Terrain Classification Using Deep Learning Techniques

1st Sumathi D
SCOPE
VIT-AP University
Andhra Pradesh, India

2nd Ansuman Sahu
22BCE7826
SCOPE
VIT-AP University
Andhra Pradesh, India

2nd DLV Sai Sameer Kumar
22BCE7194
SCOPE
VIT-AP University
Andhra Pradesh, India

2nd GVC Kiran
22BCE9574
SCOPE
VIT-AP University
Andhra Pradesh, India

***Abstract*—Terrain Classification is an indispensable component in numerous applications, such as agriculture, urban planning, disaster management, and environmental monitoring. The recent availability of high-resolution satellite imagery has created a significant challenge in the form of a need for efficient automated methods to process such imagery into actionable intelligence. This research focuses on the task of satellite image tile classification utilizing Convolutional Neural Networks (CNNs). We present a custom-designed CNN alongside a transfer learning approach that employs pre-trained architectures such as VGG16 and ResNet50 with the DeepSat-6 dataset. Preprocessing, including dimensional adjustments and label encoding, was conducted to ensure dataset compatibility with TensorFlow Pipelines. In this study, the custom CNN achieved a best validation accuracy of 97.5%, while the transfer learning approach attained a maximum of 96.4%. Model performance evaluation, conducted through classification reports and confusion matrices, as well as visual analysis of misclassifications, revealed challenges in distinguishing between relatively similar classes, such as grassland and barren land. This approach demonstrates the efficacy of deep learning for automated land use classification, thereby creating opportunities for scaling applications in remote sensing and environmental studies. Future research may focus on improving class differentiability and adapting the framework to more diverse satellite datasets for generalization.**

***Index Terms*—Terrain Classification, Deep learning, Satellite Imagery, Convolutional Neural Network (CNN)**

I. Introduction

Satellite imagery is crucial for many applications, including agriculture, city planning, natural resource management, environmental monitoring, and disaster response. In recent years, there has been significant growth in the availability of high-resolution satellite imagery, however this immense quantity of data requires processing to make it useful for analysis. Common processing tasks for satellite imagery include land use classification, object detection, semantic segmentation and object counting. Since satellite imagery can cover large areas, the main business cases for applying computer vision models to automate these tasks is that they can reduce overall costs and processing time, and potentially increase the accuracy of interpretations. Automated image processing enables mapping land uses across extensive areas that are impractical to delineate by hand. Access to digitally classified land use data facilitates applications such as tracking land use changes over time, and identifying regions of land use that are at risk. For decades, the remote sensing community has applied algorithmic techniques and classic machine learning methods (random forests, SVM, etc) to automate common satellite image processing tasks, such as classifying land uses and detecting surface features. More recently, there has been increased adoption of deep learning techniques that apply neural networks to learn classification and detection tasks.

II. Related Works

Several recent studies have advanced the field of terrain classification using machine learning and deep learning approaches. In 2023, a study developed an intelligent classification model based on SHAP for feature selection and integrated CBAM with UNet, though it faced challenges with feature interpretability and handling abrupt landform changes [1]. Another study proposed a transfer learning-based method for hyperspectral image classification, comparing CNN and MLP models. However, it encountered overfitting issues and required a large labeled dataset [2]. A study combined RNNs and CNNs for autonomous robot terrain navigation, but their approach was limited by training data and terrain variability [3]. In 2020, a study emphasized object detection in large-scale remote sensing but noted that noise and spatial textures hampered classification [4]. Many studies focused on automated terrain feature detection but struggled with small-feature detection and model generalization [5]. Long-range terrain perception with a CNN-based approach saw improvement in a study but encountered noise issues and poor generalization in new settings [6]. One study developed a robust LSTM model for proprioceptive terrain classification with noise-aware training, although it was limited by slow runtime and inadequate real-world accuracy [7]. Deep neural networks combined with CRF have been investigated for landscape classification, but their applicability to photographic imagery and the need for user-friendly tools remain unaddressed [8]. One study introduced a PolSAR classification model using a

Deep Sparse Filtering Network, though it required extensive parameter tuning and was impacted by spatial information limits for terrain boundary accuracy [9]. A detailed summary of the literature review is provided in **Table 1**.

