DATS6103 Data Mining Final Project

Landscape Classification

Jeongwon Yoo

Table of contents

01 Introduction

02 Dataset & EDA

03 Methodology

04 Models & Experiments

05 Results & Evaluation

06 Conclusion & Future Work

01

Introduction

Project Objective

➔ Build a model that classifies landscape images into five categories using deep learning

Landscapes

Coast | Desert

Forest | Mountain | Glacier

Why Landscape Classification

➔ Landscape types often appear similar

➔ Automated classification helps large-scale environmental analysis

Problem Definition

➔ High visual similarity creates misclassification challenges

➔ Environmental diversity makes the dataset fine-grained

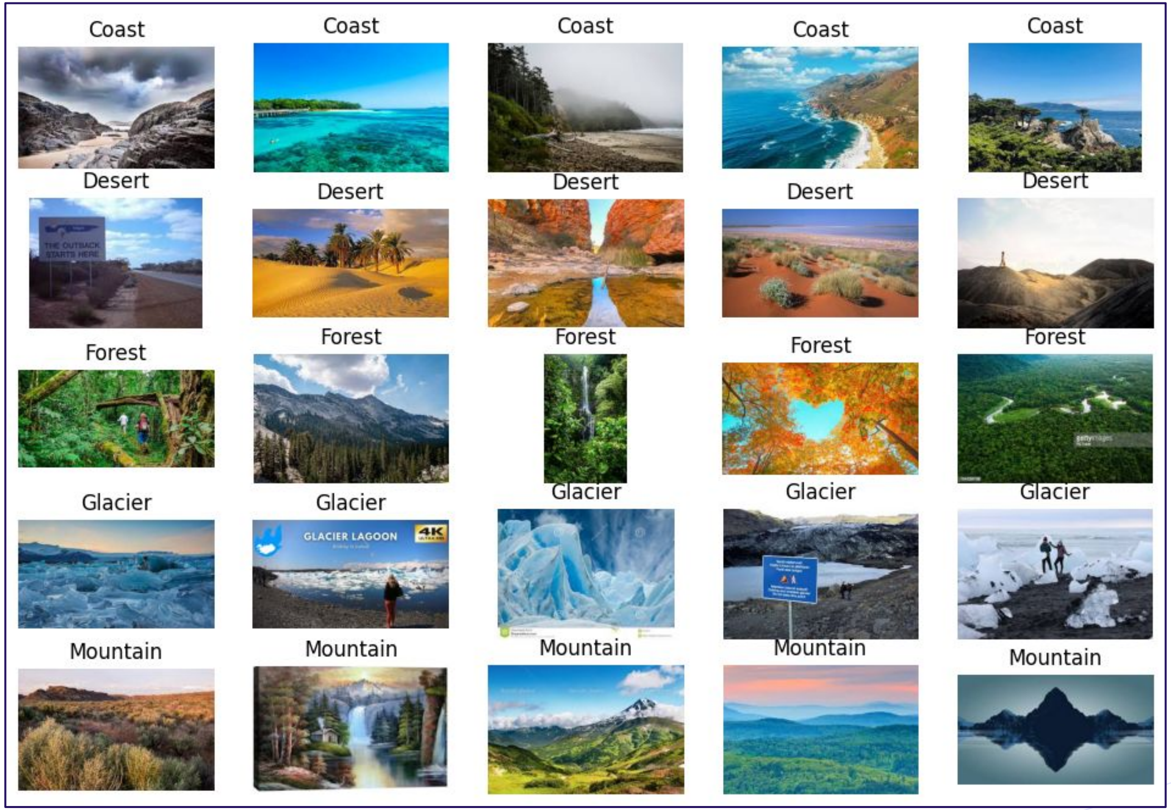
02

Dataset & EDA

Dataset Composition

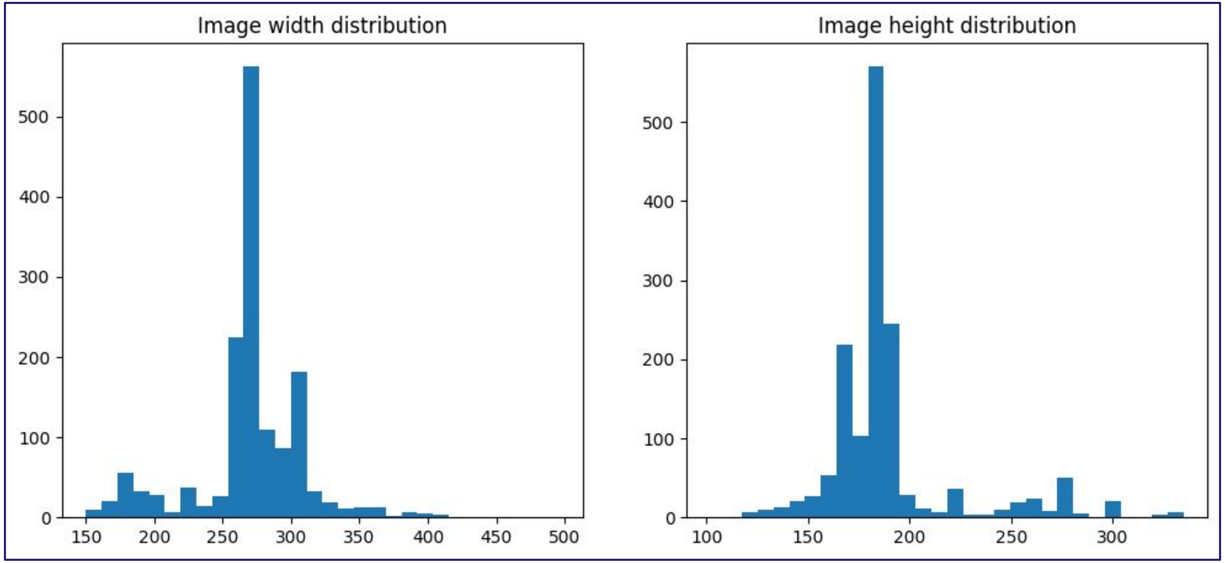
- ➔ Kaggle Data:
  Landscape Recognition Dataset
- ➔ 10,000 train | 1,500 validation | 500 test images
  - ◆ All five classes are evenly balanced

Class Overview



Random samples from each class

Image Size Distribution



➔ Image dimensions vary widely, but most cluster around 250–300 px

➔ Resizing was necessary for consistent model input

03

Methodology

Methodology

Preprocessing Pipeline

