
TERRAIN CLASSIFICATION ML MINI-PROJECT

Course: UE23CS352A Machine Learning
Team Members: Satwik R Bankapur, Sathwik H S
Date: October 13, 2025
Section: I

---

1. Problem Statement

The objective of this project is to develop an automated terrain classification system using machine learning techniques. The system classifies terrain images into distinct categories to support applications in autonomous navigation, robotics, and environmental monitoring. We implemented and compared three different approaches: Support Vector Machines, Random Forest, and Convolutional Neural Networks.

---

2. Approach and Methodology

Dataset

- **Source:** Kaggle Terrain Dataset
- **Size:** 1,000 images
- **Resolution:** 224×224×3 pixels
- **Classes:** 5 terrain types (Desert, Forest, Mountain, Grassland, Rocky)

Data Preprocessing

Images were resized to 224×224 pixels and normalized to the range. For CNN training, data augmentation techniques including rotation, shifting, and horizontal flipping were applied to improve model generalization.Instructions.pdf

Feature Extraction (Traditional ML)

For SVM and Random Forest models, we extracted 15 hand-crafted features:

- Color features: Mean and standard deviation of RGB and HSV channels
- Texture features: Gradient magnitude statistics using Sobel operators
- These features capture essential terrain characteristics like color distribution and surface texture patterns

Models Implemented

1. Support Vector Machine (SVM)

- Kernel: Radial Basis Function (RBF)
- Feature standardization using StandardScaler
- Optimized for high-dimensional feature space

2. Random Forest

- Ensemble of 100 decision trees
- Bootstrap aggregating for variance reduction
- Feature importance analysis for interpretability

3. Convolutional Neural Network (CNN)

- Architecture: Transfer learning with MobileNetV2 (pre-trained on ImageNet)
- Custom classification head: GlobalAveragePooling → Dense(128) → Dropout(0.5) → Dense(64) → Dropout(0.3) → Dense(5)
- Optimizer: Adam with learning rate 0.001
- Training: 15 epochs with batch size 32

---

3. Implementation Overview

The project was implemented in Google Colab using Python 3.x. The implementation pipeline consists of:

1. **Data Loading:** Automated dataset download from Kaggle using API
2. **Preprocessing:** Image resizing, normalization, and augmentation
3. **Model Training:** Sequential training of all three models with proper train-test splits (80:20)
4. **Evaluation:** Accuracy metrics, confusion matrices, and classification reports
5. **Visualization:** Comparative performance analysis and result visualization

Technologies Used:

- TensorFlow/Keras for deep learning
- Scikit-learn for traditional ML algorithms
- OpenCV for image processing
- Matplotlib and Seaborn for visualization

4. Results

Model	Accuracy	Approach	Training Time
SVM	100.0%	Traditional ML	Fast
Random Forest	100.0%	Ensemble ML	Fast
CNN (MobileNetV2)	97.0%	Deep Learning	Moderate

Key Observations:

- Traditional ML models achieved perfect classification accuracy, demonstrating the effectiveness of engineered features
- CNN achieved 97% accuracy, showing strong performance with transfer learning
- All models demonstrated excellent generalization on unseen test data
- Feature engineering proved crucial for traditional ML success

5. Challenges and Conclusions

Challenges Faced:

1. **Dataset Structure:** Initial difficulties with dataset loading and folder structure recognition, resolved through robust error handling
2. **Feature Selection:** Identifying optimal features for traditional ML required experimentation with various color spaces and texture descriptors
3. **Computational Resources:** CNN training required GPU acceleration, managed through Google Colab

Conclusions:

The project successfully demonstrated that both traditional machine learning and deep learning approaches are effective for terrain classification. The perfect accuracy of SVM and Random Forest indicates that well-engineered features can match or exceed deep learning performance for structured image classification tasks. Transfer learning with MobileNetV2 proved highly effective, achieving near-perfect accuracy with minimal training time. The comprehensive comparison provides valuable insights into model selection for similar image classification problems.

Future Work:

- Expand dataset with more diverse terrain types and weather conditions

- Implement additional CNN architectures (ResNet, EfficientNet)
- Deploy the model as a web service for real-time terrain classification
- Add model explainability features using LIME or SHAP

Repository: https://github.com/satwik-bankapur/terrain_classification_ml_project/tree/main
Notebook: terrain_classification.ipynb