III. Proposed Methodology

For this project, we have applied deep learning to satellite image classification by developing convolutional neural networks (CNNs) for classification of satellite image tiles. Two approaches were implemented: building a custom CNN from scratch and using two pre-trained CNN architectures. The models were trained on a benchmarking dataset (Deepsat-6) and achieved over 96% accuracy on unseen data.

A. Data Collection

This project involves the use of images from DeepSat-6 dataset. The DeepSat-6 dataset is a curated collection with pre-labelled image tiles emanating from the National Agriculture Imagery Program (NAIP), which provides aerial imagery of the U.S. at 1-meter resolution across four spectral bands: red, green, blue, and near-infrared. This dataset focuses on its subset taken out from NAIP imagery over some regions in California, and it was intended for machine learning applications, more precisely land cover classification. DeepSat-6 consists of 28×28 non-overlapping tiles. Preprocessing involved randomization to avoid biased training and evaluation. DeepSat-6 consists of six classes of land cover: barren land, trees, grassland, roads, water, and buildings. One-hot-encoded vectors were assigned for each tile representing these classes. Training and testing splits are possible thanks to the proper structuring of this dataset. It further advances the vegetation and land feature analysis by incorporating the NIR band, hence being more applicable in remote sensing and environmental studies. This process of extraction retains the high NAIP resolution while downsizing it to manageable tile sizes, which enables the effective processing of the output with machine learning models. The variance of classes within the dataset captures different terrains and urban features across California, making this a really strong dataset for the applications of land cover classification, vegetation analysis, and environmental change detection. This randomization and preprocessing of the dataset further ensure that it is ready for use in deep learning workflows, particularly in training convolutional neural networks or other image classification algorithms.

B. Data Preprocessing

After obtaining the data, we used scipy.io.readmat() to load the dataset into a dictionary. The dictionary contained key-value pairs for the test and training data, test and training labels, and a set of annotations for translating the labels into human-readable categories. The data values were all stored as ndarrays. A series of tests were applied to verify the quality of the raw data. These included testing that each image was assigned to a single category, that there were no null values in the image arrays, and that all pixel values were between 0 and 255. No data quality issues were identified. The imagery data required reworking to convert it to the format expected by Tensorflow. These steps included: 1) re-dimensioning the multidimensional image arrays from the original shape (rows, columns, channels, samples) to the channels-last convention expected by TensorFlow (samples, rows, columns, channels); 2) transposing the label arrays; and 3) creating a pandas Series of the training labels (as text) indexed by image number. After these manipulations, individual image tiles were easy to plot and identify by assigned class as shown in **Fig. 1**.

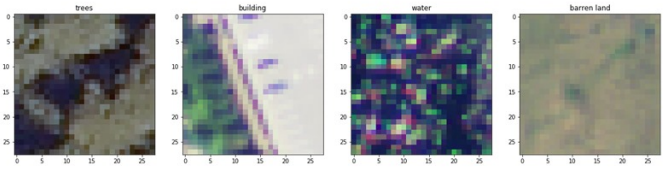

Fig. 1. Representation of Preprocessed Images

IV. Application of Deep Learning

The convolutional neural network (CNN) is a deep learning algorithm commonly used in computer vision applications. Two approaches implementing CNNs were applied to develop classification models for the DeepSat-6 data. The first model, the "baseline" CNN, uses a simple CNN architecture. The second model implements a transfer learning approach using the VGG16 model pre-trained on ImageNet. Both models were implemented in Python using the Keras interface to Tensorflow. Both classification models implement a series of convolution and pooling steps (the convolutional base) followed by a densely-connected classifier to produce the final output. The convolutions learn a hierarchy of local patterns at different scales, while the dense layers learn global patterns.

A. Baseline Model

The baseline model was implemented using TensorFlow's Keras with the following architecture as shown in **Fig. 2**. The

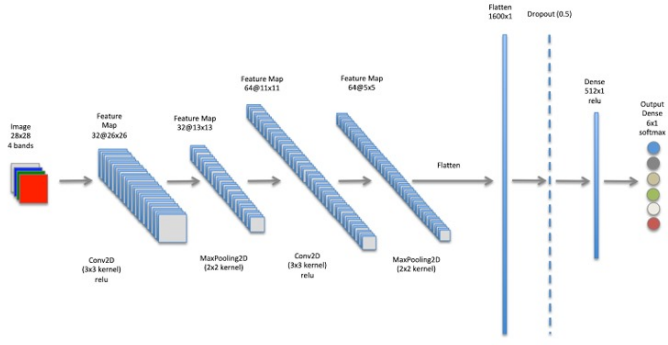

Fig. 2. Representation of Baseline Model Architecture

input to the baseline CNN is tensor representing a 28x28 4-band image tile. The first convolutional layer applies 32 convolutions (with a 3x3 kernel) to the input image followed