- Resize all images to 224×224
- Apply data augmentation
- Normalize images using EfficientNet built-in preprocessing

```
data_augmentation = keras.Sequential([
    layers.RandomFlip("horizontal"),
    layers.RandomRotation(0.05),
    layers.RandomZoom(0.1),
])
```

```
input_shape = (224,224,3)

inputs = keras.Input(shape=input_shape)
x = data_augmentation(inputs)
x = normalization_layer(x)

preprocessed = x
```

Data Pipeline

- Use TensorFlow cache() and prefetch() for faster data loading
- Batch size: optimized for GPU memory
- Ensures efficient input flow during training

```
train_ds = train_ds.cache().shuffle(1000).prefetch(AUTOTUNE)
val_ds = val_ds.prefetch(AUTOTUNE)
test_ds = test_ds.prefetch(AUTOTUNE)
```

```
img_size = (224, 224)
batch_size = 32

train_ds = tf.keras.preprocessing.image_dataset_from_directory(
    train_dir,
    image_size=img_size,
    batch_size=batch_size,
    shuffle=True
)

val_ds = tf.keras.preprocessing.image_dataset_from_directory(
    val_dir,
    image_size=img_size,
    batch_size=batch_size,
    shuffle=True
)

test_ds = tf.keras.preprocessing.image_dataset_from_directory(
    test_dir,
    image_size=img_size,
    batch_size=batch_size,
    shuffle=False
)
```

