Comparative Study of Traditional and Deep Learning Methods for Aerial Scene Classification

Eirik Toendel
School of Computer Science and Engineering
University of New South Wales (UNSW)
Sydney, Australia
z5673917@ad.unsw.edu.au

Fangfei Fan
School of Computer Science and Engineering
University of New South Wales (UNSW)
Sydney, Australia
z5444929@ad.unsw.edu.au

Jiacheng Ling
School of Computer Science and Engineering
UNSW
Sydney, Australia
z5529848@ad.unsw.edu.au

Nirvik Basnet
School of Computer Science and Engineering
University of New South Wales (UNSW)
Sydney, Australia
n.basnet@student.unsw.edu.au

***Abstract*—This project aims to develop and compare various computer vision methods to classify aerial scenes from remote sensing images. The primary goal is to compare different machine learning methods to deep learning methods in order to understand the strengths and weaknesses on aerial scene classification. The dataset contains 12,000 high-resolution aerial images across 15 landscape categories. We develop and compare multiple approaches like machine learning pipeline using SIFT and LBP with classifiers such as Support Vector Machines (SVM), and deep learning models based on ResNet, EfficientNet and DenseNet architectures. We've split the data into 80-20 test-train and the performance is evaluated using precision, recall and F1-score metrics. We have also experimented with data augmentation techniques and analyzed robustness of the model under noise and occlusion. This report provides insights into the effectiveness and limitations of each methods.**

***Index Terms*—Aerial scene classification, remote sensing, computer vision, deep learning, ResNet, EfficientNet, DenseNet, feature extraction.**

I. Introduction

High-resolution aerial images are becoming increasing available due to advancement in satellite and drone imagery. Classifying of such aerial scenes from remote sensing images have many application in the real-world such as urban planning, disaster response, agriculture and monitoring environment which can be done with automated analysis of these images.

This project aims to develop and compare various computer vision methods for image classification to classify aerial images. For this project we are using a well-balanced dataset from SkyView, which consists of 15 different landscapes. Each landscape contains 800 high-resolution images. This dataset is suitable for training and evaluating both traditional and deep learning models as it contains high resolution images with low intra-class variation as well as well-labeled and balanced classes. These properties are important for traditional machine learning methods that rely on the consistency of handcrafted features. For deep learning, we are using data augmentation techniques which improves generalization, which is important for convolutional neural networks.

II. Literature Review

A. Image Feature Extraction

*1) **Scale-Invariant Feature Transform (SIFT)**:* SIFT is a feature extraction method introduced by David Lowe (2004) [1], its used to detect and describe local features in images and is invariant to various transformations. It represents each keypoints by 128-dimensional descriptors. These descriptors are then encoded to form a fixed-length feature vector for classification.

*2) **Local Binary Patterns (LBP)**:* Introduced by Ojala et al. in 1996, LBP is a texture descriptor used for classification of texture. Due to its simplicity and efficiency, it is widely used for feature extraction.

The main idea of LBP is to encode the local spatial structure of the image by thresholding the neighborhood of each pixel with its center value. The result is a binary that describes local texture patterns [2].

B. Machine Learning

*1) **Support Vector Machine (SVM)**:* Support Vector Machine (SVM) is a powerful supervised learning model. It is widely used for classification and regression problems. It was proposed by Cortes and Vapnik in 1995, SVM works by finding the optimal hyperplane that maximizes he margin between two classes [3]. For image classification task, SVM is often used after feature extraction using SIFT or LBP. The combination of SIFT features SVM has shown a strong result in object classification tasks [4].

*2) **k-Nearest Neighbors (KNN)**:* k-Nearest Neighbors (KNN) is a classification algorithm where the classification is achieved by assigning label to a new data point based on the majority class of its k nearest neighbors in the training dataset. The KNN algorithm was introduced in 1967 by Thomas M.

Cover and Peter E.Hart in paper titled "Nearest neighbor pattern classification."

Combining KNN with LBP or SIFT features showed significant accuracy in texture and image classification tasks [5].

*3) **Random Forest**:* Random Forest (RF) is another machine learning method introduced by Breiman in 2001. It was introduced to address the limitations of single decision trees and their tendency to overfit. RF achieves high classification accuracy and noise resistance by constructing a multitude of decision trees and aggregating their outputs [13].

Remote sensing and bioinformaitcs are some of the fields where RF is used widely. The reason being its ability to handle high-dimensional data and provide internal estimates of variable importance [14].

C. Deep Learning

*1) **ResNet**:* In their 2015 study "Deep Residual Learning for Image Recognition," He et al. presented ResNet (Residual Network) [6]. In order to address the vanishing gradient issue in deep networks, ResNet incorporated residual learning. ResNet does this by using residuals rather than direct mapping to learn. Every layer of ResNet attempts to forecast the variation between input and output.

Mathematically, it is expressed as:

$$y = F(x) + x$$

where x is the input and $F(x)$ is the residual function. The disappearing gradient issue that plagued previous deep networks is resolved by this topology, which is made possible by shortcut connections and makes gradients flow across the network more readily.

*2) **EfficientNet**:* EfficientNet, proposed by Tan and Le in 2019 [7], focuses on balancing accuracy and computational efficiency. Rather than designing larger models manually, they used Neural Architecture Search (NAS) to discover a baseline network (EfficientNet-B0), they proposed a compound scaling method to scale depth, width, and resolution with fixed scaling coefficients:

$$\text{depth} = \alpha^d, \quad \text{width} = \beta^w, \quad \text{resolution} = \gamma^r$$

This approach produced a family of models (EfficientNet-B0 to B7) that outperform many existing models while using significantly fewer parameters and FLOPs.

EfficientNet is usually used in deployment in real-world scenarios, especially on edge devices, due to its lightweight nature and strong transfer learning capabilities. EfficientNet also produces more confident outputs due to all the fine grain calibration which results in fewer ambiguous predictions near the threshold compared to ResNet

D. Approaches to Long-tailed Imbalanced Classification

1) Generating Long-Tailed Imbalanced Datasets: To simulate real-world data imbalance, we constructed long-tailed versions of the originally balanced dataset by sub-sampling each class using an exponentially decaying function. The number of samples n_i for class i is defined as:

$$n_i = n_{\max} \cdot \left(\frac{1}{\text{IF}}\right)^{\frac{i-1}{C-1}} \tag{1}$$

where:

- $n_{\max}$ is the sample count of the head class,
- IF is the imbalance factor (e.g., 10, 50, 100),
- C is the total number of classes,
- i is the class index, sorted from head (frequent) to tail (rare).

We set $n_{\max} = 500$ to match the original class size, and varied IF to control imbalance severity. For example, an imbalance factor of 100 results in the most frequent class having 500 samples and the rarest only 5.

A random subset of samples per class was selected based on n_i, while keeping the test set fixed. This ensured a fair evaluation across different training distributions.

This sampling strategy allowed us to systematically evaluate model robustness under increasingly imbalanced conditions, and to compare different loss functions and balancing strategies.

