# Remote Sensing Scene Classification: A Comparison between Traditional Methods and Deep Learning Approaches

Yizhu Zhou (z5602082), Boyang He(z5575322), Kaiwen Cai(z5538440), Hao Zhang (z5640402), Juntao Zhang (z5597423)
University of New South Wales, Course Code: COMP9517

## I. Introduction

With the rapid development of modern technology and satellite systems, remote sensing images have been widely used in various fields, including environmental monitoring, land use analysis, and urban planning. Applying deep learning and computer vision techniques to classify aerial scenes have become increasingly effective and widely adopted.

This project aims to develop and compare different computer vision approaches for classifying aerial scenes from remote sensing images. By applying both traditional feature-based models and modern classification models, i.e., vgg16, ResNet and EfficientNet.

The dataset contains a total of 15 landscape categories, each containing about 800 images, and is a well-balanced classification task. Class balance ensures that the model will not be affected by bias during training, which is helpful to fair evaluation of all kinds of performance. All images in the dataset have a uniform resolution of (256 × 256) pixels, which simplifies preprocessing and model input configuration.

## II. Literature Review

Traditional machine learning methods played a crucial role in the early development of image classification systems, especially before the rise of deep learning. Two of the most representative and important technology paths include the use of color histograms in combination with k-NN classifiers and SIFT features in combination with Bag of Words models (BoVW) and SVM classifiers.

Swain and Ballard (1991) show that color histograms can effectively capture the global color distribution of images, so as to achieve robust image retrieval based on K-nearest neighbor (k-NN) classifier [1]. Subsequently, Csurka et al. (2004) proposed a word bag model method based on SIFT features, quantifying local features into "visual words" and then classifying them by support vector machine (SVM), which has become one of the important baselines in traditional scene classification [2]. Inspired by these early successful approaches, we designed and tested the following four combinations of traditional approaches: KNN+LBP, KNN+SIFT, SVM+LBP, and SVM+SIFT.

For deep learning methods, we choose to use ResNet and DenseNet as the typical CNN models. Because they can automatically learn the image representation from the images in an end-to-end manner, and have a good performance in image recognition [7]. Besides, we also choose YOLOv5, as it can achieve efficient multi-target positioning and classification in complex scenarios, and is widely used in practical applications [6].

By comparing traditional methods, modern CNN-based architectures, and other deep learning methods, we aim to analyze their strengths, limitations, and suitability in classifying diverse landscape types from remote sensing data.

## III. Methods

### *A. Traditional Machine Learning Methods:*

In this project, we adopted a combination of four traditional machine learning methods to conduct multi-category remote image classification experiments, namely: KNN+LBP, KNN+SIFT, SVM+LBP and SVM+SIFT. K-nearest Neighbor (KNN) and Support Vector Machine (SVM) are two traditional machine learning classifiers. We use them respectively in combination with two image Feature extraction methods - Local Binary Pattern (LBP) and Scale-Invariant Feature Transform (SIFT). So as to systematically compare the classification capabilities of different combinations of features and classifiers for remote sensing images.

Traditional machine learning methods still behave well in tasks where the scale of training data is small and have obvious features. In this dataset, there are a total of 800 images of each label, which is a relatively small sample set. Compared with decision trees that are easy to overfitting under high-dimensional sparse features and random forests which performs poorly when the feature structure is not that strong, KNN is more suitable for image categories with clear structure and regular features (such as mountains, cities, beaches and ports), while SVM is good at handling high-dimensional feature spaces, especially in the case of linearly

inseparable data, it can achieve relatively better performance through kernel functions.

To extract discriminative image features, we uniformly preprocess the images into 128×128 grayscale images and respectively use two classic methods for feature extraction: LBP (P=8, R=1, uniform mode) which is used to obtain texture information, and a normalized histogram is generated after extraction; SIFT extracts the 128-dimensional descriptors of the key points of the image and forms a fixed-dimensional feature vector by calculating the mean value.

Then these features are given into the KNN or SVM classifiers. We use a method combining grid search and cross-validation for model hyperparameter tuning: For KNN, we test multiple neighbor numbers (k = 3, 5, 7, 9, 11) and select the best-performing parameter based on the accuracy on the test set., and SVM searches the best performance among the performance of the combination of $C \in [0.1,1,10]$ and kernel functions (linear and rbf).

To further analyze the model performance, we designed a misclassification pair statistical function to identify the most common confusion categories in the test set and conduct error analysis accordingly. And two models with the best performance were selected from the four combinations of KNN+LBP, KNN+SIFT, SVM+LBP and SVM+SIFT as representatives for later comparison work.

### *B. ResNet18*

In this project, we also use ResNet18 as a pre-trained model and do the Transfer training. ResNet18 is a deep residual network. Therefore, it can avoid the vanishing gradients in deep architectures by applying the skip connections mechanism. ResNet18 consists of multiple residual blocks, each of them has a convolutional layer, and a batch normalization layer. It also uses ReLU as the activation function. ResNet directly adds the input to the output layer through identity mapping, which allows it to have the end-to-end capability to directly classify the input images.

During the feature extraction, ResNet18 has four stages in the middle layer, each of which contains several convolutional layers and residual connections. These layers can efficiently extract low-level features (such as borders and corners), middle-level features(such as textures and shapes), high-level features(such as the remote sensing image patterns). The residual structure helps to obtain the extracted information in different layers of the network, allowing the deep layers to learn more complex feature expressions.

In the output layer, ResNet18 applies a global average pooling to reduce the dimension of the feature map. Then, connect to a fully connected layer and output the classification result. Since there are 15 categories of remote sensing images, 15 neural cells are set, each of them corresponds to one category. Finally, the softmax function is used to do the normalization and obtain the probability distribution of each category. The category with the highest probability is selected as the final classification result of the remote sensing image.

During the training process, backpropagation is used to update the parameters, allowing the cross-entropy loss function to be minimized. After 10 iterative training processes, the model can automatically extract discriminative features from the input remote sensing images.

### *C. DenseNet121*

The biggest difference between DenseNet121 and other mainstream convolutional neural networks (such as ResNet) is its "Dense Connectivity" structure. It passes the feature maps of all previous layers to subsequent layers through dense connections. This feature effectively handles the gradient disappearance problem and improves the feature reuse rate. This model performs well in picture classification tasks, especially in remote sensing image scenes with medium or small data volumes, and has strong robustness and high efficiency. Due to the strong connections between layers, DenseNet can achieve deeper modelling capabilities with fewer parameters.

The main network of DenseNet121 can be divided into four Dense blocks, each of which is composed of multiple Bottleneck layers (1×1 and 3×3 convolution). It also contains a transition layer, which controls the feature map size and the number of channels. This structure gives the middle layer the following properties:

- The shallow layer can extract edge, texture, and other low-level features, such as the farmland boundary, water outline;
- The middle layer can extract image structure and local shape features, such as city blocks and building density;
- The Deep layer is used to extract highly semantic information, such as judging the pattern characteristics of "port", and "airport" scenarios.

Besides, for the output layer of DenseNet121, the feature map of 7×7×1024 is compressed to 1×1024 by GAP. Then the GAP output is connected to a linear classifier of size 15 during the fully connected layer. The SoftMax normalization is automatically performed when calculating CrossEntropyLoss, which is used to obtain the probability for each class. Overall, compared with the traditional SVM/KNN method, this model has more advantages in feature extraction, model generalization, and interpretation.

### *D. Yolov5*

Yolov5 is also an ideal choice for this project. YOLOv5 (You Only Look Once version 5) is a deep learning-based object detection model that is widely known for its speed, accuracy, and efficiency. It performs object detection in a single forward pass, making it suitable for real-time applications. The architecture of YOLOv5 is modular and consists of four major components: the backbone, the neck, the head, and the output layer. Each of these components plays a specific role in the overall detection pipeline, contributing to the model's performance and flexibility.

The backbone of YOLOv5 is responsible for extracting essential visual features from the input image. In YOLOv5, the backbone is based on a variation of CSPDarknet, which incorporates Cross Stage Partial (CSP) connections into the traditional Darknet architecture. This design allows for better gradient flow and feature reuse while reducing computational complexity. As the image passes through the backbone, it is progressively downsampled using convolutional layers, and high-level abstract features are captured that represent different parts and characteristics of objects in the image.

Following the backbone is the neck, which functions as a feature aggregator. Its main purpose is to enhance and fuse features extracted at different levels of the backbone so that the model can detect objects of varying sizes more effectively. YOLOv5 employs both a Feature Pyramid Network (FPN) and a Path Aggregation Network (PANet) in the neck. The FPN facilitates top-down feature fusion, which helps to maintain semantic richness in higher-resolution layers, while PANet introduces a bottom-up path to strengthen localization signals and improve information flow. Together, these components enable the model to combine low-level spatial details with high-level semantic information.

The next component is the head, which is responsible for generating the actual predictions. YOLOv5 predicts bounding boxes, objectness scores, and class probabilities at three different scales. This multi-scale detection approach allows the model to effectively handle small, medium, and large objects within the same image. For each grid cell in the output feature maps, the head predicts a fixed number of bounding boxes along with associated confidence scores and class labels. These predictions are made using anchor boxes, which serve as reference shapes for object localization.

Finally, the output stage involves a post-processing step that filters and refines the raw predictions. The most important technique used at this stage is Non-Maximum Suppression (NMS), which removes redundant or overlapping bounding boxes by retaining only those with the highest confidence scores. The final output consists of a list of detected objects, each characterized by a bounding box, a class label, and a confidence score indicating how certain the model is about the detection. All these will be shown in the result analysis.

### *E. VGG16*

We also use VGG16 for this project. VGG16 is a very classic convolutional neural network (CNN) architecture and is widely used in computer vision tasks such as image classification. The model has a total of 16 weighted network layers (13 convolutional layers + 3 fully connected layers).

The structure of VGG16 can be divided into the following parts:

1. Input layer: The input size is usually an RGB image of 224×224×3.

2. Convolution Blocks: There are a total of 5 convolutional blocks. Each block contains 2 to 3 3 convolutional layers, followed by a 2×2 Max Pooling layer. The size of all convolution kernels is 3×3, the step size is 1, and padding keeps the output size unchanged. The number of channels gradually increased from 64 to 512.

3. Fully Connected Layers: Two 4096-dimensional fully connected layers. A 1000-dimensional fully connected layer (1000 classes for the ImageNet classification task)

4. Softmax: Used for outputting the final classification probability

The features of VGG16 are:

• Use multiple small convolution kernels instead of large convolution kernels to reduce the number of parameters while retaining the receptive field;

• The network depth is moderate, and it performed very well in the early stage of the rise of deep learning;

• The structure is simple and unified, which is convenient for understanding and modification.

## IV. RESULT

| Model_name | Precision | Recall | F1-score |
|---|---|---|---|
| KNN + LBP | 0.5073 | 0.5075 | 0.5032 |
| SVM + SIFT | 0.6414 | 0.6450 | 0.6396 |
| vgg16 | 0.7817 | 0.7746 | 0.7749 |
| DenseNet | 0.9109 | 0.9083 | 0.9075 |
| YOLOv5 | 0.938 | 0.908 | 0.85 |
| ResNet18 | 0.9714 | 0.9708 | 0.9709 |

Fig. 1. Performance matrix of different classification models

Fig. 1. summarizes the overall precision, recall, and F1-score of all traditional and deep learning models evaluated on the remote sensing image classification task. A detailed analysis and comparison will be discussed in the following sections.

### *A. Traditional machine learning evaluation*

| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| KNN + LBP | 0.5075 | 0.5073 | 0.5075 | 0.5032 |
| SVM + LBP | 0.5408 | 0.5423 | 0.5408 | 0.5388 |
| KNN + SIFT | 0.5017 | 0.4987 | 0.5017 | 0.4886 |
| SVM + SIFT | 0.6450 | 0.6414 | 0.6450 | 0.6396 |

Fig. 2. Performance matrix between four traditional model

Fig.2 evaluated the performance of four model combinations on the test set, including Accuracy, Precision, Recall, and F1-score:

• KNN + LBP
• SVM + LBP
• KNN + SIFT
• SVM + SIFT

Among them, SVM + SIFT has the best performance on all indexes. The reasons for this result can be explained in two ways:

1. **SVM is superior to KNN in generalization ability:**

SVM works better when dealing with high-dimensional and sparse feature Spaces, building clearer decision boundaries by maximizing the spacing between classes. Especially when the categories are unevenly distributed or the image features are complex (for example, there are a lot of similarities or blurred boundaries in the scenes of cities, ports, residential areas, railways, etc.), SVM can distinguish more stably.

2. **SIFT is more suitable for natural scene feature extraction than LBP:**

Unlike LBP, which relies on local texture and grayscale, SIFT can extract local structural features with more scale and rotation invariance, especially for natural scenes with diverse geometric shapes (mountains, forests, rivers, lakes, beaches, etc.). SIFT is better at capturing key points and contours, so it performs better on complex backgrounds.

SVM + SIFT combination shows strong classification and generalization ability in traditional methods, but the overall performance is still inferior to deep learning model.

Then we analyze the behaviour of the model on each label

```
[KNN + LBP] Most confused: 'Highway' misclassified as 'Railway' (19 times)
[SVM + SIFT] Most confused: 'Desert' misclassified as 'Grassland' (27 times)
```

Fig. 3. Misclasssification performance

| Class | SVM+SIFT_P | SVM+SIFT_R | SVM+SIFT_F1 | KNN+LBP_P | KNN+LBP_R | KNN+LBP_F1 |
|---|---|---|---|---|---|---|
| 0 | 0.71 | 0.72 | 0.72 | 0.61 | 0.50 | 0.55 |
| 1 | 0.66 | 0.57 | 0.61 | 0.39 | 0.39 | 0.39 |
| 2 | 0.67 | 0.60 | 0.63 | 0.58 | 0.49 | 0.53 |
| 3 | 0.68 | 0.82 | 0.75 | 0.42 | 0.72 | 0.53 |
| 4 | 0.56 | 0.36 | 0.44 | 0.52 | 0.29 | 0.37 |
| 5 | 0.65 | 0.69 | 0.67 | 0.61 | 0.69 | 0.65 |
| 6 | 0.55 | 0.60 | 0.57 | 0.45 | 0.31 | 0.37 |
| 7 | 0.55 | 0.60 | 0.57 | 0.43 | 0.45 | 0.44 |
| 8 | 0.45 | 0.41 | 0.43 | 0.35 | 0.25 | 0.29 |
| 9 | 0.69 | 0.85 | 0.76 | 0.61 | 0.69 | 0.65 |
| 10 | 0.81 | 0.80 | 0.81 | 0.62 | 0.66 | 0.64 |
| 11 | 0.75 | 0.65 | 0.70 | 0.53 | 0.41 | 0.46 |
| 12 | 0.66 | 0.78 | 0.71 | 0.45 | 0.72 | 0.56 |
| 13 | 0.79 | 0.81 | 0.80 | 0.58 | 0.75 | 0.66 |
| 14 | 0.44 | 0.40 | 0.42 | 0.31 | 0.20 | 0.24 |

Fig. 4. Fig.Result_comparision

Fig.3 shows the most frequent misclassification cases

Fig.4 Per-class performance comparison of KNN + LBP and SVM + SIFT models. The table lists precision, recall, and F1-score for each of the 15 scene categories.

**KNN + LBP**

As shown in the right part of the classification report(Fig. 4) , the performance of this model varies significantly in different categories.

In Grassland (class 6), Forest (class 5), Mountain (class 9), Parking (class 10) and other scenes with clear texture and obvious structure, LBP can effectively extract features, and the model performance is better with F1-score ≥ 0.60.

However, in categories with weak textures, blurred borders or complex backgrounds, such as Airport (class 1), Beach (class 2), Desert (class 4) and River (class 14), the model performance is poor (F1-score ≤ 0.40), and the model would be confused about the feature between the labels.

Additionally, based on the confusion matrix analysis, the most common misclassification in KNN + LBP was Highway (class 7) being frequently misclassified as Railway (class 12) (19 times) which is shown in Fig.3. These two classes share similar visual characteristics like straight rows and man-made railways, which LBP alone cannot sufficiently distinguish.

In general, KNN + LBP is more suitable for categories with obvious texture features, and has weak adaptability to natural scenes and limited generalization ability.

**SVM + SIFT**

When it comes to the SVM + SIFT model,which shows in the left part of the Fig.4, there are still differences in the performance of different categories.

Among them, Agriculture, Bridge, Mountain, Parking, Port, Residential and other categories perform well (F1-score ≥ 0.70). These categories have obvious structural textures or artificial features and SIFT can effectively extract their local key point information.

In natural scenarios such as deserts and rivers, the model performed poorly (F1-score ≤ 0.45). Desert, in particular, was most likely to be misjudged as Grassland, with 27 confusion cases.

According to our analysis, the main reason for this misjudgment is that all images are processed by gray scale before feature extraction, while desert and grassland in gray scale images are often presented as image areas with large area distribution, close brightness, and regular texture, resulting in the spatial distribution of features extracted by SIFT being very similar, thus causing confusion.

In summary, although SVM + SIFT has stronger classification and structural feature extraction ability than KNN + LBP, its performance is still limited by traditional feature extraction methods when dealing with complex scenes, and it is difficult to compete with deep learning methods.

### *B. Vgg16 evaluation*

We also use VGG16 model, a very classic convolutional neural network, which is always used for data classifying.

We can analyze the result from VGG16 model:

Evaluating model on the test set...

Classification Report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Agriculture | 0.8333 | 0.7143 | 0.7692 | 182 |
| Airport | 0.5325 | 0.4633 | 0.4955 | 177 |
| Beach | 0.8603 | 0.7748 | 0.8153 | 151 |
| City | 0.7282 | 0.8353 | 0.7781 | 170 |
| Desert | 0.8506 | 0.9250 | 0.8862 | 160 |
| Forest | 0.9433 | 0.8750 | 0.9078 | 152 |
| Grassland | 0.8994 | 0.9346 | 0.9167 | 153 |
| Highway | 0.5407 | 0.7290 | 0.6209 | 155 |
| Lake | 0.8848 | 0.9182 | 0.9012 | 159 |
| Mountain | 0.6705 | 0.7436 | 0.7052 | 156 |
| Parking | 0.9291 | 0.8851 | 0.9066 | 148 |
| Port | 0.9306 | 0.8874 | 0.9085 | 151 |
| Railway | 0.5833 | 0.5966 | 0.5899 | 176 |
| Residential | 0.7838 | 0.8112 | 0.7973 | 143 |
| River | 0.8240 | 0.6168 | 0.7055 | 167 |
| accuracy | | | 0.7746 | 2400 |
| macro avg | 0.7863 | 0.7807 | 0.7803 | 2400 |
| weighted avg | 0.7817 | 0.7746 | 0.7749 | 2400 |

Overall Test Accuracy: 0.7746

Fig. 5. VGG16 performance

We can see for most categories, VGG16 can effectively extract features, and the model performance is better with F1-score ≥ 0.70. This shows VGG16 model is good at picture classifying project.

However, VGG16 also has some shortcomings. For instance, in Highway(Class 8) and Railway(Class 13), the F1-score is significantly lower than that of other categories(F1-score is approximately 0.60). This may because the visual features of these two types of pictures are similar: similar structural lines and similar environment.

Additionally, we can also focus on confusion matrix of VGG16:

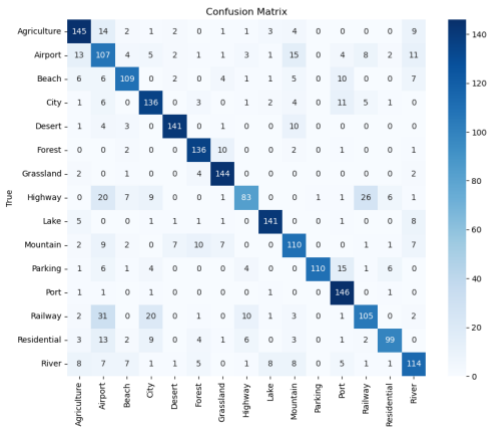


Fig. 6. Confusion matrix of VGG16

Based on the confusion matrix analysis, the most common misclassification in VGG16 is Railway (Class 13), which is misclassified as airport (Class 2) for 31 times shown in Fig.6. These two classes share similar visual characteristics like straight rows and man-made railways, which caused difficulties for correct classification.

Overall, VGG16 has been quite successful in image classification and can basically categorize images correctly.

### *C. DenseNet121 evaluation*

Precision: 0.9109763536543725
Recall: 0.9083333333333333
F1 Score: 0.9075323235439048

Fig. 7. DenseNet121 Overall Evaluation Metrics (Precision, Recall, F1 Score)

According to the macro average index, the macro average accuracy of the model is 0.91, the recall rate is 0.9083, and the F1 score is 0.9075. This set of data comprehensively reflects that DenseNet's judgment of each category is not only accurate, but also stable. High accuracy of macro average indicates that the model has strong classification ability in all categories, especially in the categories with strong structure and clear characteristics, such as City and Port. The value of the recall rate further demonstrates that the model can effectively capture real samples of each type of image, and there is no significant bias.

However, in some natural images, such as Desert and Mountain, although the overall performance is still good, the index is relatively low, and the F1 score fluctuates around 0.80.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Agriculture | 1.00 | 1.00 | 1.00 | 16 |
| Airport | 0.89 | 1.00 | 0.94 | 16 |
| Beach | 0.94 | 0.94 | 0.94 | 16 |
| City | 1.00 | 0.88 | 0.93 | 16 |
| Desert | 0.86 | 0.75 | 0.80 | 16 |
| Forest | 1.00 | 1.00 | 1.00 | 16 |
| Grassland | 0.84 | 1.00 | 0.91 | 16 |
| Highway | 0.88 | 0.94 | 0.91 | 16 |
| Lake | 0.87 | 0.81 | 0.84 | 16 |
| Mountain | 0.76 | 0.81 | 0.79 | 16 |
| Parking | 1.00 | 0.94 | 0.97 | 16 |
| Port | 0.88 | 0.94 | 0.91 | 16 |
| Railway | 0.88 | 0.94 | 0.91 | 16 |
| Residential | 0.94 | 0.94 | 0.94 | 16 |
| River | 0.92 | 0.75 | 0.83 | 16 |
| accuracy | | | 0.91 | 240 |
| macro avg | 0.91 | 0.91 | 0.91 | 240 |
| weighted avg | 0.91 | 0.91 | 0.91 | 240 |

Fig. 8. Classification Report by Category

The reason for this difference may be that there is a high degree of texture similarity between these categories, blurred visual boundaries, and more confusion in different lighting and shooting angles. In addition, water body images (such as rivers and lakes) also show a certain difficulty in classification, indicating that only static feature extraction may

still face the problem of insufficient spatial context information.

In order to better understand the classification behavior of the model, we further draw the confusion matrix of 15 types of scenarios.

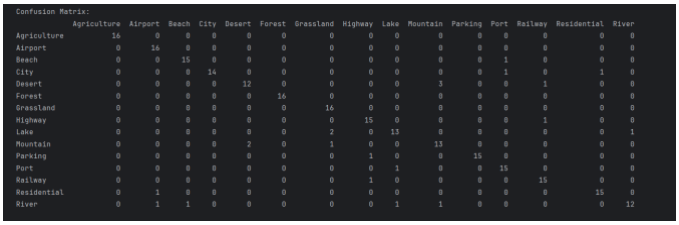


Fig. 9. Confusion Matrix of Model Predictions

Notably, a small number of cross misjudgments occurred in contiguous or semantically close categories such as Desert vs. Mountain and River vs. Lake. These error patterns show that the model has some challenges in processing images with similar textures, blurred edges, or severe scene coexistence. For example, rivers and lakes often appear as continuous, curvilinear water areas in high-altitude images, which leads to overlapping visual features. As a result, the difficulty of classification will increase.

### *D. YOLOv5 evaluation*

First we check the training process in the whole 30 epochs there is a clear drop in the all three loss functions, and a clear improvement in the accuracy.

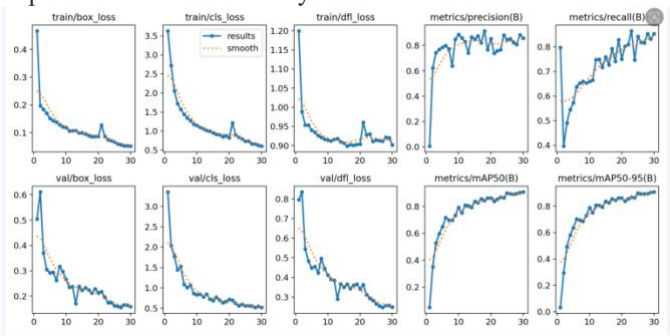


Fig. 10. Loss function to the epochs

Then pick the training batch , boxing and the result randomly..

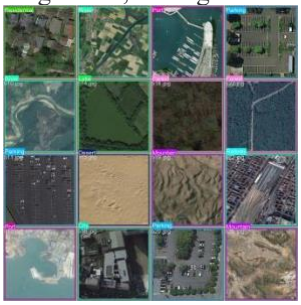

Fig. 11. Training batch


Fig. 12.. River

Fig. 12. River is the most usual case that in the test set there are also 0.938 that are right with over 0.95 confidence,


Fig. 13. Highway

Fig. 13. Highway is the case when the model makes the wrong prediction though it shows a 0.92 confidence, this is clearly a river instead of the highway.

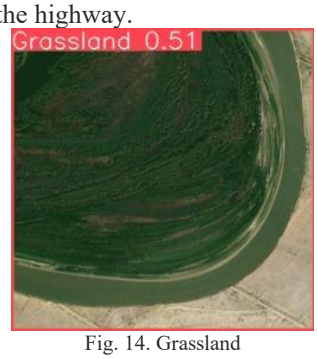


Fig. 14. Grassland

Fig. 14. Grassland is that the model is not confident enough that this is greenland, but actually the prediction is right.

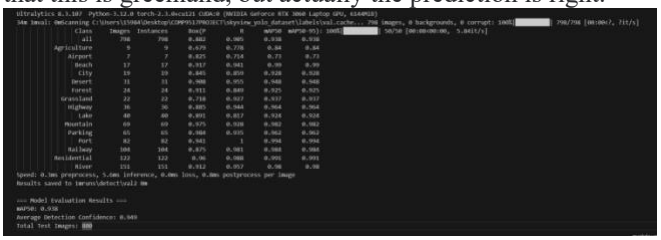

Fig. 15. result metrics

Then we can have an analysis on our final result of the yolo v5s model, The YOLO model demonstrates strong overall performance with an mAP50 of 0.938 and an average detection confidence of 0.949 across 800 test images. The model performs exceptionally well on several classes, particularly Residential areas (0.991 mAP50), Ports (0.994 mAP50), Railways (0.984 mAP50), and Mountains (0.982 mAP50), showing high precision and recall values for these categories. However, there are some areas that need improvement, particularly in the detection of Airports (0.73

mAP50) and Agricultural areas (0.84 mAP50), which show lower precision and recall values. These weaker performing classes also happen to have the smallest number of training instances (7 and 9 respectively), suggesting that the model's performance might be improved by increasing the training data for these specific classes. The model's processing speed is efficient, with a total processing time of 6.7ms per image, distributed across preprocessing (0.3ms), inference (5.6ms), and postprocessing (0.8ms) stages. For more detail we can check the full output in our source data.

In conclusion, YOLOv5 stands out as a powerful and reliable object detection model due to its combination of speed, accuracy, and architectural efficiency.


Fig. 16. result with imbalance sampling

We also made an experiment in the unbalanced sampling situation, but the result remains similar to my last result. This also confirms that Yolov5 can hold the complex unbalanced sampling situation.

### *E. ResNet18 evaluation*

To evaluate the performance of ResNet18, we also calculate the Precision, Recall rate and the F1 score. The result is as follows: Precision：0.9714, Recall：0.9708, F1 Score：0.9708. As we can see, the model has a high level of accuracy and robustness for classifying the remote sensing images.

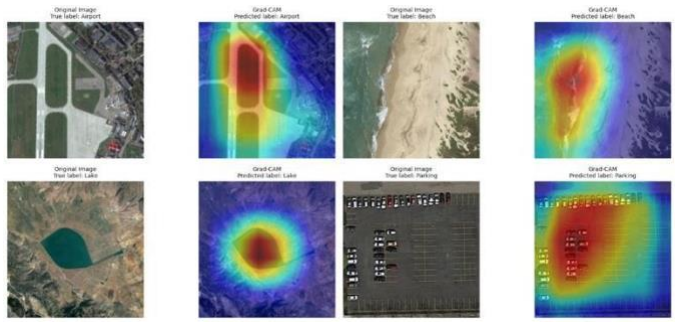

Fig. 17. Grad-CAM of ResNet18

We use Grad-CAM to visualize the model performance. As we can see from Fig. 17. above, Grad-CAM highlights the key regions in the image that the model focuses on when making predictions. ResNet18 can constantly focus on the semantically meaningful areas, such as distinctive land patterns, shapes, and boundaries.

In terms of training strategy, transfer learning with fine-tuning is used by initializing ResNet18 with pre-trained weights on ImageNet, and then training the model on a relatively small amount of labeled remote sensing dataset. It is an efficient way to train the model to do the specific classification problem. The result shows that, even with a limited training dataset, ResNet18 can still learn the visual features of the remote sensing images. Therefore, we can say that the ResNet18 architecture has a good generalizability for remote sensing applications.

## V. DISCUSSION

In our experiment, KNN + LBP is taken as the representative model of traditional machine learning methods and used as the baseline for comparative analysis with other deep learning models to evaluate the performance of each model in the remote sensing image classification task.

It can be seen that the F1-score of KNN + LBP is significantly lower than that of all deep learning models, only 0.5032, which is basically at the level of random guessing. ResNet18 achieved the best result, with an F1-score as high as 0.9709. DenseNet and YOLOv5 followed closely behind, with stable performances.Although the performance of VGG16 is relatively limited (with a precision of 0.7818), it still significantly outperforms traditional methods.

The disadvantages of KNN + LBP are mainly reflected in the following points:

a) Weak feature extraction ability: LBP is only based on local gray-level changes and has difficulty recognizing complex textures and shapes.

b) Confusion between categories: For instance, Highway and Railway have been confused 19 times, indicating that LBP is unable to effectively distinguish similar artificial structures.

c) The classifier has no learning process: KNN is sensitive to noise and cannot extract effective decision boundaries from the training data.

s

**In contrast, deep learning models demonstrate significantly stronger performance**, benefiting from their ability to learn hierarchical feature representations and capture complex spatial semantics directly from raw image data.

1. VGG16 performs well in classifying categories with clear structures and stable textures (such as Desert, Parking, and Port), but its recognition ability is relatively weak in categories with blurred boundaries and overlapping visual features, such as Airport, Highway, and Railway. The Precision is 0.7818. Although it is not as good as other deep models, it is still superior to the traditional methods.
2. Combining detection and classification capabilities, YOLOv5 achieves 0.938 at mAP@50 and performs exceptionally well in Residential, Port, Mountain and other scenarios. Even in the setting of sample imbalance, the model remains stable, demonstrating strong generalization ability and robustness. The reasoning efficiency is high, and the processing time of a single graph is only 6.7ms.
3. DenseNet121 enhances the overall stability through the feature reuse mechanism, achieving an F1-score of

0.9075 and performing evenly in most categories. However, in natural scenes with similar textures such as Desert and Mountain, River and Lake, there are still certain misjudgments, indicating that the model has an improving space in recognizing small and complex texture.

4. ResNet18 is the best-performing model (with an F1-score of 0.9709) among all the model, and its residual structure effectively promotes the extraction of deep semantic features. Combined with the transfer learning strategy, a high accuracy rate can be maintained with limited training data. The Grad-CAM visualization display that model can focus on the key areas, further verifying its discriminative ability and interpretability.

**All of these models share some common features:**

i. **Strong automatic feature learning**: Deep models can automatically extract multi-level semantic information such as texture, edge, and shape from images without relying on manual features in traditional methods (such as SIFT or LBP), and are suitable for remote sensing image classification tasks that contain rich details and various terrains and landforms.

ii. **Good generalization ability**: Even when the training data is limited or there is a class imbalance (such as reducing the Airport class samples for training), the model can still maintain a high accuracy rate and robustness, demonstrating strong cross-class generalization ability.

iii. **Strong adaptability to various scenarios**: Whether it is artificial areas with obvious structures and clear boundaries (such as Port, Residential), or natural areas with similar textures and ambiguous semantics (such as Desert, Mountain, River and Lake), these models all demonstrate stable and reliable recognition performance.

## VI. CONCLUSION

Our project focuses on the classification task of multi-category remote images and evaluates the performance of traditional machine learning based methods and various deep learning models.

Among traditional methods, SVM + SIFT performs best, but it still loses to deep models in complex or texture-similar scenes. When KNN + LBP was used as the baseline model, the F1-score was only 0.5032, which showed obvious limitations.

Among the deep models, ResNet18 performed the best and had excellent feature extraction and generalization capabilities. DenseNet has stable classification, but there is occasional confusion in natural scenes with similar textures. YOLOv5 achieves a good balance between accuracy and speed, and is robust in handling imbalanced samples. Although VGG16 is slightly inferior, it is still superior to traditional methods.

Although deep models perform well as a whole, they still have limitations: they are easily confused in texture-similar classes such as Desert and Grassland; Insufficient adaptability to lighting, perspective changes or occlusion; The accuracy rate is unstable in the few-sample category.

## VII. FUTURE IMPROVEMENT

To further enhance the application value of the model in real remote sensing, we suggest expanding and optimizing it in the following three directions:

1. **Enhance adversarial robustness**: In future training, image samples containing occlusion or noise can be added to improve the model's recognition stability in complex or non-ideal scenarios.
2. **3D scene understanding via multi-view integration**: Remote sensing images are often captured from different angles. Future work can explore 3D reconstruction or multi-view learning to improve spatial awareness and classification in scenario with complex patterns.
3. **Introduce multimodality and basic models**: Conduct joint modeling by combining images and text (such as geographic semantic information), and utilize multimodality or large models (such as CLIP) to enhance recognition and generalization capabilities in open scenes.

## REFERENCES


[1] Y. Zhang et al., “Deep long-tailed learning: A survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(9): 10795-10816, 2023. (https://arxiv.org/abs/2110.04596)

[2] R. Selvaraju et al., “Grad-CAM: Visual explanations from deep networks via gradient based localization”, IEEE International Conference on Computer Vision, pp. 618-626, 2017. (https://arxiv.org/abs/1610.02391)

[3] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in Proc. ECCV Workshop on Statistical Learning in Computer Vision, Prague, Czech Republic, 2004, pp.122.(https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/csurka-eccv-04.pdf)

[4] H. Xu, "LBP texture feature extraction," CSDN Blog, Mar. 31, 2018. [Online]. Available: https://blog.csdn.net/hongbin_xu/article/details/79924961

[5] LYRIQ777, "SIFT (Scale-Invariant Feature Transform)," CSDN Blog, Jul. 9, 2023. [Online].

[6] G. Jocher et al., “YOLOv5: A family of object detection architectures and models pretrained on the COCO dataset,” Ultralytics, 2020. (https://github.com/ultralytics/yolov5)

[7] C. Zhang, P. Benz, D. M. Argaw, S. Lee, J. Kim, F. Rameau, J.-C. Bazin, and I. S. Kweon, "ResNet or DenseNet? Introducing Dense Shortcuts to ResNet," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Waikoloa, HI, USA, 2021, pp. 3545–3554. Available: https://arxiv.org/abs/2010.12496