Model Architecture

Baseline: **ResNet50**

- Used as the initial benchmark
- Showed signs of underfitting (accuracy ~0.40)

Improved Model: **EfficientNetB0**

- Lightweight architecture
- Strong transfer learning
- Significantly improved test accuracy

04

Models & Experiments

Experiment Settings

Epochs	10
Optimizer	Adam (learning rate = 1e-3)
Loss	Sparse Categorical Crossentropy
Batch size	32

→ **Data augmentation + EfficientNet** preprocessing

→ **Prefetch()** and **cache()** used to accelerate training

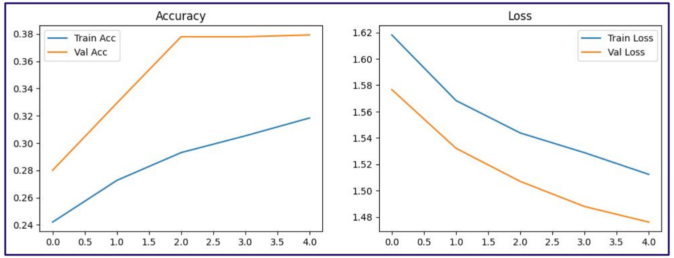
ResNet50 (Baseline)

➔ Used as the initial benchmark

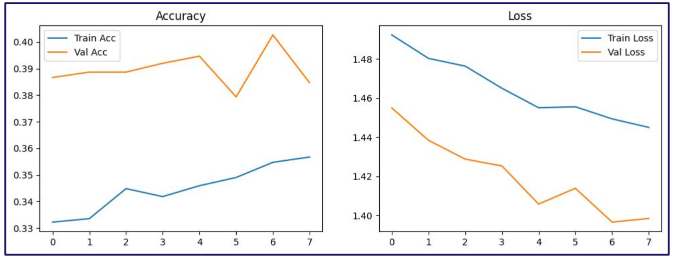
➔ **Validation accuracy** reached ***~0.40***