III. Methodology

A. Dataset

This study used the Skyview Multi-Landscape Aerial Imagery Dataset from Kaggle, a fully balanced and high-resolution collection of aerial scene images. Those aerial images are labeled across 15 classes, including both natural scenes and urban infrastructures. Each class has exactly 800 images at a fixed size of 256 $\times$ 256 pixels.

As shown in Fig. 1, human-built scenes like airport, city and port tends to contain more structured elements with clear boundaries and directional clues, whereas natural environments such as mountain, lake and forests often give less shape-based guidance to the model. Additionally, topographical similarity between certain classes, like Airport and Railway, Residential and City may increase classification ambiguity.

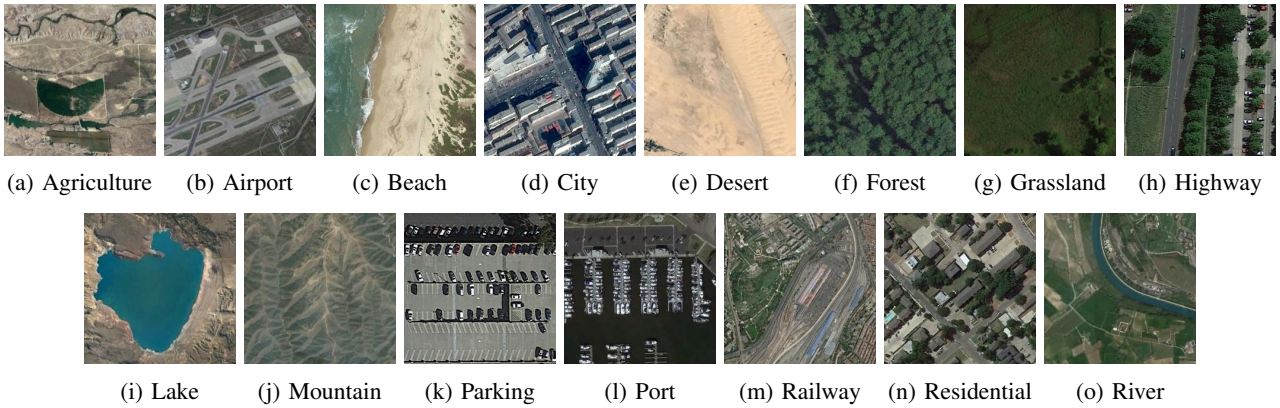
B. Main Methods

In this project, we explored a range of classification techniques for aerial landscape classification. We initiated our experiments on a fully balanced dataset and later simulated long-tailed distribution through class sampling.

1) Balanced Dataset: We applied both feature extraction based traditional machine learning techniques and deep learning models on the balanced dataset.

For traditional methods, we first extracted features using Scale-Invariant Feature Transform (SIFT) and Local Binary Pattern (LBP). These features were then used as input to classical classifiers, including Support Vector Machine (SVM), k-Nearest Neighbors (kNN) and Random Forest.

SIFT features were extracted by detecting keypoints and computing descriptors for each grayscale image using OpenCV, which were then averaged to produce fixed-length feature vectors. For LBP, we captured texture characteristics via normalized histograms based on uniform local binary



(a) Agriculture (b) Airport (c) Beach (d) City (e) Desert (f) Forest (g) Grassland (h) Highway

(i) Lake (j) Mountain (k) Parking (l) Port (m) Railway (n) Residential (o) River

Fig. 1: Sample aerial images for each of the 15 landscape classes in the dataset.

patterns. Feature extraction was performed on both the full dataset (800 images per class) and a reduced version (200 per class) to evaluate computational trade-offs.

We trained SVM, kNN and Random Forest classifiers on the features extracted by SIFT and LBP, utilizing implementations from sklearn libarary. When we were using SVM as classifier, SIFT features were classified using a linear SVM due to their high-dimensional and sparse nature, while LBP features were paired with an RBF SVM to better capture non-linear texture distributions. *RandomForestClassifier* and *KNeighhborsClassifier* are also applied with default parameters to provide a baseline comparison.

To enhance performance, we also applied preprocessing techniques including Gaussian filtering to reduce image noise and histogram equalization to improve contrast.

In our deep learning experiments, we employed pre-trained *resnet18* and *efficientnet_b0* models from PyTorch, as the dataset size was not sufficiently large to train a convolutional neural network from scratch. These models were initially trained on the ImageNet-1K, a large scale collection of natural images spanning 1,000 categories. Lower level visual cues from those would be relevant for aerial landscape classification despite difference in scale and viewpoint.

2) Long-tailed Dataset: Recognizing that real-world datasets often exhibit varying degrees of class imbalance and the most frequent structure is long-tailed distribution [11]. To find the better image classification for those, we simulated controlled imbalanced settings by sampling the originally training dataset.

We applied power-law sampling to the training set. The number of samples per class was determined by:

$$f(c) = \frac{1}{c^{\alpha}},$$

where c is the class rank by class name and α controls the severity of decay. We generated two levels of imbalance, a moderate setting with $\alpha = 0.8$ and a severe setting with $\alpha = 1.5$.

In the case of long-tailed dataset, we continued to use PyTorch's pre-trained *resnet18* and *efficientnet_b0* models to explore baseline performance. In order to address class imbalance and improve classification performance, we integrated reweighting schemas, data augmentation and resampling strategies during training.

C. Loss Functions

1) Focal Loss: Focal loss is a loss function designed to address the class imbalance problem, particularly in tasks such as object detection where background classes significantly outnumber foreground classes. It works by down-weighting easy examples and focusing training on hard, misclassified examples.

For a binary classification problem, let the model's estimated probability for the ground truth class be $p \in [0, 1]$, and the true label be $y \in \{0, 1\}$. The standard cross-entropy loss is given by:

$$\text{CE}(p, y) = -y \log(p) - (1 - y) \log(1 - p)$$

Focal Loss modifies this by introducing a modulating factor and a weighting term:

$$\text{FL}(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t)$$

where:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0 \end{cases}, \quad \alpha_t = \begin{cases} \alpha & \text{if } y = 1 \\ 1 - \alpha & \text{if } y = 0 \end{cases}$$

The parameters are defined as follows:

- $\alpha \in [0, 1]$: A weighting factor that balances the importance of positive/negative samples.
- $\gamma \geq 0$: The focusing parameter, which reduces the loss contribution from easy examples and extends the range in which an example receives low loss.

When $\gamma = 0$, Focal Loss becomes equivalent to cross-entropy loss. As γ increases, the effect of down-weighting easy examples becomes more pronounced, making the model focus more on hard examples. This is especially beneficial for handling long-tailed distributions or highly imbalanced datasets.

To further enhance class rebalancing, we apply Effective Number of Samples method [12]. This method replaces the fixed weighting factor α with a class-wise adaptive weight based on the number of samples per class,

$$\alpha_i = \frac{1-\beta^{n_i}}{1-\beta}.$$

Here, n_i is the number of training samples in class i, and $\beta \in (0,1)$ is a smoothing parameter, typically close to 1 (e.g., 0.999). This down-weights frequent classes and gives higher importance to rare classes in a smooth, non-discrete way.

2) Dice Loss: Dice loss is a commonly used loss function for image segmentation tasks, especially when dealing with highly imbalanced datasets. It is derived from the Dice Similarity Coefficient (DSC), which measures the overlap between two sets—typically, the predicted segmentation and the ground truth.

The Dice coefficient for two binary masks P (prediction) and G (ground truth) is defined as:

$$\text{Dice}(P,G) = \frac{2|P \cap G|}{|P| + |G|} = \frac{2\sum_i p_i g_i}{\sum_i p_i + \sum_i g_i}$$

where:

- $p_i \in [0,1]$ is the predicted probability for pixel i
- $g_i \in \{0,1\}$ is the ground truth label for pixel i

Dice Loss is simply defined as:

$$\text{Dice Loss} = 1 - \text{Dice}(P,G)$$

To improve numerical stability and differentiability, a small constant ϵ is often added to both the numerator and the denominator:

$$\text{Dice Loss} = 1 - \frac{2\sum_i p_i g_i + \epsilon}{\sum_i p_i + \sum_i g_i + \epsilon}$$

This loss function is particularly effective in scenarios where the foreground class (e.g., a tumor in medical images) occupies a very small portion of the image compared to the background. By directly optimizing for overlap, Dice Loss helps the model focus on correctly segmenting sparse target regions.

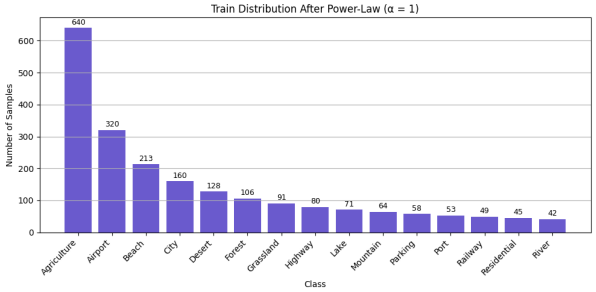
IV. Experimental Results

A. Experimental Setup

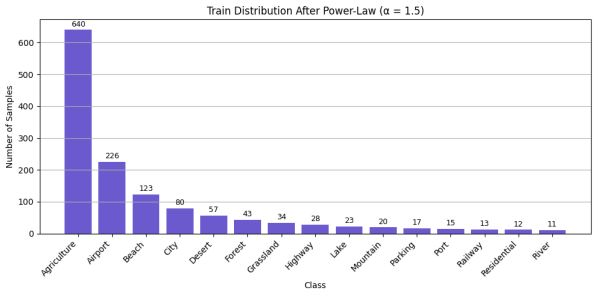
1) Dataset: We used an 80/20 train-test split for all experiment, where 80% of the dataset was used for training and the remaining 20% was held out for evaluation.

For imbalanced dataset simulation, we applied the sampling only on the training set after splitting, keeping the test set unchanged to unsure unbiased evaluation.

2) Computational Environment: Feature extraction and traditional machine learning tasks were executed on a CPU environment, while deep learning experiments were accelerated using an NVIDIA A100 GPU to ensure efficient training and evaluation.



(a) Moderate ($\alpha = 1$)



(b) Severe ($\alpha = 1.5$)

Fig. 2: Testing Data Distribution on Different Imbalance Levels

3) Evaluation Metrics: To evaluate performance, we used the following metrics:

- **Confusion Matrix** – to visualize performance and misclassifications.
- **Precision** – the proportion of true positive predictions among all positive predictions

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- **Recall** – the proportion of true positives correctly identified.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **F1-score** – harmonic mean of precision and recall, which summarizes the balance between precision and recall in a single number

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

These metrics offer a comprehensive performance evaluation beyond simple accuracy, especially in multi-class classification scenarios.

B. Traditional Machine Learning on Balanced Dataset

1) SIFT based: The results presented in Table I demonstrate the comparative performance of several traditional machine learning classifiers applied to SIFT-based image features. The linear SVM achieves the best overall performance, with an accuracy of 64.95% and a F1-score of 65.16%, which suggests that the class boundaries in SIFT feature space are mostly linearly separable. Random Forest performs slightly worse

than SVM but remains competitive, with only minor reductions in both metric, around 1% in accuracy and less than 3% in F1-score. The kNN is the least effective with averaged SIFT descriptors.

TABLE I: Overall Performance of SIFT with Different Classifiers

Classifier	Accuracy	F1-score
SVM (Linear)	**0.6495**	**0.6516**
Random Forest	0.6361	0.6225
kNN	0.5736	0.5671

[a]F1-score unweighted average of per-class F1-scores
[b]Test size for each class is slightly different, but all around 160

Table II indicates across all SIFT based classifiers, performance on the lake and river categories is notably weak, where both precision and recall metrics consistently fall below 50%. This suggests that SIFT descriptors struggle to capture meaningful information in such scenes, likely due to the lack of distinctive gradients and textures on flat water surfaces. On the other hand, the models achieve high scores on forest, parking and port, since these classes contain sharp edges and repeatable patterns, providing ideal conditions for SIFT to generate robust descriptors.

TABLE II: SIFT Based Per-Class Performance

Class	SVM			Random Forest			kNN		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Agriculture	0.6120	0.7000	0.6531	0.6500	0.6500	0.6500	0.5802	0.5875	0.5839
Airport	**0.5079**	**0.6062**	**0.5527**	**0.4899**	**0.4562**	**0.4725**	**0.3755**	**0.5563**	**0.4484**
Beach	0.6296	0.6375	0.6335	**0.5767**	0.5875	0.5820	0.5940	0.4938	0.5392
City	0.6810	0.6937	0.6873	0.6313	0.7063	0.6667	0.5128	0.7500	0.6091
Desert	0.6689	0.6556	0.6622	0.6866	0.6093	0.6456	0.6667	0.5033	0.5736
Forest	0.8303	0.8562	0.8431	0.7833	0.8812	0.8294	0.7833	0.8812	0.8294
Grassland	**0.5592**	**0.5484**	**0.5537**	0.6891	**0.5290**	**0.5985**	0.6979	**0.4323**	**0.5339**
Highway	**0.5706**	**0.5813**	**0.5759**	**0.4842**	**0.5750**	**0.5257**	**0.4703**	0.5437	**0.5043**
Lake	**0.3663**	**0.3962**	**0.3807**	**0.4141**	**0.3333**	**0.3693**	**0.3784**	**0.2642**	**0.3111**
Mountain	0.8146	0.7688	0.7910	0.6649	0.7688	0.7130	0.6646	0.6813	0.6728
Parking	0.9020	0.8625	0.8818	0.8373	0.8688	0.8528	0.8511	0.7500	0.7973
Port	0.8417	0.7312	0.7826	0.6871	0.6312	0.6580	0.7143	**0.4375**	0.5426
Railway	0.6755	0.6375	0.6559	0.6242	0.6438	0.6338	**0.5127**	0.6312	0.5658
Residential	0.8043	0.6937	0.7450	0.7135	0.8562	0.7784	0.6308	0.8438	0.7219
River	**0.3831**	**0.3688**	**0.3758**	**0.4062**	**0.3250**	**0.3611**	**0.3193**	**0.2375**	**0.2724**

[a]Bolded values indicate the five lowest-performing scores in each metric column.

The confusion matrix in Fig. 3 and visualization of SIFT keypoints in Fig. 4 further illustrate the challenges faced by SIFT based model in correctly identifying lake and river images.

Fig. 3 shows that lake is frequently predicted as beach, grassland or river, while river is most commonly misclassified as lake, agriculture or beach. These misclassifications often occur among classes characterized by large homogeneous regions, typically flat or water-dominant areas. Despite such visual similarity, classification performance across these varies. For instance, agriculture and beach are classified more accurately, likely due to their more structured or well-defined boundaries that facilitate keypoints detection by SIFT. Interesting, grassland also achieves relatively better performance despite lacking a sharp edge, suggesting its distinctive nature in the dataset.

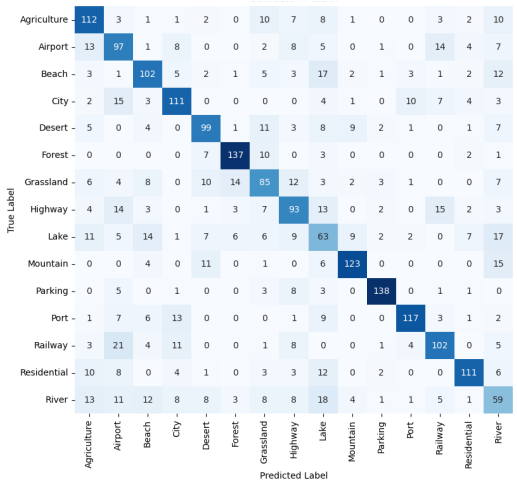
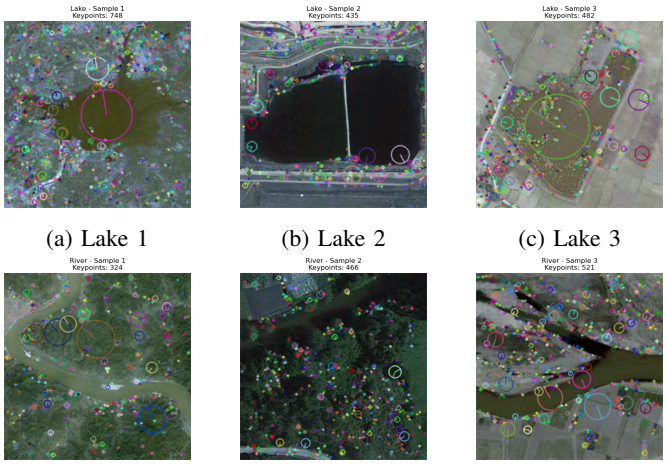


Fig. 3: Confusion matrix of SIFT + SVM

Fig. 4 reveals that SIFT struggles to extract meaningful keypoints from lake and river images due to the presence of diffuse, unstructured boundaries and random internal textures within water bodies. The detected keypoints lack coherent alignment with identifiable edge structures, which reflects the limitations of SIFT based classification in handling such scenes.

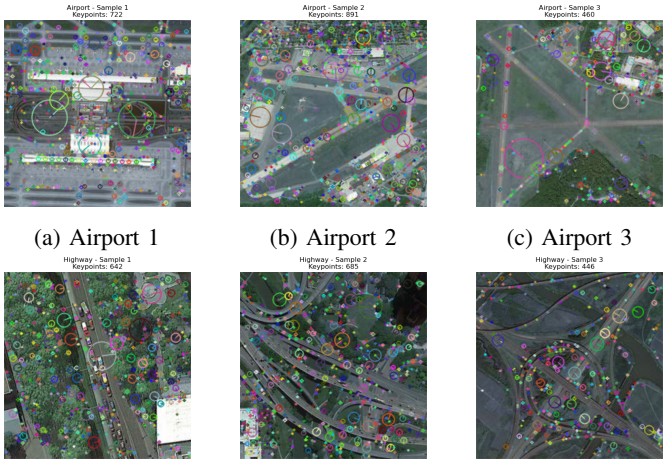


(a) Lake 1 (b) Lake 2 (c) Lake 3

(d) River 1 (e) River 2 (f) River 3

Fig. 4: SIFT keypoints on Lake and River sample images

As shown in Table II, the classification performance for the airport and highways categories remains moderate. The airport class has a precision of roughly 51%, meaning that only about half of the images labeled as airport were correct, and a recall of around 61%, indicating that 61% of all actual airport images were successfully detected by the model. For highway, it achieves a slightly higher precision of 57% but a lower recall of 58%, which also indicates a substantial number of misclassifications. The confusion matrix in Fig. 3 provides further evidence of these classification challenges, showing that airport images are frequently misclassified as agriculture



(a) Airport 1 (b) Airport 2 (c) Airport 3

(d) Highway 1 (e) Highway 2 (f) Highway 3

Fig. 5: SIFT keypoints on Airport and Highway sample images

and railway, while highway images are often confused with airport, lake and railway. This misclassification pattern can be further understood through the keypoint visualizations in Fig. 5. SIFT is fundamentally designed to detect distinctive local features rather than capturing global structural cues. As a result, it fails to effectively represent the long, parallel or intersection lines. Such limitations constrain its performance in differentiate between visually similar, line-based patterns.

SIFT-based representations demonstrate strong performance on texture-rich categories such as forest and parking, yet struggle with scenes dominated by smooth surfaces or repetitive linear patterns, such as lake, river and airport. This is largely due to SIFT's inherent constraint in modeling global patterns and extended structural elements.

2) Local Binary Pattern (LBP) Based: Among the classifier evaluated, Table III shows that Random Forest outperforms SVM (RBF) and kNN, with an accuracy of around 55.75% and a macro F1-score of about 55.43%.

TABLE III: Overall Performance of LBP with Different Classifiers

Classifier	Accuracy	F1-score
SVM (RBF)	0.5054	0.5009
Random Forest	**0.5575**	**0.5543**
kNN	0.5038	0.5019

[a]F1-score unweighted average of per-class F1-scores
[b]Test size for each class is 160

As presented in Table IV, LBP based classification performance slightly varies across different classifiers, but some general trends among classes exist. Forest, grassland and port consistently rank among top-performing categories, while classes such as airport, river, city, beach and highway exhibit persistently poor performance. The latter group typically yields precision, recall and F1-scores under 50%. Even with the strongest classifier, Random Forest, the airport class achieves only 39.01% precision, 34.38% recall and a F1-score of 36.54%. Similarly, river classification performance remains

TABLE IV: LBP Based Per-Class Performance

Class	SVM (RBF)			Random Forest			kNN		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Agriculture	0.4780	0.5437	0.5088	0.5465	0.5875	0.5663	0.4709	0.6062	0.5301
Airport	**0.3424**	**0.3937**	**0.3663**	**0.3901**	**0.3438**	**0.3654**	**0.2684**	**0.3187**	**0.2914**
Beach	0.4906	**0.3250**	**0.3910**	**0.4748**	**0.4125**	**0.4415**	**0.3765**	**0.4000**	**0.3879**
City	**0.3566**	**0.3187**	**0.3366**	**0.4511**	0.5188	0.4826	**0.4022**	0.4500	0.4248
Desert	0.4833	0.5437	0.5118	0.5494	0.5563	0.5528	0.5063	0.5000	0.5031
Forest	0.7077	0.8625	0.7775	0.8182	0.8438	0.8308	0.7791	0.8375	0.8072
Grassland	0.7203	0.6438	0.6799	0.6842	0.7312	0.7069	0.7353	0.6250	0.6757
Highway	0.5089	**0.3563**	**0.4191**	**0.4684**	**0.4625**	**0.4654**	**0.4196**	**0.3750**	**0.3960**
Lake	0.6476	0.4250	0.5132	0.5942	0.5125	0.5503	0.5135	0.4750	0.4935
Mountain	**0.4498**	0.7000	0.5477	0.5789	0.6875	0.6286	0.5253	0.6500	0.5810
Parking	0.4875	0.4875	0.4875	0.5723	0.5938	0.5828	0.5865	0.4875	0.5324
Port	0.6887	0.6500	0.6688	0.6974	0.6625	0.6795	0.6144	0.5875	0.6006
Railway	0.5500	0.4813	0.5133	0.5263	**0.4375**	**0.4778**	0.5143	**0.3375**	**0.4075**
Residential	**0.4354**	0.5687	0.4932	0.5532	0.6500	0.5977	0.5131	0.6125	0.5584
River	**0.3191**	**0.2812**	**0.2990**	**0.4113**	**0.3625**	**0.3854**	**0.3983**	**0.2938**	**0.3381**

[a]Bolded values indicate the five lowest-performing scores in each metric column.

suboptimal, with precision at 41.13%, recall at 36.25% and F1-score at 38.52%.

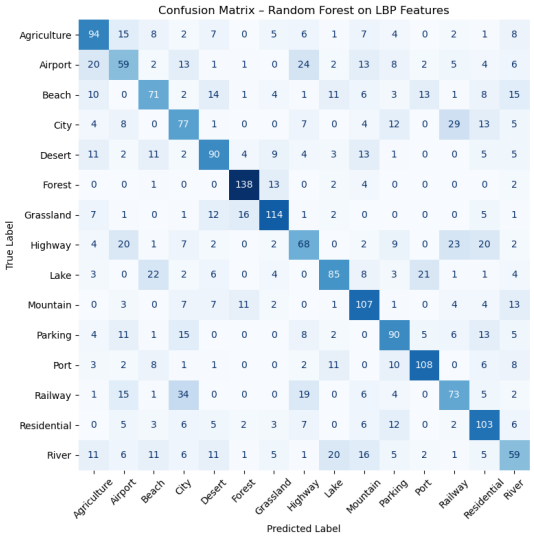
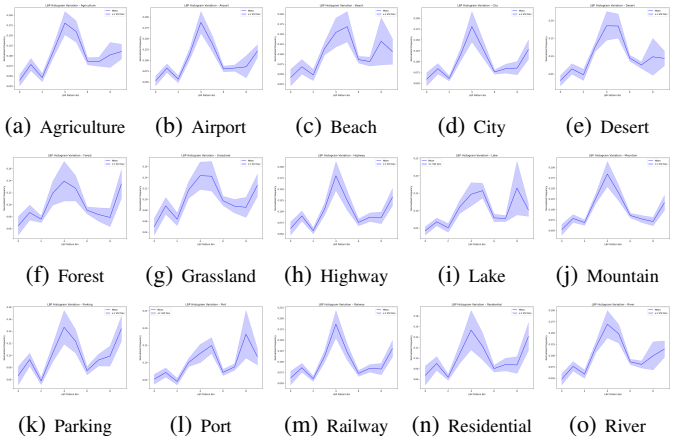


Fig. 6: Confusion matrix of LBP + Random Forest

A deeper understanding of classification behavior of LBP based methods can be obtained by inspecting the confusion matrix in Fig. 6, where Random Forest is selected as a representative model due to its comparatively robust performance and generally consistent performance among all tested classifiers. As shown in the figure, airport images are frequently misclassified as agriculture, highway, city or mountain, while river scenes are commonly labeled as lake, mountain, desert, beach or agriculture.

Fig 7 further reveals that superior performance is generally associated with classes whose histograms are more clearly distinguishable from others. Conversely, misclassifications often arise when either the mean histograms or individual samples of one class resemble those of others. Airport images, for example, are often predicted as agriculture, highway, city or mountain, which reflects the visual similarity illustrated in Fig. 7 (a) (b) (d) (h) (j).



(a) Agriculture (b) Airport (c) Beach (d) City (e) Desert

(f) Forest (g) Grassland (h) Highway (i) Lake (j) Mountain

(k) Parking (l) Port (m) Railway (n) Residential (o) River

Fig. 7: Mean and Std of LBP histograms from each class

However, according to Fig. 6, mountain images are rarely misclassified as airport and desert images are seldom labeled as river. This observation aligns with Fig. 8, which indicates that histogram distributions of river and airport class are closer to those of many other classes, with generally smaller inter-class distances. In comparison, the histograms of mountain and desert classes appear more distinctive, reducing the likelihood of confusion with other categories.

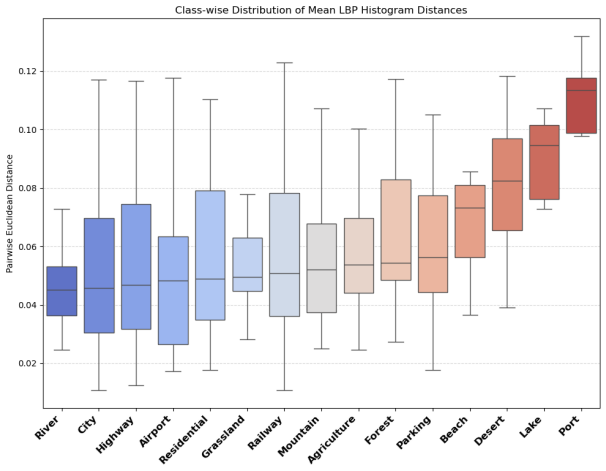


Fig. 8: Mean LBP Histogram Euclidean Distance Distribution

Moreover, certain classes like beach display distinctive mean histograms (Fig. 7) and relatively large distances from other classes (Fig. 8), still suffer from poor performance. This may be attributed to high intra-class variability, as indicated by the wide standard deviation band in Fig. 7 (c). The large variability in LBP patterns among beach samples reduces feature consistency, making it more difficult for the model to generalize effectively.

LBP based models perform well for classes with consistent and distinctive textures, such as forest and port, but struggle with categories like airport, river and beach due to overlapping histogram patterns with other classes or high intra-class variability. While LBP captures local texture features at the pixel level, the histogram representation discards spatial information, making the model lack awareness of local structure, further limiting its performance.

3) Image Preprocessing: As shown in Table V, preprocessing techniques like Gaussian blur and histogram equalization generally boost the performance of SIFT based models, improving both accuracy and F1-score across all classifiers. The most significant improvement is observed with the linear SVM, whose accuracy increases from 64.95% to 69.00%, and F1-score from 65.16% to 68.87%, which confirms linear SVM as the best choice for SIFT features. On LBP based model, preprocessing has mixed or even negative effects. For example, LBP + SVM(RBF) shows a performance drop, with accuracy decreasing from 50.54% to 49.63% and F1-score from 50.09% to 48.99%. Despite these variations, Random Forest consistently outperforms other classifiers for LBP features, retaining the highest accuracy (58.54%) and F1-score (58.36%) after preprocessing.

TABLE V: Comparison of Preprocessing Effects on Accuracy and F1-score for Each Classifier

Feature	Metric	SVM		Random Forest		kNN	
		With	Without	With	Without	With	Without
SIFT-based	Accuracy	0.6900	0.6495	0.6379	0.6285	0.6058	0.5736
	F1-score	0.6887	0.6516	0.6311	0.6225	0.5993	0.5671
LBP-based	Accuracy	0.4963	0.5054	0.5854	0.5575	0.5092	0.5038
	F1-score	0.4899	0.5009	0.5836	0.5543	0.5081	0.5019

[a] F1-score reported is the macro-average over all classes.
[b] **With** = images preprocessed by Gaussian blur and histogram equalization; **Without** = raw input.
[c] SVM uses a linear kernel for SIFT-based and an RBF kernel for LBP-based features.

Preprocessing likely enhances SIFT performance by reducing noise in flat areas and emphasizing edges. This makes SIFT detect fewer irrelevant keypoints and focus on more discriminative features. Preprocessing may help LBP by boosting contrast or removing noise, it can also disrupt local pixel relationships, which reduces feature consistency. Additionally, when using LBP for feature extraction, classifier response differs on preprocessing. Random Forest is more tolerant feature changes, while SVM and kNN are more sensitive to shifts in the feature space.

4) Comparison and Computational Trade-off: Table VI presents a comparison between the SIFT based and LBP based models using their respective best-performing classifiers and with preprocessing applied. When evaluated on the full dataset, the SIFT based model requires significantly more time for both feature extraction and model training than LBP based counterpart. However, it also achieves better performance, reaching 69.00% accuracy and 68.87% F1-score, compared to 58.46% accuracy and 58.28% F1-score for the LBP-based model. To reduce computational cost, we evaluated the SIFT based model on a reduced training set comprising 25% of the original data. The result remain competitive, achieving 61.17% accuracy and 61.09% F1-score, which still outperforms the LBP based model while requireing a similar amount of total computation time.

TABLE VI: Performance and Efficiency of Traditional Models

Model	Split	Acc	F1	Train Time	Total Time
SIFT + SVM	640/160	0.6900	0.6887	40.93s	3m25.5s
SIFT + SVM	160/160	0.6117	0.6109	0.80s	1m7.4s
LBP + RF	640/160	0.5846	0.5828	2.47s	41.2s

C. Deep Learning on Balanced Dataset

The results in Table VII show that both models achieve high levels of accuracy and F1-score, above 97%. Although EfficientNet_b0 marginally outperforms ResNet18 in both metrics by approximately 0.6%, the difference is relatively small. It is worth noting that ResNet18 completes training about 40 seconds faster than EfficientNet_b0, which makes it a better choice when computational resources or time are limited. Given the comparable performance, ResNet18 offers a rational trade-off between accuracy and efficiency.

TABLE VII: General Performance Comparison of Deep Learning Models

Model	Accuracy	F1-score	Training Time
ResNet18	0.9733	0.9733	91.05s
EfficientNet_b0	0.9792	0.9793	130.78s

[a]F1-score refers to macro average F1-score

D. Deep Learning on Imbalanced Dataset

In our experiments on the balanced dataset, both ResNet18 and EfficientNet_b0 demonstrates strong classification performance. To evaluate their robustness under class imbalance, both models were applied to imbalanced datasets exhibiting moderate and severe long-tailed distributions. As shown in Table VIII, under moderate imbalance, the overall accuracy and F1-scores of both models remain closely aligned. However, in the severe imbalance setting, the divergence in tail-class performance becomes more evident. According to Table IX and Table X, EfficientNet_b0 shows a shaper decline in recall for minority classes compared to Resnet18. For example, the recall for the river class, EfficientNet_b0 even reaches 0, while ResNet18 stays around 35%.

Moreover, as seen in Table IX, several majority classes such as agriculture, airport, beach and city show much higher recall than precision. This indicates that EfficientNet_b0 frequently predicts these majority classes even when incorrect, suggesting an excessive bias towards head classes. This reflects the model's sensitivity to class frequency, which can be caused by EfficientNet_b0's deeper architecture and higher capacity. These characteristics make it powerful under balanced conditions, but tend to overfit to dominant patterns in severely imbalanced data, compared to ResNet18. Thus ResNet18 is selected for further method exploration.

The application of focal loss with Effective Number of Samples was motivated by the poor recall for tail classes (e.g., river: 35.00%, railway: 36.25%) in Table X. Table XII shows a general improvement in tail-class recall and average recall across both moderate and severe long-tailed datasets.

TABLE VIII: Performance Comparison of ResNet18 and EfficientNet_b0 on Imbalanced Datasets

Model	Moderate Imbalance		Severe Imbalance	
	Accuracy	**F1 Score**	**Accuracy**	**F1 Score**
ResNet18	0.9125	0.9116	0.7992	0.7900
EfficientNet_b0	0.9096	0.9084	0.6833	0.6274

TABLE IX: Classification Report of EfficientNet_b0

Class	Moderate			Severe		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.8071	0.9938	0.8908	0.5745	0.9875	0.7264
Airport	0.7811	0.9812	0.8698	0.3902	1.0000	0.5614
Beach	0.8989	1.0000	0.9467	0.6426	1.0000	0.7824
City	0.9281	0.9688	0.9480	0.6379	0.9688	0.7692
Desert	0.9551	0.9313	0.9430	0.9304	0.9187	0.9245
Forest	0.7950	0.9938	0.8833	0.7240	1.0000	0.8399
Grassland	0.9792	0.8812	0.9276	0.9301	0.8313	0.8779
Highway	0.9250	0.9250	0.9250	0.7235	0.7688	0.7455
Lake	0.8859	0.8250	0.8544	0.8444	0.2375	0.3707
Mountain	0.8655	0.9250	0.8943	0.7598	0.8500	0.8024
Parking	0.9634	0.9875	0.9753	0.8250	0.8250	0.8250
Port	0.9855	0.8500	0.9128	0.9219	0.3688	0.5268
Railway	0.9512	0.7312	0.8269	0.9545	0.1313	0.2308
Residential	0.9866	0.9187	0.9515	1.0000	0.3812	0.5520
River	0.9320	0.6000	0.7300	0.0000	0.0000	0.0000
Macro Avg	0.9093	0.9008	0.8986	0.7239	0.6846	0.6357

TABLE X: Classification Report of ResNet18 on Imbalance

Class	Moderate			Severe		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.8595	0.9938	0.9217	0.7149	0.9875	0.8294
Airport	0.7248	0.9875	0.8360	0.5048	0.9875	0.6681
Beach	0.8541	0.9875	0.9159	0.7970	0.9812	0.8796
City	0.9212	0.9500	0.9354	0.6667	0.9750	0.7919
Desert	0.9181	0.9812	0.9486	0.9162	0.9563	0.9358
Forest	0.8791	1.0000	0.9357	0.8081	1.0000	0.8939
Grassland	0.9618	0.9437	0.9527	0.9295	0.9062	0.9177
Highway	0.9597	0.8938	0.9256	0.8621	0.7812	0.8197
Lake	0.9521	0.8688	0.9085	0.9076	0.6750	0.7742
Mountain	0.9320	0.8562	0.8925	0.8600	0.8063	0.8323
Parking	1.0000	0.9688	0.9841	0.9643	0.8438	0.9000
Port	0.9739	0.9313	0.9521	0.9147	0.7375	0.8166
Railway	0.9516	0.7375	0.8310	0.8923	0.3625	0.5156
Residential	0.9865	0.9125	0.9481	0.9533	0.6375	0.7640
River	0.9391	0.6750	0.7855	0.9492	0.3500	0.5114
Avg	0.9209	0.9125	0.9116	0.8427	0.7992	0.7900

TABLE XI: Performance Comparison for ResNet18 Variants under Long-tailed Imbalance

Method	Moderate Imbalance		Severe Imbalance	
	Accuracy	F1-score	Accuracy	F1-score
ResNet18	0.9125	0.9116	0.7992	0.7900
ResNet18 + Focal	0.9204	0.9199	0.8167	0.8105
ResNet18 + Focal + Dice	0.9254	0.9245	0.8538	0.8482
ResNet18 + Focal + Dice + Aug	0.9296	0.9288	0.8575	0.8511
ResNet18 + Focal + Dice + Aug + Sampler	0.9346	0.9337	0.8625	0.8581

TABLE XII: Classification Report of ResNet18 + Focal on Imbalance

Class	Moderate			Severe		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.8579	0.9812	0.9155	0.6598	0.9938	0.7930
Airport	0.7500	0.9187	0.8258	0.5487	0.9500	0.6957
Beach	0.9325	0.9500	0.9412	0.7585	0.9812	0.8556
City	0.8914	0.9750	0.9313	0.8011	0.9313	0.8613
Desert	0.9679	0.9437	0.9557	0.9613	0.9313	0.9460
Forest	0.9634	0.9875	0.9753	0.8404	0.9875	0.9080
Grassland	0.9750	0.9750	0.9750	0.8538	0.9125	0.8822
Highway	0.9481	0.9125	0.9299	0.9103	0.8250	0.8656
Lake	0.9045	0.8875	0.8959	0.9615	0.7812	0.8621
Mountain	0.8982	0.9375	0.9174	0.8696	0.8750	0.8723
Parking	0.9873	0.9750	0.9811	0.9773	0.8063	0.8836
Port	0.9934	0.9437	0.9679	0.9250	0.6937	0.7929
Railway	0.9339	0.7063	0.8043	0.7686	0.5813	0.6619
Residential	0.9869	0.9437	0.9649	0.9732	0.6813	0.8015
River	0.8723	0.7688	0.8173	0.9444	0.3187	0.4766
Avg	0.9242	0.9204	0.9199	0.8502	0.8167	0.8105

Focal loss emphasizes hard or low-confidence predictions, which may not be minority class in some cases. Focal + dice loss was applied to address both prediction prediction difficulty and class imbalance simultaneously.

As shown in Table XII and Table XIII, the application of focal + dice loss yields limited gain under moderate imbalance. Conversely, its impact is more substantial in the severe imbalance setting. The recall for river increase from 31.87% to 41.25%, residential from 68.13% to 81.25%, and the average recall improves from 81.67% to 85.38%. Furthermore, Table VIII shows that classification accuracy improve slightly under moderate imbalance (about 0.5%), while under severe imbalance, accuracy rises significantly, reaching 85.38%. These results suggest that combining focal and dice loss enhance the model's ability to capture both difficult and rare class patterns, which offers more balanced and robust performance in long-tailed distributions.

TABLE XIII: Classification Report of ResNet18 + Focal + Dice on Imbalance

Class	**Moderate**			**Severe**		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.8205	1.0000	0.9014	0.7488	0.9875	0.8518
Airport	0.7806	0.9563	0.8596	0.6581	0.9625	0.7817
Beach	0.8785	0.9938	0.9326	0.8516	0.9688	0.9064
City	0.9560	0.9500	0.9530	0.7696	0.9812	0.8626
Desert	0.9684	0.9563	0.9623	0.9255	0.9313	0.9283
Forest	0.8939	1.0000	0.9440	0.8478	0.9750	0.9070
Grassland	0.9618	0.9437	0.9527	0.9057	0.9000	0.9028
Highway	0.9864	0.9062	0.9446	0.9320	0.8562	0.8925
Lake	0.9648	0.8562	0.9073	0.8681	0.7812	0.8224
Mountain	0.9231	0.9000	0.9114	0.8805	0.8750	0.8777
Parking	1.0000	0.9875	0.9937	0.9427	0.9250	0.9338
Port	1.0000	0.9625	0.9809	0.9562	0.8187	0.8822
Railway	0.9379	0.8500	0.8918	0.8919	0.6188	0.7306
Residential	0.9503	0.9563	0.9533	0.9701	0.8125	0.8844
River	0.9464	0.6625	0.7794	0.8684	0.4125	0.5593
Avg	0.9312	0.9254	0.9245	0.8678	0.8538	0.8482

Although focal + dice loss effectively improves recall for minority class and hard examples, performance remains constrained by the limited diversity and quantity of tail-class samples. To address this, random resized cropping and random horizontal flipping were applied to the training data, introducing moderate visual variability to support generalization. As shown in Table XIII and Table XIV, this augmentation slightly improves in both class-wise and average performance metrics. To maintain interpretability and excessive distortion, more aggressive transformations such as random rotation or heavy noise were intentionally omitted, which ensures that meaningful image features are retained while adding diversity.

While focal + dice loss and moderate data augmentation improve model sensitivity to minority classes, these methods alone may not sufficiently correct the imbalance in sample frequency during training. To further address this, a weighted sampling strategy was introduced to ensure more balanced exposure to rare classes during mini-batch construction. According to Table XIV and Table XV, this approach boost ResNet18 further. Table XI indicates that, compared with baseline ResNet18, under severe imbalance, average recall increases from 79.92% to 86.25%, and moderate recall improves from 91.25% to 93.46%. Tail-class performance shows consistent improvement, and overall metrics also benefit. The accuracy and F1-score increase by over 2% in the moderate setting, and more than 6% under severe imbalance.

TABLE XIV: Classification Report of ResNet18 + Focal + Dice + Augmentation on Imbalance

Class	**Moderate**			**Severe**		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.8833	0.9938	0.9353	0.8093	0.9812	0.8870
Airport	0.8351	0.9812	0.9023	0.6878	0.9500	0.7979
Beach	0.8674	0.9812	0.9208	0.8734	0.8625	0.8679
City	0.9448	0.9625	0.9536	0.8031	0.9688	0.8782
Desert	0.9226	0.9688	0.9451	0.8708	0.9688	0.9172
Forest	0.9627	0.9688	0.9657	0.8525	0.9750	0.9096
Grassland	0.9345	0.9812	0.9573	0.9264	0.9437	0.9350
Highway	0.9167	0.8938	0.9051	0.8957	0.9125	0.9040
Lake	0.9728	0.8938	0.9316	0.9431	0.7250	0.8198
Mountain	0.9459	0.8750	0.9091	0.8333	0.8125	0.8228
Parking	0.9936	0.9750	0.9842	0.9799	0.9125	0.9450
Port	0.9935	0.9500	0.9712	0.7745	0.9875	0.8681
Railway	0.9034	0.8187	0.8590	0.9604	0.6062	0.7433
Residential	0.9623	0.9563	0.9592	0.9565	0.8250	0.8859
River	0.9444	0.7438	0.8322	0.9079	0.4313	0.5847
Avg	0.9322	0.9296	0.9288	0.8716	0.8575	0.8511

TABLE XV: Classification Report of ResNet18 + Focal + Dice + Aug + Sampler on Imbalance

Class	**Moderate**			**Severe**		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Agriculture	0.9064	0.9688	0.9366	0.7947	0.9437	0.8629
Airport	0.8220	0.9812	0.8946	0.6596	0.9688	0.7848
Beach	0.8833	0.9938	0.9353	0.7889	0.9812	0.8747
City	0.9554	0.9375	0.9464	0.7336	0.9812	0.8396
Desert	0.9390	0.9625	0.9506	0.9212	0.9500	0.9354
Forest	0.9349	0.9875	0.9605	0.8238	0.9938	0.9008
Grassland	0.9075	0.9812	0.9429	0.8970	0.9250	0.9108
Highway	0.9735	0.9187	0.9453	0.9182	0.9125	0.9154
Lake	0.9613	0.9313	0.9460	0.9496	0.8250	0.8829
Mountain	0.9231	0.9000	0.9114	0.8947	0.8500	0.8718
Parking	0.9634	0.9875	0.9753	0.9800	0.9187	0.9484
Port	1.0000	0.9688	0.9841	0.9735	0.9187	0.9453
Railway	0.9329	0.8688	0.8997	0.9684	0.5750	0.7216
Residential	0.9867	0.9250	0.9548	0.9826	0.7063	0.8218
River	0.9826	0.7063	0.8218	1.0000	0.4875	0.6555
Avg	0.9381	0.9346	0.9337	0.8857	0.8625	0.8581

V. Discussion

Our comparison began with traditional methods, where SIFT based methods clearly outperform LBP based ones, requiring fewer samples and achieving higher accuracy. Applying preprocessing, such as Gaussian filtering and sharpening, on SIFT based method further improved performance, since SIFT is sensitive to high-frequency regions, where textured or noisy areas may generate excessive or unstable keypoints.

SIFT + preprocessing + SVM pipeline provided a lightweight and effective baseline. However, traditional methods struggled on classes with low texture or indistinct boundaries, suggesting a shift toward deep learning approaches.

On the balanced dataset, deep learning models (ResNet18 and EfficientNet_b0) demonstrated superior performance, which confirms their advantages in learning visual patterns. Under moderate long-tailed imbalanced setting, both methods show an acceptable performance. However, under severe imbalanced conditions, especially on tail classes, models show a sharp drop in minority-class recall, especially for EfficientNet.

To handle this, we applied focal loss with class reweighting using effective number of samples method, which improved recall on underrepresented classes. Combining focal with dice loss further helped by optimizing both classification

confidence and overlap, particularly in the severe imbalance setting. Data augmentation (cropping, flipping) introduced visual diversity, but it makes performance unstable due to it randomness. When it is combined with focal + dice loss and a weighted sampler, ResNet18 achieved significant gains, minority-class recall improved by over 6% and overall accuracy and F1-score by over 2% under severe imbalance.

VI. Conclusion

In this study we have presented a comparative study of traditional machine learning and deep learning methods for aerial scene classification using a balanced and artificially imbalanced version of the skyview dataset. We implemented various methods ranging from handcrafted feature extraction (SIFT, LBP) with classical classifier (SVM, kNN, Random Forest) to deep convolutional models (ResNet18, EfficientNet_b0) under different data distributions.

We explored preprocessing on traditional machine learning methods and imbalance handling strategies on ResNet18, such as reweighting, focal loss, dice loss, augmentation and weighted sampling. Our focus was not only on achieving high accuracy, but also on understanding the strength and failure modes of each method across class types and imbalance scenarios.

However, our study has limitations. Most models used default hyperparameters, and we did not perform fine-tuning optimization for each due to limitation of time. For traditional methods, we expected to explore more solutions on ensemble learning and feature fusion techniques. Another continuation of work could be to look at Explainable AI with Grad-CAM, which may offer additional insight into model decision boundaries and failure cases. That would give us a great chance to improve our model to the next stage.

In conclusion, this research gives a benchmark on the different trade-offs when finding an appropriate method for a certain computer vision task. For the Aerial Scene Classification task, the deep learning methods are better suited, if performance is prioritized. The choice of method should be based off the requirements of the application, computational resources that are available, and the distribution characteristics of the dataset.

References

[1] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004.

[2] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," Pattern Recognit., vol. 29, no. 1, pp. 51–59, Jan. 1996.

[3] C. Cortes and V. Vapnik, "Support-vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, Sep. 1995.

[4] M. Bosch, S. Lacoste-Julien, and X. Zabulis, "Image classification using SIFT features and SVMs in a stacked generalization framework," in Proc. IEEE Int. Conf. Comput. Vis., 2007.

[5] T. Cover and P. Hart, "Nearest neighbor pattern classification," IEEE Trans. Inf. Theory, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016.

[7] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019.

[8] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017.

[9] X. Zhou, X. Zhang, H. Chen, P. Li, S. Bai, A. Yuille, and L. Zhang, "A Meta Survey of Long-Tailed Visual Recognition," *arXiv preprint arXiv:2301.10509*, 2023.

[10] G. Van Horn and P. Perona, "The Devil is in the Tails: Fine-grained Classification in the Long Tail," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2017.

[11] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and C. C. Loy, "Learning from Long-Tailed Data: A Survey," *arXiv preprint arXiv:1903.11240*, 2019.

[12] Cui, Yin, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. "Class-balanced loss based on effective number of samples." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 9268-9277. 2019.

[13] Breiman, Leo. "Random forests." *Machine learning* 45, no. 1 (2001)

[14] Pal, Mahesh. "Random forest classifier for remote sensing classification." *International journal of remote sensing* 26, no. 1 (2005): 217-222.