective in terrain recognition tasks. Its stacked convolutional layers excel in extracting hierarchical features, making it suitable for identifying different terrain types based on texture, color, and spatial patterns.

ResNet shows an efficient performance in the case of image classification and with the help of VGG16,which is another Image Classification Algorithm containing 16 layers, the accuracy increases and the processing time of the overall model and the loss entropy decreases.

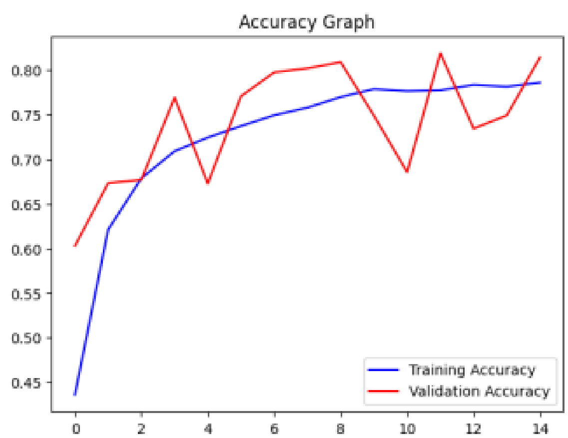

Fig2: Training accuracy vs validation accuracy graph

Fig2 shows the efficiency of the ResNet101 and VGG16 models.

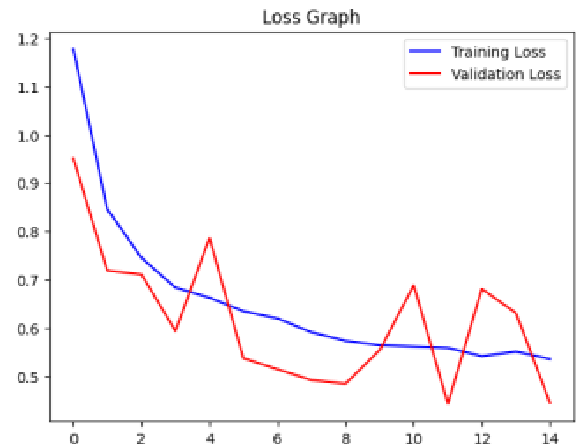

Fig3: training Lose vs the Validation loss

Fig 3 shows the comparison of loss between the training dataset and the validation dataset.

VI. References:

[1] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition.

[2] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.