Helped reveal dataset difficulty and need for stronger architecture

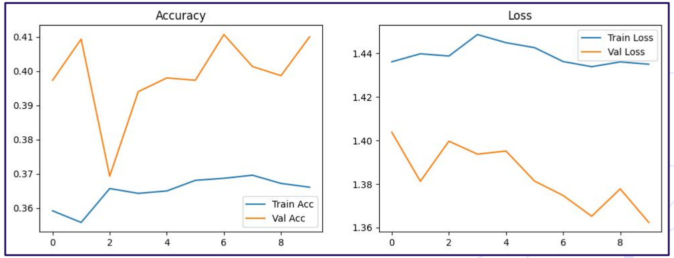
Epochs = 5



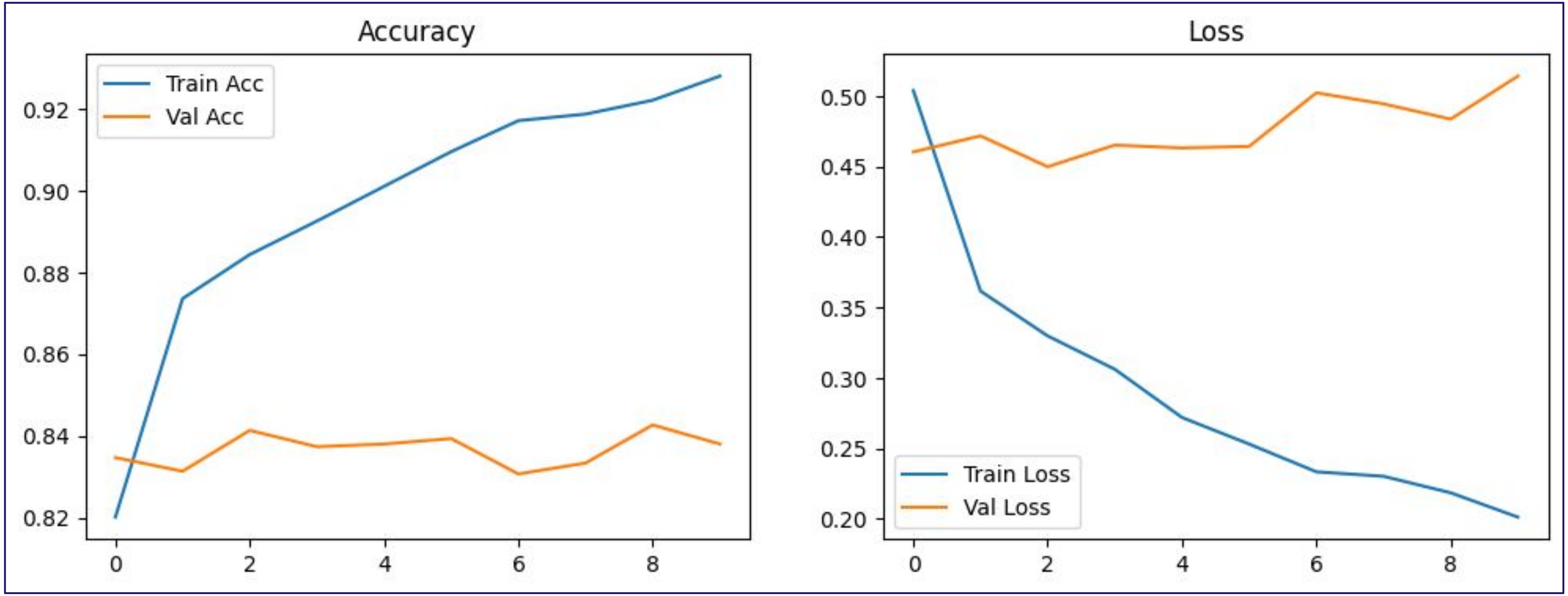
Epochs = 8



Epochs = 10



EfficientNetB0 (Improved)



→ Efficient and effective for natural image features

→ **Validation accuracy *~0.83,* Test accuracy *0.884***

Stable learning patterns and high accuracy

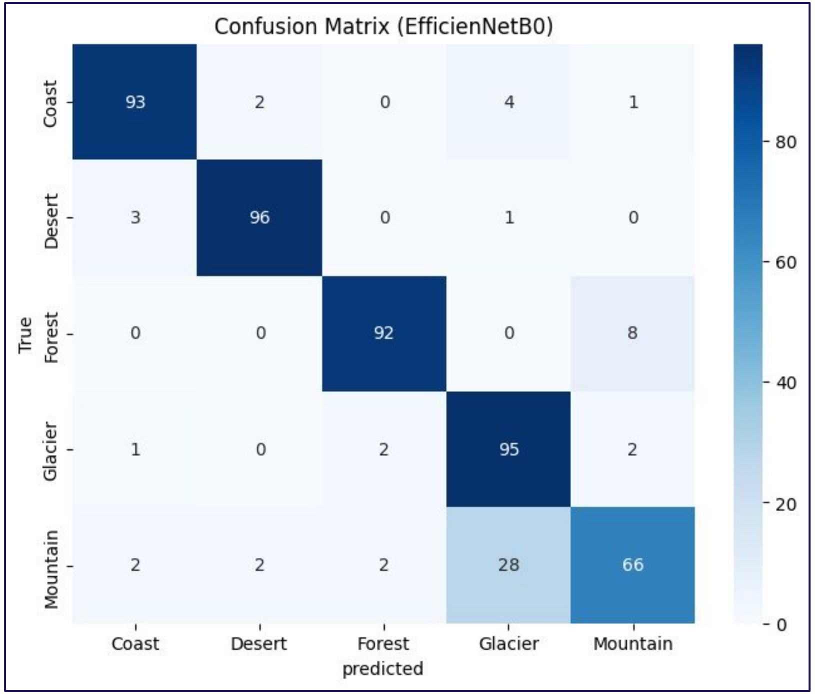
05

Results & Evaluation

Model Comparison

Model	Test Accuracy	Macro F1	Notes
ResNet50	~0.40	low	Underfitting
EfficientNetB0	**0.884**	0.88	Strong generalization