TABLE I
SUMMARY OF TERRAIN CLASSIFICATION STUDIES

Sl. No	Title of Paper	Authors	Contributions	Limitations
1	Intelligent classification and analysis of regional landforms based on automatic feature selection (2023)	Yuexue Xu, Hong-Chun Zhu, Zhiwei Lu, Yanrui Yang, Guobin Zhu	Feature selection importance based on SHAP method. Integration of CBAM into UNet for landform classification.	Lack of consideration for interpretability and transparency in feature selection. UNet network unable to capture abrupt landform features.
2	Terrain Classification using Transfer Learning on Hyperspectral Images: A Comparative Study (2022)	Uphar Singh, Kumar Saurabh, Neelaksh Trehan, Ranjana Vyas, Om Prakash Vyas	Transfer learning approach for hyperspectral image classification. Comparison of CNN and MLP models for optimum architecture selection.	Overfitting observed with scaling layers, affecting accuracy and training time. Long training time and large labeled dataset requirement.
3	Recurrent and convolutional neural networks for deep terrain classification by autonomous robots (2021)	Fabio Vulpi, Annalisa Milella, Roberto Marani, Giulio Reina	Utilized recurrent and convolutional neural networks for terrain classification. Improved autonomous robot navigation through deep learning techniques.	Limited by the size of the training dataset. Performance affected by varying terrain conditions. Limited exploration of different types of terrains.
4	Adoption of Machine Learning in Intelligent Terrain Classification of Hyperspectral Remote Sensing Images (2020)	Li Yanyi, Jian Wang, Tong Gao, Qiwen Sun, Liguo Zhang, Mingxiu Tang	Object detection importance in remote sensing. Terrain changes analysis for large-scale ground objects.	Noise and spatial texture impact classification performance in hyperspectral images. Non-linear mapping increases difficulty in solving classification problems.
5	Automated terrain feature identification from remote sensing imagery: A deep learning approach (2020)	Wenwen Li, Chia Yu Hsu	Terrain feature identification using deep learning. Enhancing terrain analysis and landscape scene interpretation through feature detection.	Challenges in detecting small or subtle terrain features. Lack of discussion on model generalization and transferability. Limited exploration of feature detection in varying environmental conditions.
6	Long-range terrain perception using convolutional neural networks (2018)	Wei Zhang, Chen Qi, Weidong Zhang, Xuanyu He	Near-to-far strategy with CNNs for terrain perception. End-to-end training architecture to enhance terrain segmentation accuracy.	Redundancy and noise in reference maps inclusion. Poor generalization ability of traditional classifiers for new environments. Need for improved accuracy and robustness in wild terrain perception.
7	Deep Spatiotemporal Models for Robust Proprioceptive Terrain Classification (2018)	Abhinav Valada, Wolfram Burgard	Deep LSTM model for proprioceptive terrain classification. Noise-aware training scheme for robust real-world deployments.	Inadequate accuracy, robustness, and slow run-times hinder widespread adaptation. Existing techniques lack robustness, accuracy, and efficiency for deployment.
8	Landscape Classification with Deep Neural Networks (2018)	Daniel Buscombe, Andrew C. Ritchie	Application of deep learning to landscape classification. Efficient approach using conditional random field (CRF).	General usefulness of deep learning on photographic imagery unproven. Need for demonstrable successes and accessible tools for wider application.
9	Terrain Classification with Polarimetric SAR based on Deep Sparse Filtering Network (2016)	Hongying Liu, Min Qiang, Chen Sun, Jin Zhao, Shuyuan Yang, Biao Hou, Jie Feng, Licheng Jiao	New classification method for PolSAR data. Design of Deep Sparse Filtering Network (DSFN) for feature extraction.	Deep networks have many parameters to tune. Lack of spatial information affects terrain boundary division. Improved classification accuracy needed as compared to conventional deep networks.

by the rectified linear unit (relu) activation function to introduce non-linearity into the model. The result is a 3D tensor (feature map) with dimensions 26x26X32. Each of the two max pooling operations downsamples the feature maps by a factor of 2. This downsampling allows each convolution stage to learn patterns at a different scale. Due to the small extent of the input image tiles, only two stages of convolutions and max pooling were applied. After the second max pooling operation, the input data has been transformed to a feature map with dimensions 5x5x64. This representation is flattened before being input a densely-connected classifier consisting of a fully-connected dense layer followed by a fully-connected dense output layer. The final dense layer estimates the class of the image, by applying the 'softmax' activation function to generate a probability of the input image being in each class. The class with the highest probability is assigned to the image. As shown above, a dropout layer with a dropout rate of 0.5 was inserted after the flatten layer. The dropout layer randomly zeros out a portion of the output features of the prior layer during training, and is used as a regularization tool to prevent overfitting. This model has a total of 842,470 parameters. Approximately 20,000 of these parameters are the weights to be trained in the convolution filters and the bias values to be applied after the convolutions. The majority of parameters are the weights of the first fully-connected dense layer of the classifier. Compiling a model in Keras involves selecting a loss

function, optimizer function, and output metrics to be reported during training. Since this is a multi-class classification problem, the categorical cross-entropy loss function was selected, which penalizes misclassifications based on the differences in log-probabilities between predicted and actual outcomes. The baseline model was fit by backpropagation using the stochastic gradient descent (SGD) optimizer with a learning rate of 0.01 and batch size of 512. SGD was chosen by trial and error by testing different optimizers (RMSprop, Adam) and learning rates. SGD performs gradient descent using small batches of training samples. The Keras ImageDataGenerator class was used to create python generators to supply batches of pre-processed images during the model fitting process. The pre-labeled training data was randomly split into training and validation datasets, comprised of 280,000 and 64,800 images, respectively. Each generator was given a batch size of 512 images. Given the large size of the dataset, data augmentation was not needed to prevent overfitting.

B. Transfer Learning with VGG16 and ResNet50

Transfer learning involves repurposing a convolutional base that was previously trained on a large imagery dataset for a new computer vision problem. Pre-trained CNNs can recognize general spatial features at different scales. By adding a custom backend (such as a dense classifier) to the pretrained-CNN, the combined model can be applied to solve new computer vision tasks. For this project the VGG16 and ResNet50 models pre-trained on ImageNet was applied to classify the DeepSat-6 dataset. The following diagram in **Fig. 3** shows the architecture used to construct the model, which was again implemented in Keras.

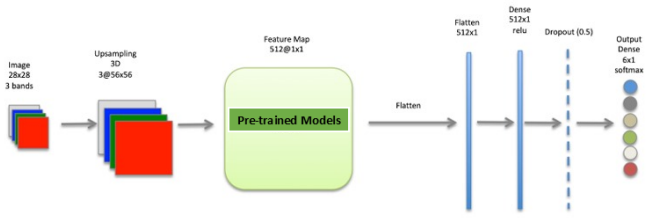

Fig. 3. Architectural Representation of Pre-trained Model Working

The pre-trained model layers required some additional pre-processing of the input data pipeline used for the baseline model. The CNN from the baseline model was trained on all 4 bands (R,G,B,IR) of the DeepSat-6 image tiles. Since the pre-trained models were only trained on images with 3 bands, the IR band was dropped from the input data. In addition, VGG16 and ResNet50 have a minimum image size of 32x32 pixels. Since the image tiles in the dataset are 28x28, the Upsampling3D layer was added to upsample the rows and columns by a factor of two, transforming each image to dimensions of 56x56x3 for input to the pre-trained models. The dense classifier on the backend was similar to that used in the baseline model. There are two approaches for feature extraction using transfer learning: 1) running the convolutional base over the data and saving the resulting numpy array, then training a dense classifier on the array; or 2) extending the convolutional base by directly connecting a dense classifier, then running the entire model on the input data. Although more computationally intensive, the second approach was selected, due to its simplicity and potential to enable image augmentation and/or fine tuning of the top layers of VGG16 and ResNet50 if necessary (neither was needed). To implement the second approach, the convolutional base was frozen so the weights of the VGG16 and ResNet50 layers would not adjust during training. The models were fit with the RMSProp optimizer (learning-rate=0.0001) over 30 epochs.

V. Results and Analysis

In this section, we highlight the key outcomes of our research, emphasizing the main findings and observations obtained from our experiments and analyses.

A. Baseline Model Evaluation

The accuracy and loss values for the training and validation datasets, as shown in **Fig. 4**, were recorded over the course of training. The training process achieved more than 95% accuracy on the validation data after only a few epochs. As shown below, additional epochs beyond 10 did not produce significant gains in performance, and early stopping was triggered after 23 epochs. There was no evidence of overfitting to the training data. The training history plots suggested that training loss was slightly higher than the validation loss after each epoch.

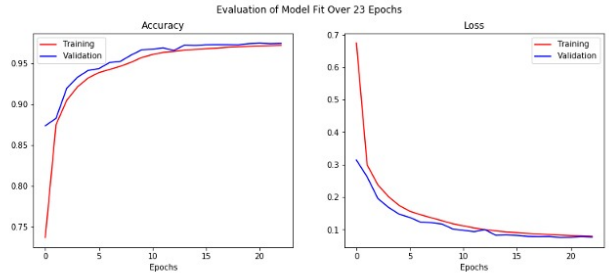

Fig. 4. Visualization of Accuracy and Loss Values of Baseline Model

This counterintuitive result may be the result of applying regularization with a dropout layer. Regularization increases the ability of the model to generalize to unseen data at the expense of the accuracy of individual training predictions. Dropout is only applied during the training stage and not during validation, this can result in validation scores that exceed training scores. Also, training loss is continuously measured over an epoch, while validation loss is measured at the end of an epoch. This can produce higher training losses when the model fit improves over the epoch. The results for the validation set during training, as shown in **Fig. 5**, suggested that the model would generalize well to unseen data. This was tested by using the baseline model to predict the land

use classification on a set of 81,000 labeled training images provided with the DeepSat-6 dataset. The model predicted the image classes of the held-out set with 97.5% accuracy. The following bar chart in **Fig. 6** compares the frequency of the labelled and predicted classes, and demonstrates that the classes are similarly distributed, with some misclassifications of barren land, trees, and grassland.

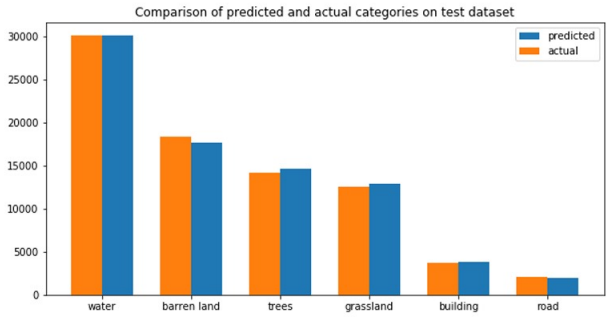

Fig. 5. Comparison of Predicted vs Actual Categories on Test Dataset

The classification report presents the precision (proportion of classifications into that class that were correct), recall (proportion of items in the class that were identified), and f1-score (harmonic mean of precision and recall) of each class. The results show slightly lower precision and recall for the grassland and road classes. To thoroughly evaluate the baseline model's performance, a confusion matrix was generated, as shown in **Fig. 7**.

Classification Report:

	precision	recall	f1-score	support
barren land	0.99	0.95	0.97	18367
building	0.95	0.97	0.96	3714
grassland	0.93	0.95	0.94	12596
road	0.93	0.87	0.90	2070
trees	0.97	0.99	0.98	14185
water	1.00	1.00	1.00	30068
accuracy			0.97	81000
macro avg	0.96	0.96	0.96	81000
weighted avg	0.98	0.97	0.97	81000

Fig. 6. Classification Metrics of the Baseline Model

This matrix provides a detailed overview of the model's classification accuracy across each of the six land use categories, offering insight into how well the model distinguishes between different classes and where it encounters challenges. The confusion matrix reveals that the most frequent classification errors involve the model confusing barren land with grassland, as well as misclassifying grassland as trees. These misclassifications suggest that the spectral or spatial characteristics of barren land and grassland, and similarly those of grassland and tree-covered areas, may share similarities in the feature space used by the model. Such overlap can make it difficult for the model to precisely differentiate between these classes, especially in areas where vegetation density or land cover transitions gradually. By identifying and understanding these specific areas of misclassification through the confusion matrix, further adjustments to the model and training data can be targeted, helping to improve the model's overall accuracy and its ability to distinguish between closely related land use categories.

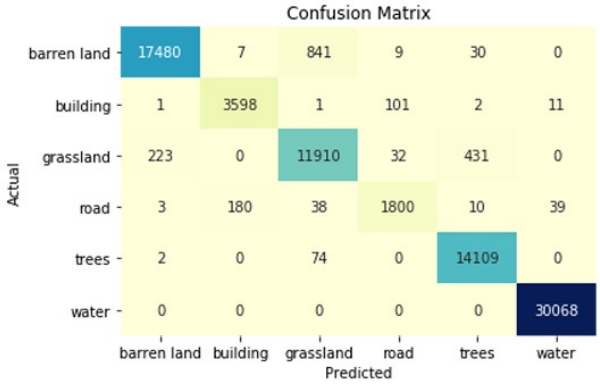

Fig. 7. Representation of Confusion Matrix of Baseline Model

The following set of images, as illustrated in **Fig. 8**, demonstrate a few misclassifications made by the model. Some misclassifications occur when multiple classes are present in the same image, resulting in ambiguous class assignments.

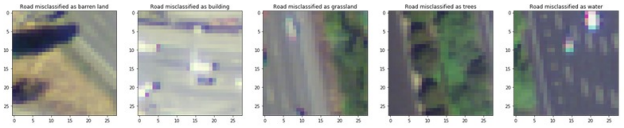

Fig. 8. Representation of Misclassified Predictions by Baseline Model

B. Pre-trained VGG16 Model Evaluation

The accuracy and loss curves for fitting the transfer learning model, shown below in **Fig. 9**, suggest no overfitting after 30 epochs. Additional iterations may have produced minor improvement in model fit.

The VGG16 model was fit to the unseen data (DeepSat-6 test set). The resulting accuracy score was 96.4%, slightly lower than the baseline model (97.5%). The following classification report and confusion matrix, as shown in **Fig.10** and **Fig. 11** respectively were used to assess how well the transfer learning model classified individual land use classes.

The VGG16 model was slightly stronger at classifying water and road tiles in the test dataset compared to the baseline model, and slightly worse at the other categories. Additional training iterations may have reduced this difference. The most difficult category for the transfer learning model to classify was grassland. The following examples of misclassifications,

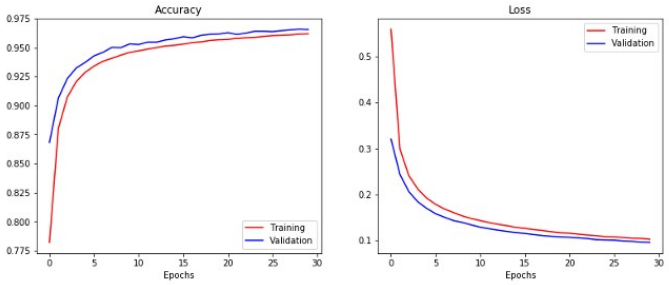

Fig. 9. Visualization of Accuracy and Loss Values of VGG16 Model

Classification Report:

	precision	recall	f1-score	support
barren_land	0.94	0.94	0.94	18367
building	0.95	0.96	0.95	3714
grassland	0.91	0.91	0.91	12596
road	0.96	0.93	0.94	2070
trees	0.97	0.97	0.97	14185
water	1.00	1.00	1.00	30068
accuracy			0.96	81000
macro avg	0.95	0.95	0.95	81000
weighted avg	0.96	0.96	0.96	81000

Fig. 10. Comparison of Predicted vs Actual Categories on Test Dataset

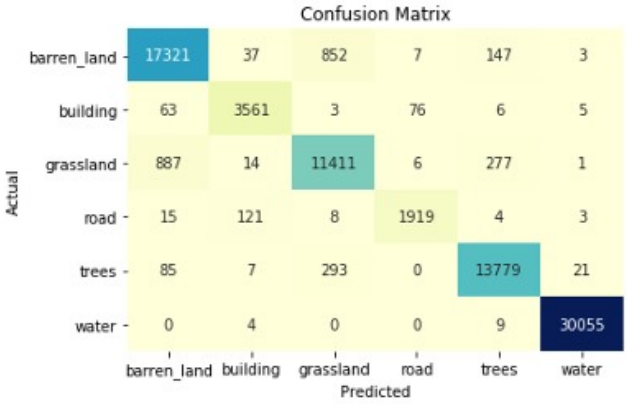

Fig. 11. Classification Metrics of VGG16 Model

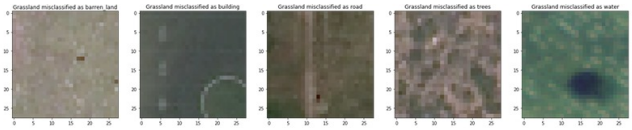

Fig. 12. Representation of Confusion Matrix of VGG16 Model

as shown in **Fig. 12**, suggest that there is substantial variation within the grassland class, and that a human classifier may not have been able to do much better.

C. Pre-trained ResNet50 Model Evaluation

The accuracy and loss plots, as shown in **Fig. 13**, indicate that the model generalizes effectively on both training and validation sets, achieving approximately 80% accuracy in 25 epochs, suggesting effective learning without significant overfitting or underfitting. Furthermore, the loss curves exhibit a continuous downward trend toward the minimum for both training and validation sets, indicating substantial model stability during the training process.

A comparative analysis of the predicted category counts with the actual counts from the test dataset, as illustrated in **Fig. 14**, demonstrates the model's robustness for most categories, such as water and barren land, where the predicted values closely align with the actual counts. However, discrepancies are observed, particularly for categories like grassland or trees, potentially due to overlapping visual features or similar spatial characteristics.

However,the performance of the ResNet50 model was lower as compared to the baseline model and the pre-trained VGG16 model. The model achieved an accuracy of 79%, which in case of other two models was very high. The classification metrics of the model is presented in **Fig. 15**.

Additional insights into the model's classification performance are provided by the confusion matrix in **Fig. 16**. The matrix reveals high accuracy for classes such as water and trees, as evidenced by their strong diagonal values. Nevertheless, some misclassifications are present, such as a portion of grassland instances being misclassified as barren land and road instances occasionally overlapping with building categories. These errors suggest that classes with similar characteristics or spatial proximity are incorrectly identified, possibly due to subtle differences in texture, colour, or contextual information.

VI. Conclusion and Future Work

Two approaches were implemented: building a CNN from scratch and using pre-trained CNN architectures to automate classification of satellite image tiles into 6 land use classes. Both approaches achieved over 96% accuracy on held-out data. A workflow was also developed to apply classification models trained on DeepSat-6 to classify contiguous regions of NAIP data, and to produce gridded classification outputs that can be used for other spatial data workflows. Follow-on work to extend this application could include developing an additional workflow to georeference (i.e. restore the spatial metadata) the reassembled land use classification. This enhancement would enable more complex spatial evaluations like calculating statistics over different regions (like the distribution of classes by county), and provide the capability to develop user interfaces to interact with the classified output. A limitation of the current classification method is the reduction in resolution that occurs when classification is performed on entire image tiles rather than on individual pixels. This tile-based approach sacrifices some spatial detail, potentially leading to less precise boundaries between land use categories. To mitigate this limitation and improve output resolution, a more refined workflow could be implemented. This refined approach would involve classifying partially overlapping tiles

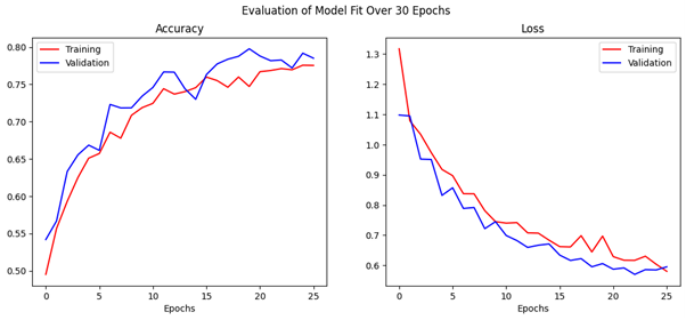

Fig. 13. Visualization of Accuracy and Loss Values of ResNet50 Model

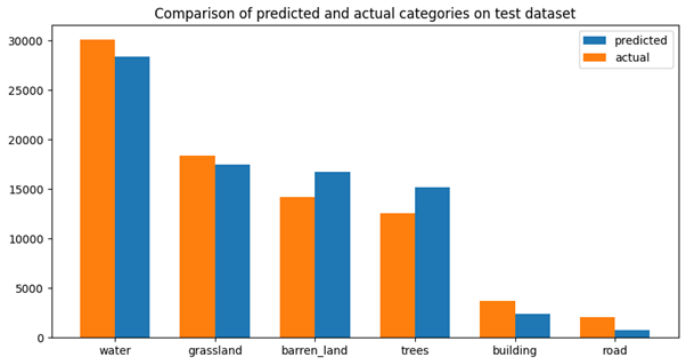

Fig. 14. Comparison of Predicted vs Actual Categories on Test Dataset

and assigning the classification result specifically to the center-most pixels of each tile. This method would increase the spatial accuracy of the final classification map by enhancing pixel-level resolution and providing finer granularity in land use delineation across large areas.

REFERENCES

[1] Y. Xu, H. Zhu, Z. Lu, Y. Yang, and G. Zhu, "Intelligent classification and analysis of regional landforms based on automatic feature selection," *Earth Surface Processes and Landforms*, vol. 49, 11 2023.

[2] U. Singh, K. Saurabh, N. Trehan, R. Vyas, and O. Vyas, "Terrain classification using transfer learning on hyperspectral images: A comparative study," pp. 1–6, 11 2022.

[3] F. Vulpi, A. Milella, R. Marani, and G. Reina, "Recurrent and convolutional neural networks for deep terrain classification by autonomous robots," *Journal of Terramechanics*, vol. 96, pp. 119–131, 08 2021.

[4] Y. Li, J. Wang, T. Gao, Q. Sun, L. Zhang, and M. Tang, "Adoption of machine learning in intelligent terrain classification of hyperspectral remote sensing images," *Computational Intelligence and Neuroscience*, vol. 2020, pp. 1–13, 09 2020.

[5] W. Li and C.-Y. Hsu, "Automated terrain feature identification from remote sensing imagery: a deep learning approach," *International Journal of Geographical Information Science*, vol. 34, pp. 1–24, 11 2018.

[6] W. Zhang, Q. Chen, W. Zhang, and X. He, "Long-range terrain perception using convolutional neural networks," *Neurocomputing*, vol. 275, 09 2017.

[7] A. Valada and W. Burgard, "Deep spatiotemporal models for robust proprioceptive terrain classification," *The International Journal of Robotics Research*, vol. 36, p. 027836491772706, 08 2017.

[8] D. Buscombe and A. Ritchie, "Landscape classification with deep neural networks," *Geosciences*, vol. 8, 07 2018.

[9] H. Liu, Q. Min, C. Sun, J. Zhao, S. Yang, B. Hou, J. Feng, and L. Jiao, "Terrain classification with polarimetric sar based on deep sparse filtering network," pp. 64–67, 07 2016.

Classification Report:

	precision	recall	f1-score	support
barren_land	0.83	0.76	0.80	18367
building	0.77	0.49	0.60	3714
grassland	0.59	0.83	0.69	12596
road	0.64	0.23	0.33	2070
trees	0.74	0.79	0.77	14185
water	0.90	0.85	0.88	30068
accuracy			0.79	81000
macro avg	0.75	0.66	0.68	81000
weighted avg	0.80	0.79	0.78	81000

Fig. 15. Classification Metrics of ResNet50 Model

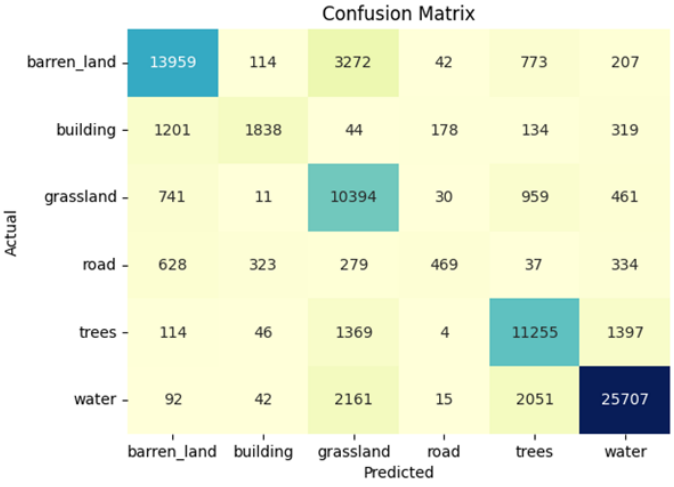

Fig. 16. Representation of Confusion Matrix of ResNet50 Mode