Confusion Matrix



→ Strong performance across most classes

→ Major confusion occurs in **Mountain and Glacier classes**

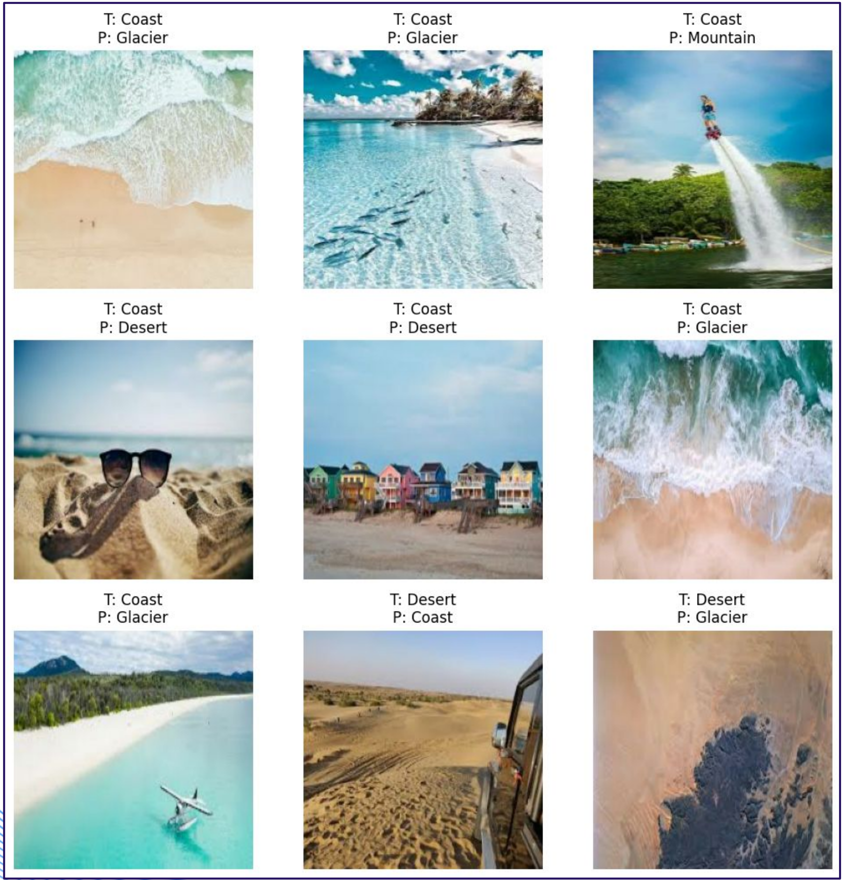
Classification Report

Classification Report:

	precision	recall	f1-score	support
Coast	0.94	0.93	0.93	100
Desert	0.96	0.96	0.96	100
Forest	0.96	0.92	0.94	100
Glacier	0.74	0.95	0.83	100
Mountain	0.86	0.66	0.75	100
accuracy			0.88	500
macro avg	0.89	0.88	0.88	500
weighted avg	0.89	0.88	0.88	500

- High precision and recall for Coast, Desert, Forest, Glacier
- Mountain shows lower recall due to visual overlap
- Macro F1-score: 0.88

Key Error Pattern



- Most misclassifications are **Mountain, Glacier**
- Most errors occurred in scenes with mixed landscape elements, such as blue water or coast–mountain combinations.

Expected in fine-grained natural scene classification tasks

Glacier

Mountain

06

Conclusion & Future Work

Conclusion

- EfficientNetB0 clearly outperformed the baseline and achieved strong generalization
- Most errors came from visually similar categories, mainly Mountain and Glacier
- The project successfully demonstrated the value of transfer learning for landscape recognition

Future Work

- Fine-tuning EfficientNet variants (B1–B3)
- More targeted augmentation for snowy terrain
- Including metadata or elevation data to improve separation

Thank You

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution