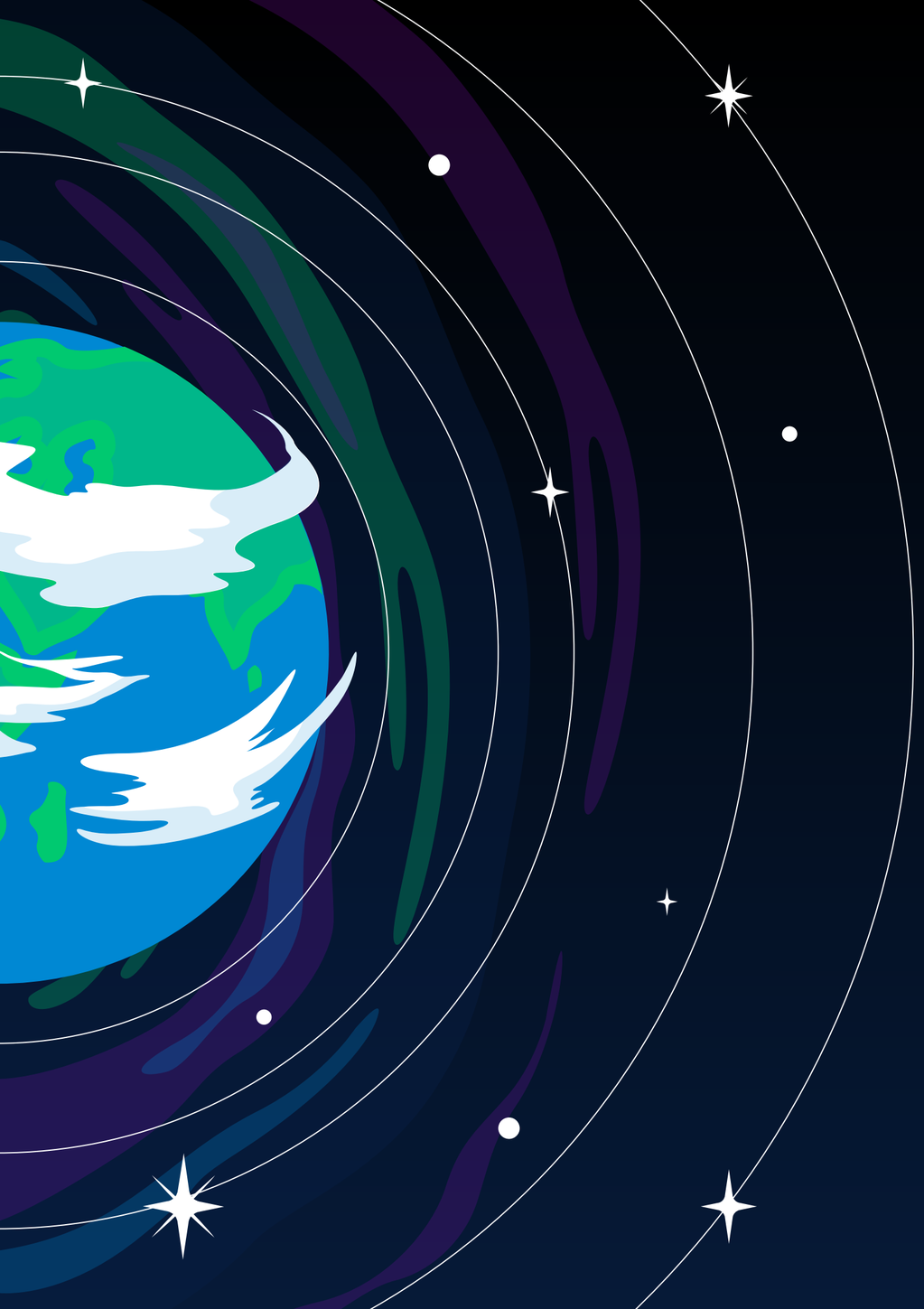


SATELLITE IMAGERY ANALYSIS

TEAM MEMBERS

Hamed Ahmed
ID: 1110147227

Mostafa Walid
ID: 1110148782

Mohamed Ibrahim
ID: 21027119

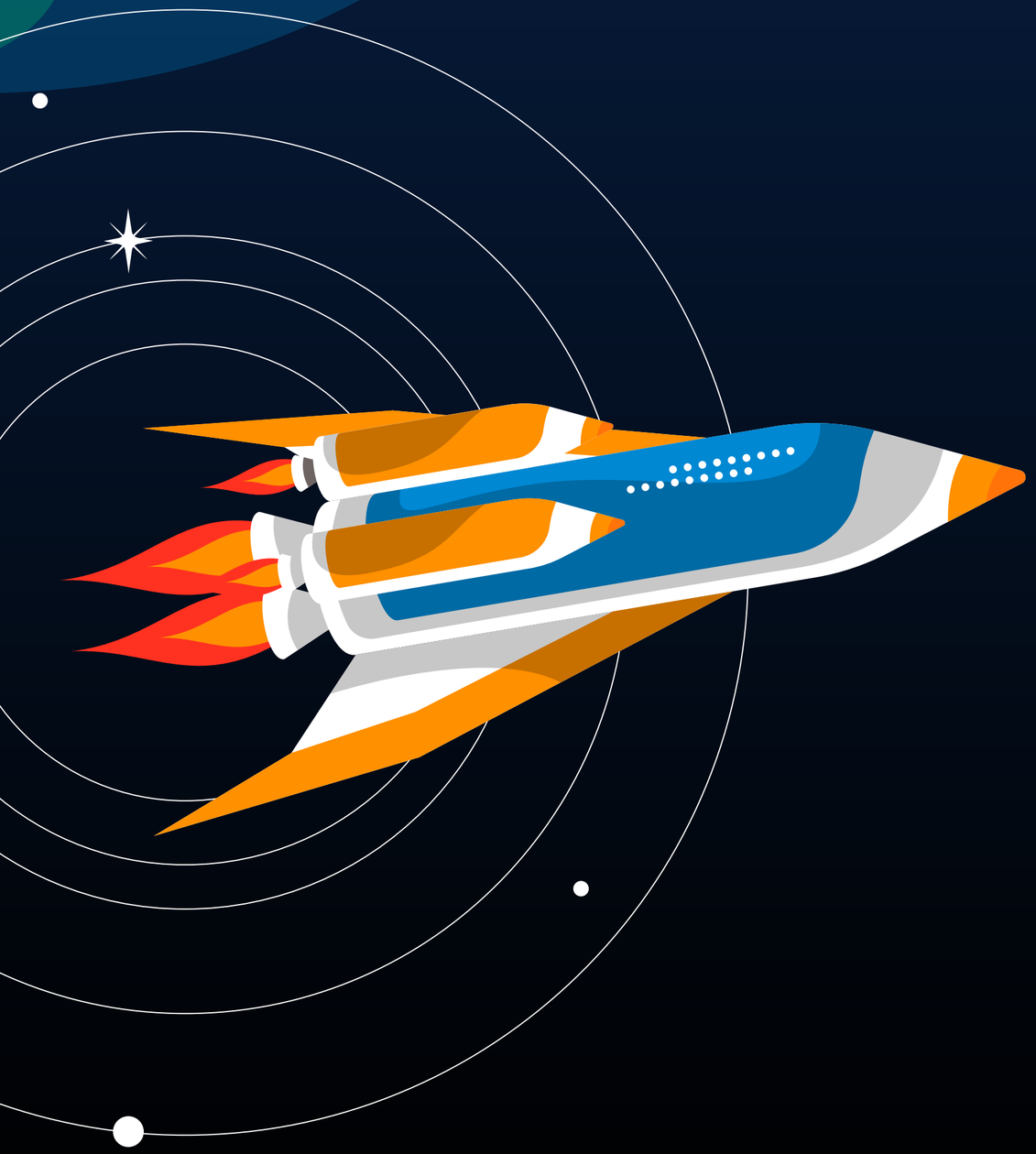
Menna Ahmed
ID: 1110147161

Mohamed Sobhy
ID: 21017293

Table of Content

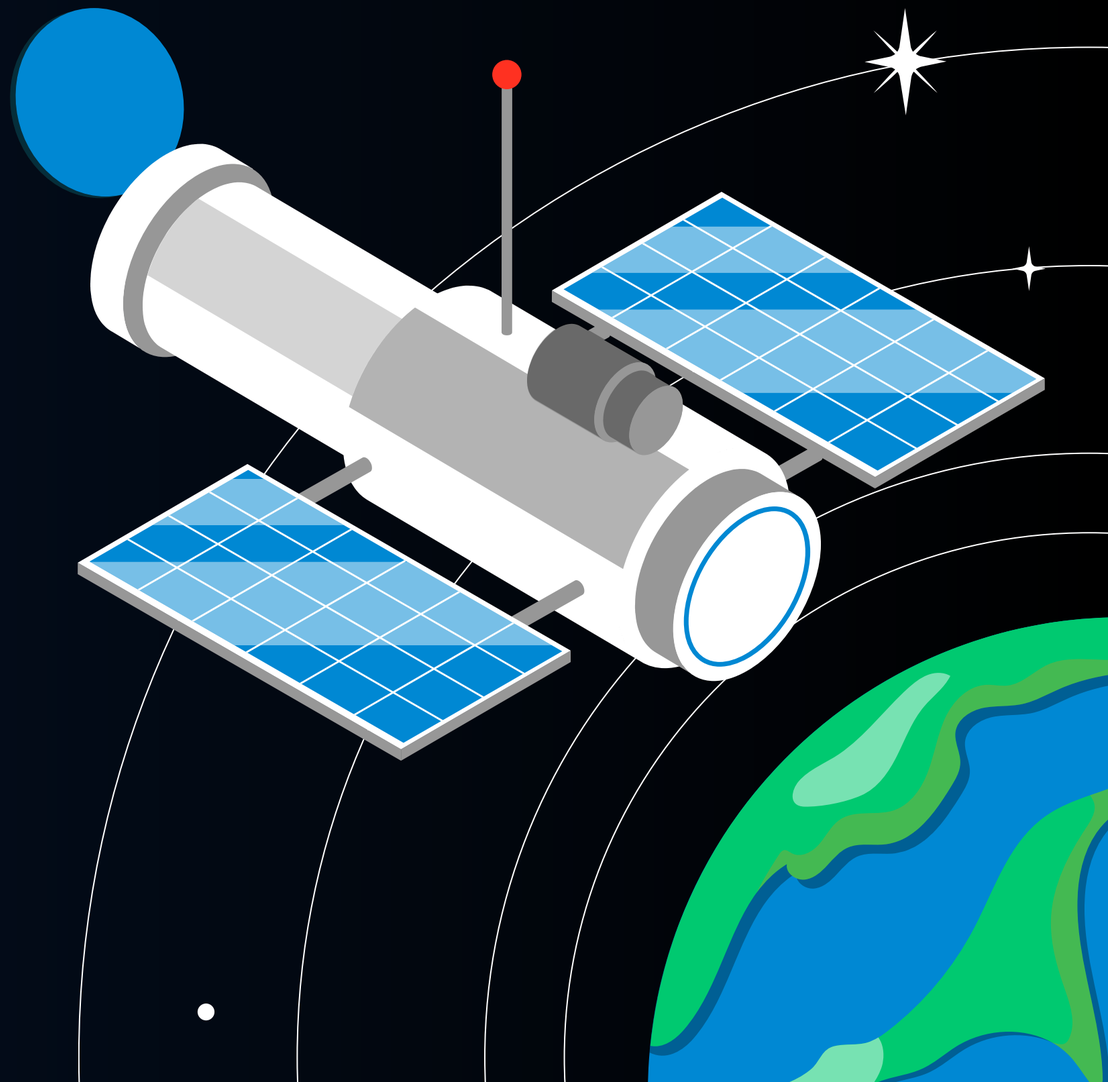
01
Introduction

Introduction



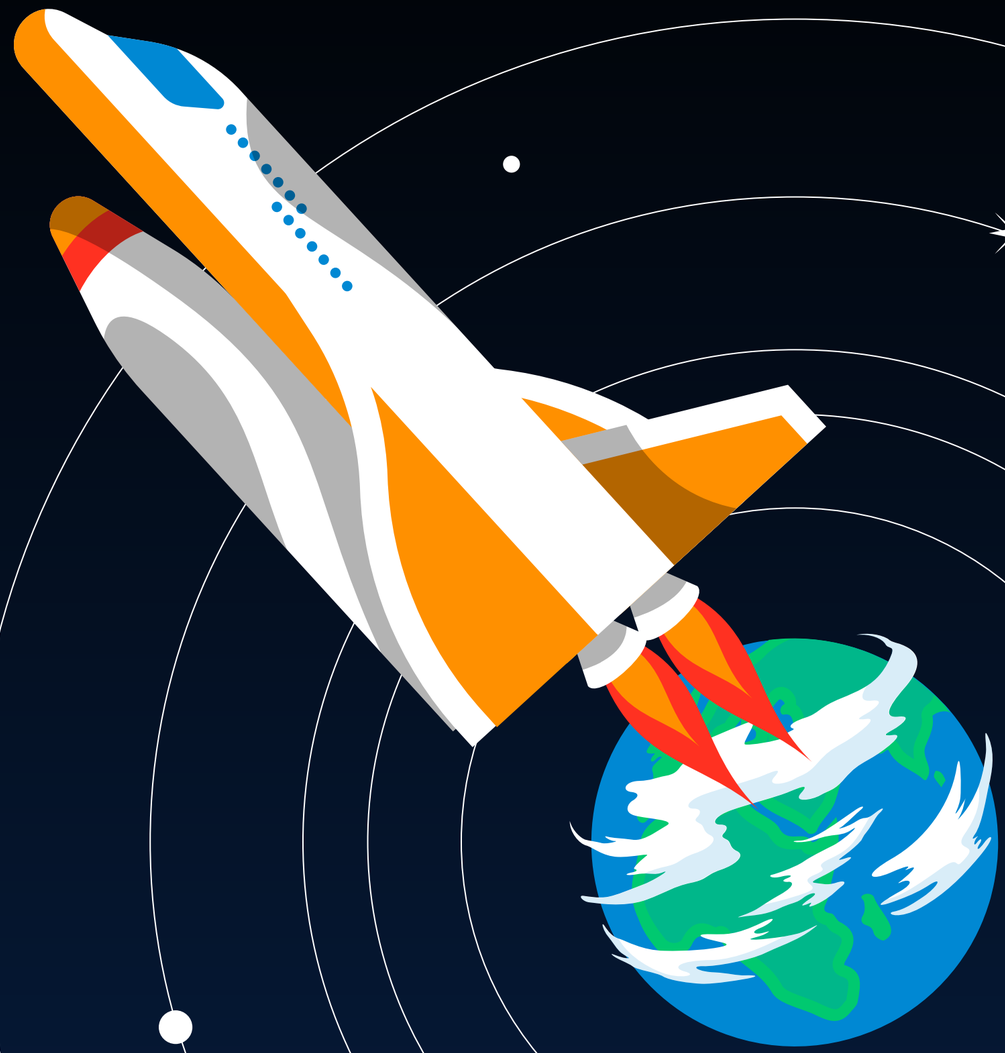
- We observe that satellite imagery is a powerful source of information as it contains more structured and uniform data, compared to traditional images.
- We seek the help of DeepGlobe Dataset, as well as using Deep Learning for Computer Vision techniques we will able to accomplish the desired outcome.
- Using those mentioned techniques, we can surely know what type of land it is among the 7 classes.
- Segmentation methods focus on processing relevant and useful pixels within the image, contributing directly to the final result.

02
Project Overview



Problem Statement

- **Environmental Monitoring:** Helps environmental agencies track changes in land use, detect deforestation, and assess the impact of climate change on various ecosystems.
- **Urban Planning and Development:** Assists city planners and developers in identifying urban growth patterns, optimizing land use, and making informed decisions about infrastructure development.
- **Disaster Management:** Supports emergency response teams by providing accurate land cover maps that help assess flood zones, fire risks, and areas vulnerable to natural disasters.



Proposed Approach

- By leveraging advanced deep learning algorithms for semantic segmentation, users can input satellite images into the application to receive detailed land cover classifications.

- The system provides immediate results, facilitating efficient analysis for environmental monitoring, urban planning, and resource management, while minimizing the time and cost associated with traditional mapping methods.

Dataset Description

Dataset

DeepGlobe Land Cover Classification Dataset:

- DeepGlobe was created in response to a competition hosted by CodaLap in May 2018
- **It Provides:**
  - High-Resolution Imagery: (2448x2448x3).
  - Annotated Ground Truth Mask: each satellite image is coupled with its mask image.
  - Diverse Land Cover Types: The dataset includes categories like (urban areas, agricultural land, forests, water bodies, and barren landscapes).

Preprocessing

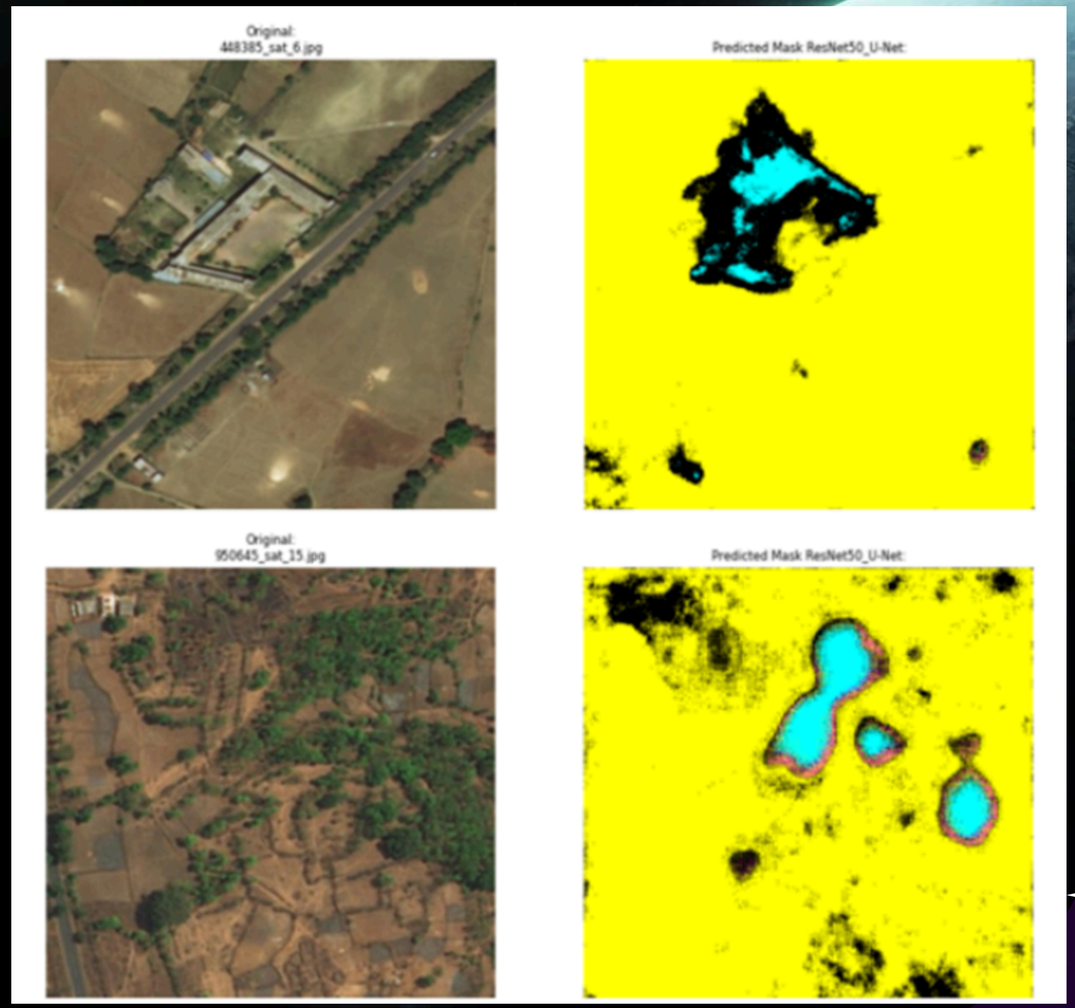
- Images are resized to 256x256 pixels and Converted to .png to avoid color distortion.
- Normalization is applied for better convergence.
- Data is split into training, validation, and test sets.
- Custom Directories and Merges are created.

1,949 Samples in Total

Example of satellite images and their corresponding masks from the dataset.

Color Labels:

- Urban Land --> Cyan
- Agriculture Land --> Yellow
- Rangeland --> Purple
- Forest Land --> Green
- Water Bodies --> Blue
- Barren Land --> White
- Unknown --> Black



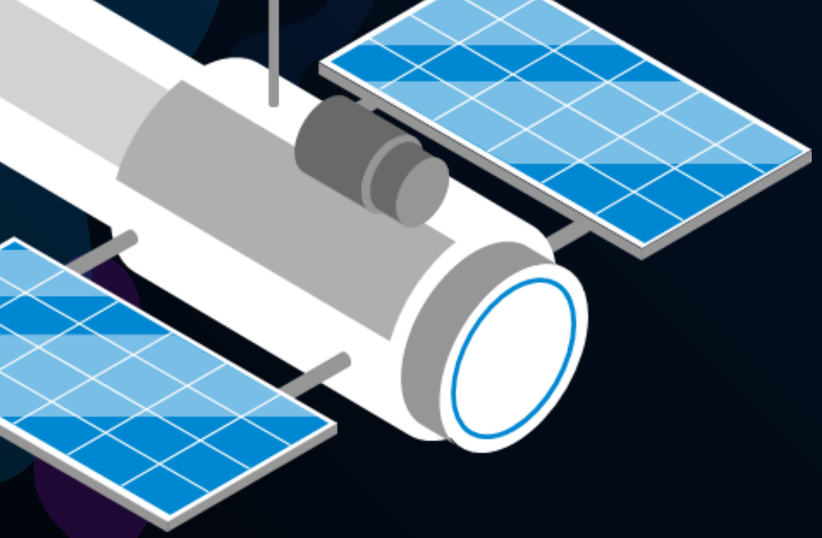
U-Net Model with ResNet50 As Backbone

U-Net

- Effective for segmentation tasks with limited data.
- Captures high-level features in downsampling and reconstructs the spatial dimensions in upsampling.

ResNet50

- A pre-trained network with proven success in feature extraction from images.
- Skip connections in ResNet allow deeper networks while avoiding vanishing gradient problems.



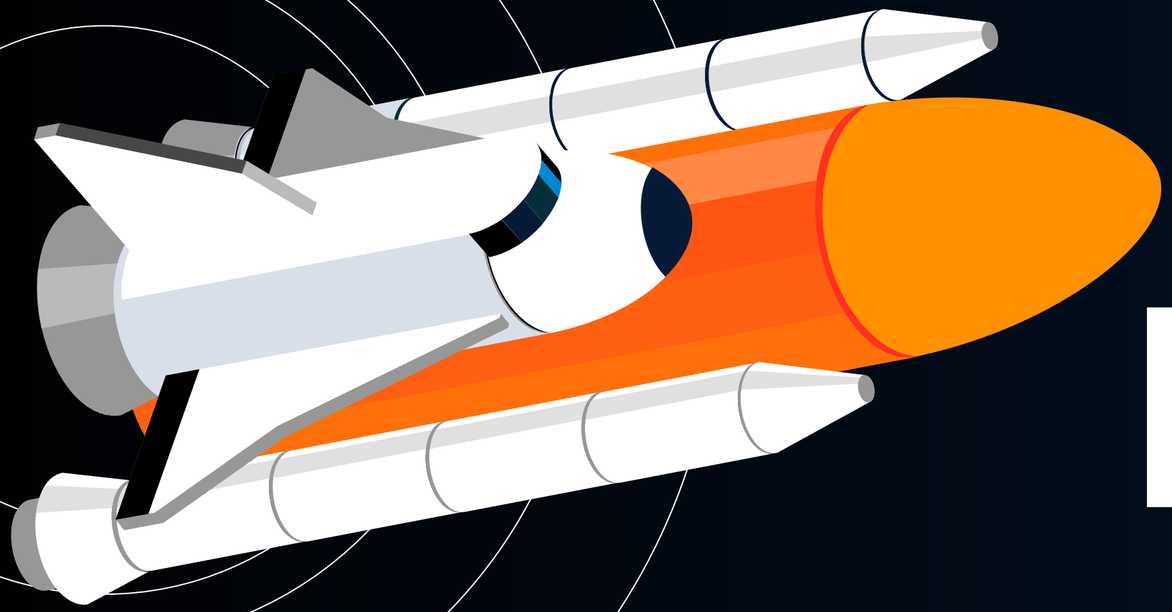
03

Literature Review

Literature Review

- Paper names and their used approaches (Methods, Dataset, and Features)

Paper	Methods	Dataset	Key Features	Accuracy	Performance
MKANet (Zhang et al., 2022) [7]	- MKANet (Multibranch Kernel-sharing Atrous convolution) network - Sobel Boundary Loss - Semantic segmentation	DeepGlobe Land Cover	- Lightweight and fast - Supports large image input size - Sobel operator for boundary detection - Focus on spatial detail recovery	**84.62% (mIoU)**	- Achieves state-of-the-art accuracy on two datasets - 2x faster inference compared to other lightweight networks
Improved Agricultural Field Segmentation in Satellite Imagery Using TL-ResUNet Architecture Safarov et al. (2022)	TL-ResUNet (Transfer Learning with Residual UNet) IoU, Precision, Recall, F1 Score, Jaccard Index	DeepGlobe,	Residual blocks for better feature learning. - Transfer learning from ImageNet-pretrained ResNet-50. - Skip connections for feature enhancement. - Special focus on agricultural land segmentation.	Focus on IoU (0.81)	Efficient agricultural field segmentation using transfer learning and residual connections.
Multiscale Context-Aware Feature Fusion Network for Land-Cover Classification of Urban Scene Imagery Siddique et al. (2023)	Multiscale Context-Aware Feature Fusion (MCN) Network	DeepGlobe	- Multiscale feature extraction using varying receptive fields - Attention mechanism to refine features. - Pixel Shuffle Decoder (PSD) for artifact-free up sampling. -pretrained ResNet-50	mIoU of 73.73	Multiscale context-aware feature fusion reduces artifacts and improves classification accuracy.



04
Methodology

Data Augmentation

```python
def augment_image_and_mask(image, mask):
    image = tf.image.resize(image, (256, 256))
    mask = tf.image.resize(mask, (256, 256))

    if np.random.rand() < 0.5:
        image = tf.image.flip_left_right(image)
        mask = tf.image.flip_left_right(mask)

    if np.random.rand() < 0.5:
        image = tf.image.flip_up_down(image)
        mask = tf.image.flip_up_down(mask)

    if np.random.rand() < 0.5:
        image = tf.image.rot90(image)
        mask = tf.image.rot90(mask)

    image = tf.image.random_brightness(image, max_delta=0.1)

    return image, mask
```

✓ 0.0s

We used different methods of data augmentation techniques like:

- Resize
- Random Flip Horizontally
- Random Flip Vertically
- Random Rotate
- Randomly Adjust Brightness: for the image only (as the mask represents classes, not visual data).

Model Architecture Breakdown

THERMAL ENERGY

Base Model: ResNet50 (with ImageNet weights).

conv1, conv2_block3_out, conv3_block4_out, conv4_block6_out, and conv5_block3_out layers used for downsampling.

Upsampling Decoder

Skip connections for detailed spatial reconstruction.
Cropping and concatenation between downsampling and upsampling layers.
Final softmax activation for pixel-wise classification.

```
#decoder (upsampling)
up_conv5 = layers.UpSampling2D(size=(2, 2))(conv5)
ch, cw = get_crop_shape(conv4, up_conv5)
crop_conv4 = layers.Cropping2D(cropping=(ch, cw))(conv4)
up6 = layers.concatenate([up_conv5, crop_conv4], axis=3)
conv6 = layers.Conv2D(512, (3, 3), padding='same')(up6)
conv6 = layers.BatchNormalization()(conv6)
conv6 = layers.Activation('relu')(conv6)
conv6 = layers.Conv2D(512, (3, 3), padding='same')(conv6)
conv6 = layers.BatchNormalization()(conv6)
conv6 = layers.Activation('relu')(conv6)
```

Loss Functions and Metrics

Dice Coefficient Loss

Custom loss function to measure segmentation performance.

```python
def dice_coef(y_true, y_pred, smooth=1):

    y_pred = tf.cast(y_pred, dtype=tf.float32)
    y_true = tf.cast(y_true, dtype=tf.float32)

    #compute intersection and sums
    intersection = K.sum(y_true * y_pred, axis=[1, 2])  #sum over height and width
    sum_true = K.sum(y_true, axis=[1, 2])  #sum over height and width
    sum_pred = K.sum(y_pred, axis=[1, 2])  #sum over height and width

    #compute the Dice Coefficient
    return (2. * intersection + smooth) / (sum_true + sum_pred + smooth)
```

Weighted Pixel-wise Crossentropy

Applied to handle class imbalance.

- **Metrics**:
  - **Dice Coefficient**: Measures overlap between predicted and true mask.
  - **Mean IoU**: Intersection-over-Union across classes.
  - **Recall** and **Precision**: For pixel-wise classification.

```python
def weighted_pixelwise_crossentropy(class_weights):
    def loss(y_true, y_pred):
        y_pred = tf.cast(y_pred, dtype=tf.float32)
        y_true = tf.cast(y_true, dtype=tf.float32)
        epsilon = K.epsilon()
        y_pred = tf.clip_by_value(y_pred, epsilon, 1. - epsilon)
        return -tf.reduce_sum(y_true * tf.math.log(y_pred) * class_weights)

    return loss
```

Training Strategy

Optimizer

Adam with an initial learning rate of 1e-3 (0.001).

Fine-Tuning Strategy

Freezing the first 25 layers of the ResNet50 backbone to retain the learned features from the ImageNet dataset.
Fine-tuning from layer 26 onwards to adapt the model specifically for the land cover classification task. This ensures that the model leverages the general knowledge from ResNet50 while adjusting the deeper layers for our specific dataset.

Training Strategy

Callbacks

ModelCheckpoint: Saves the model with the best validation score.

EarlyStopping: Stops training when no improvement in validation loss.

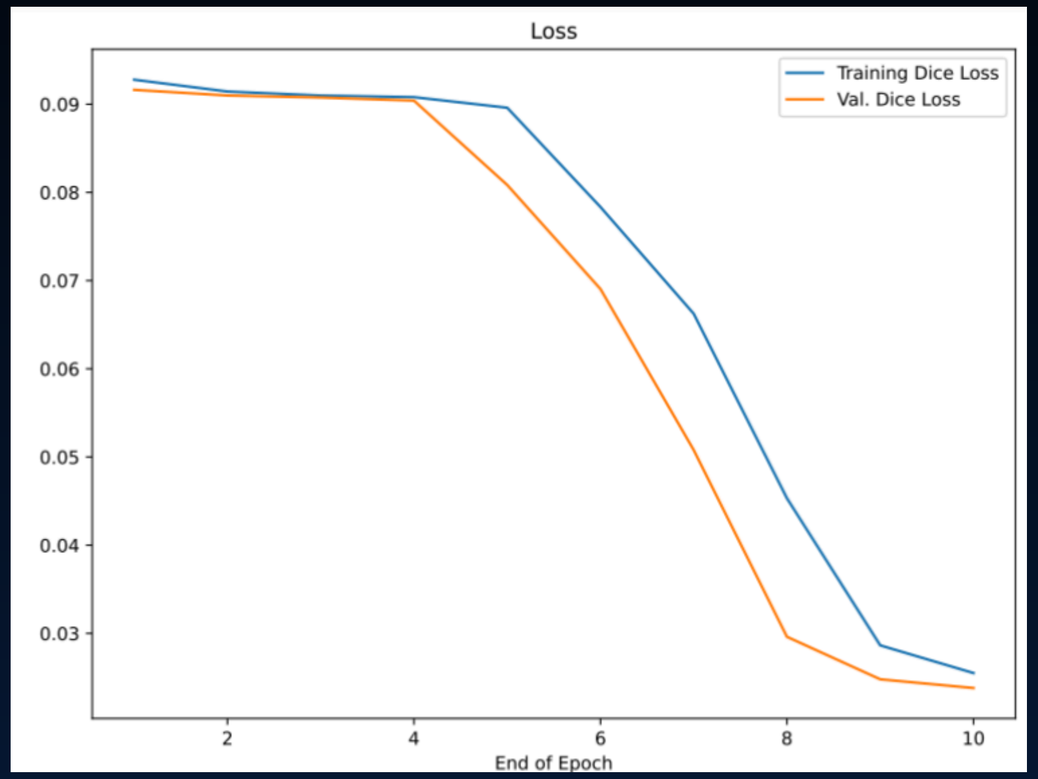
ReduceLROnPlateau: Reduces the learning rate by 0.1 when validation metrics plateau.

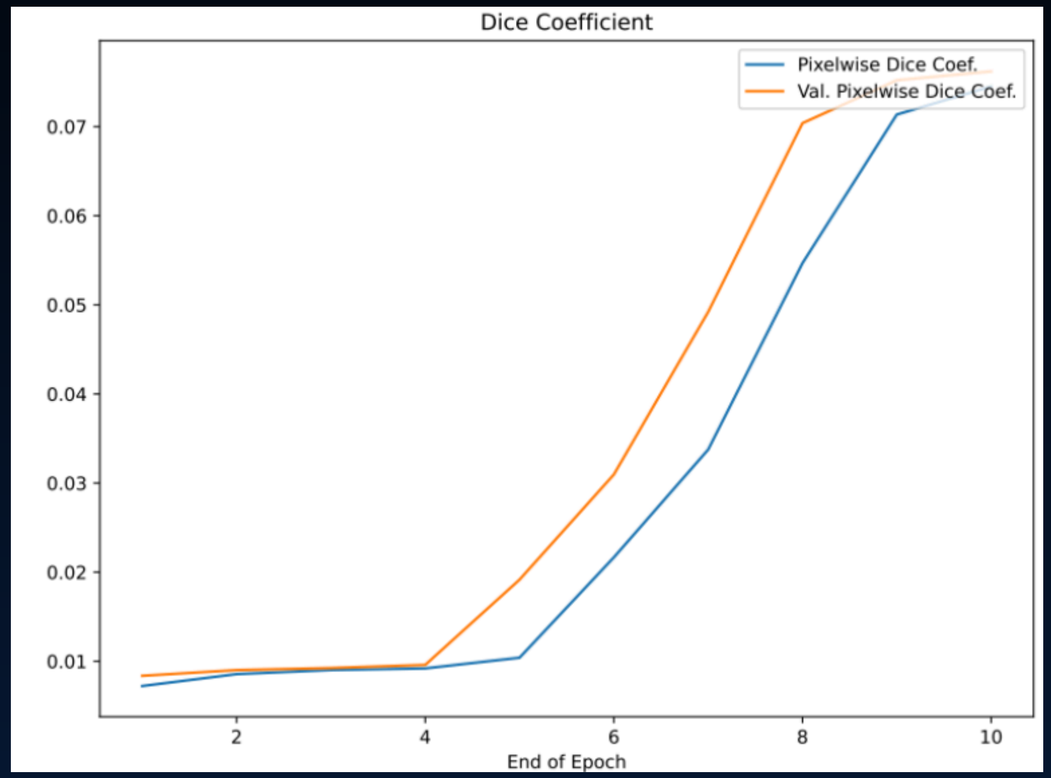
```
model_checkpoint_cb = keras.callbacks.ModelCheckpoint(
    filepath=abspath_curr + '/result/model/transfer_learning_modelv3_epoch18_21.weights.h5',
    save_best_only=True,
    save_weights_only=True  #save every 2 epochs
)

#earlyStopping callback
early_stopping_cb = keras.callbacks.EarlyStopping(patience=2,
                                                  restore_best_weights=True) #'patience' the
                                                       #'restore_best_weights' the model will
# ReduceLROnPlateau callback
reduce_lr_on_plateau_cb = keras.callbacks.ReduceLROnPlateau(
    factor=0.1, #'factor' reduces the learning rate in this case by 0.1 (90%)
    patience=3)
```

Comparison with Other Models

InceptionV3 with ResNet18:

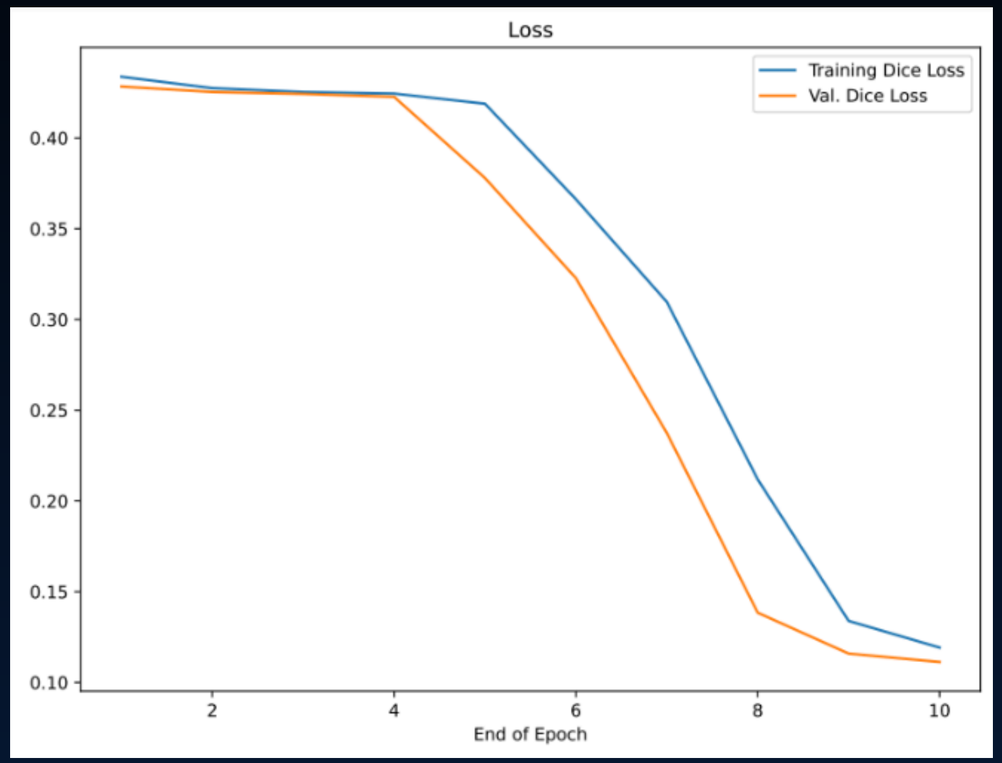


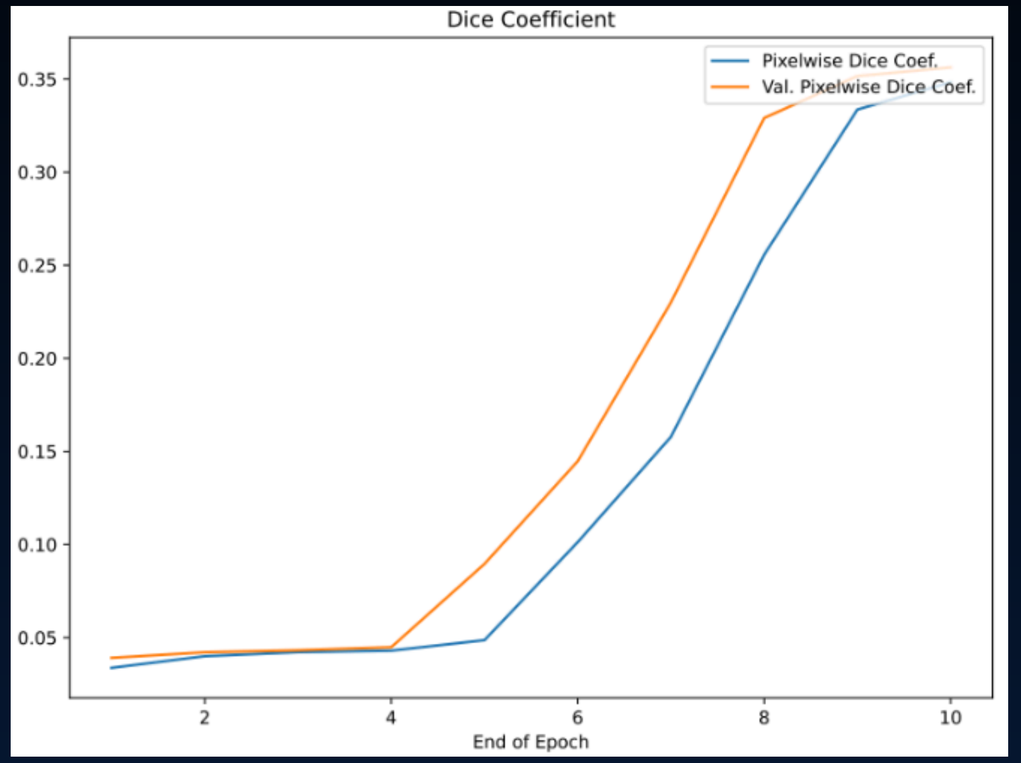


As we can see, the model didn't do well as its not build for segmentation task0 as it is best suited for classification tasks. (Dice_Coeff: 0.075 & Loss: 0.925).

Comparison with Other Models

DeepLapV3+ with ResNet50:

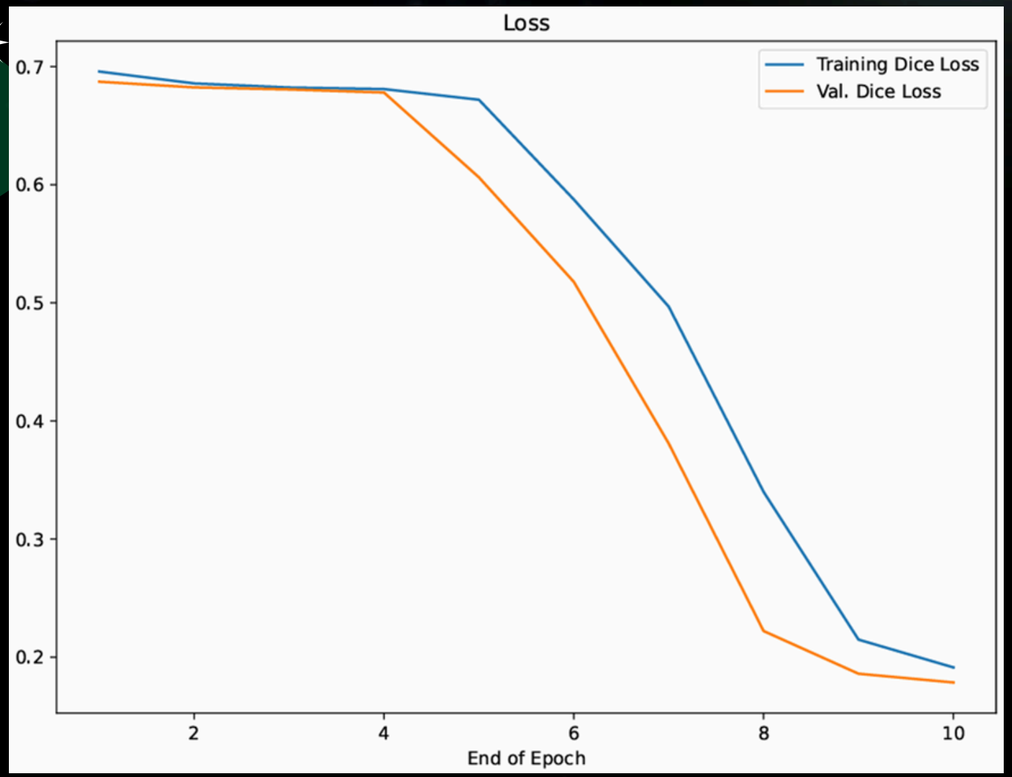


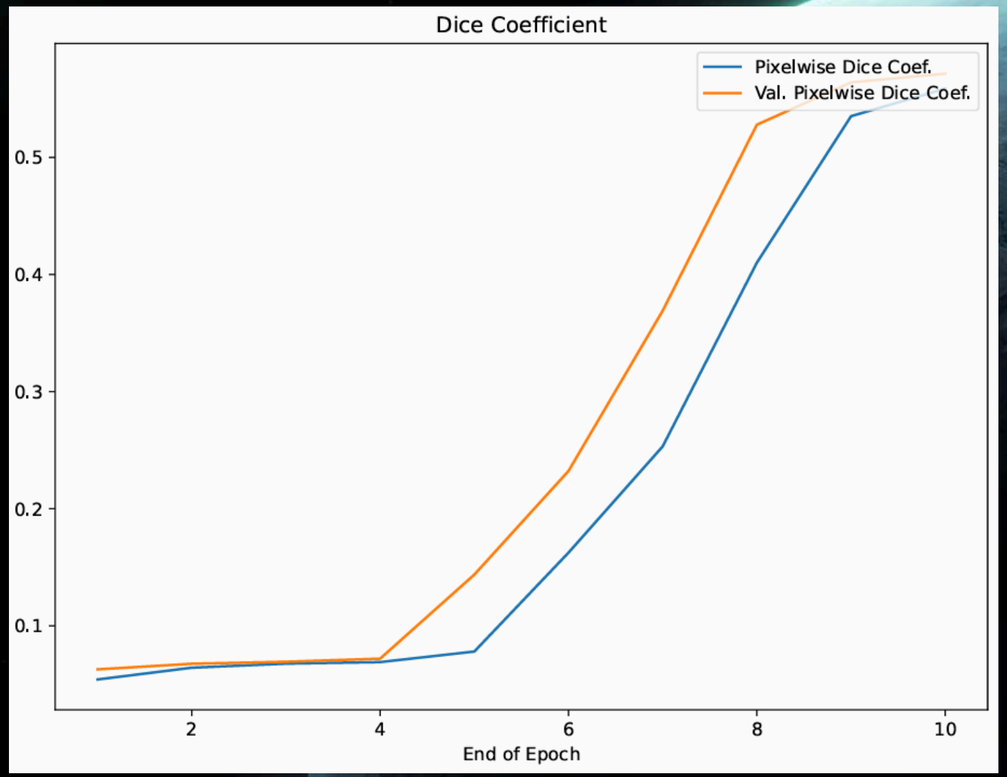


As for this one, accuracy was improved significantly, but still there's ground for more improvement. (Dice_Coeff: 0.46 & Loss: 0.54).

Comparison with Other Models

U-Net with ResNet50 (No Data Augmentation):





Here’s the final model without data augmentation, and we will view the model with the data augmentation with the in-coming slides.(Dice_Coeff: 0.56 & Loss: 0.44).

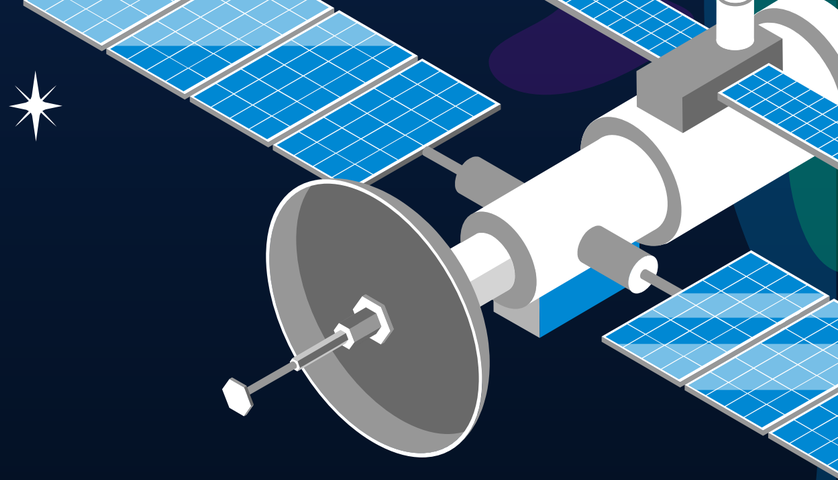
Tools Used

GoogleColab & Vscode: To run the code.

Kaggle's API: To obtain the dataset directly into notebooks without the need to save it locally.

GoogleDrive: This acts as online storage to enable easy access and manipulation.

MS Azure: (we tried to use Azure but we faced issues with DEPI provided subscriptions)



05

Results

Training Results

Epochs

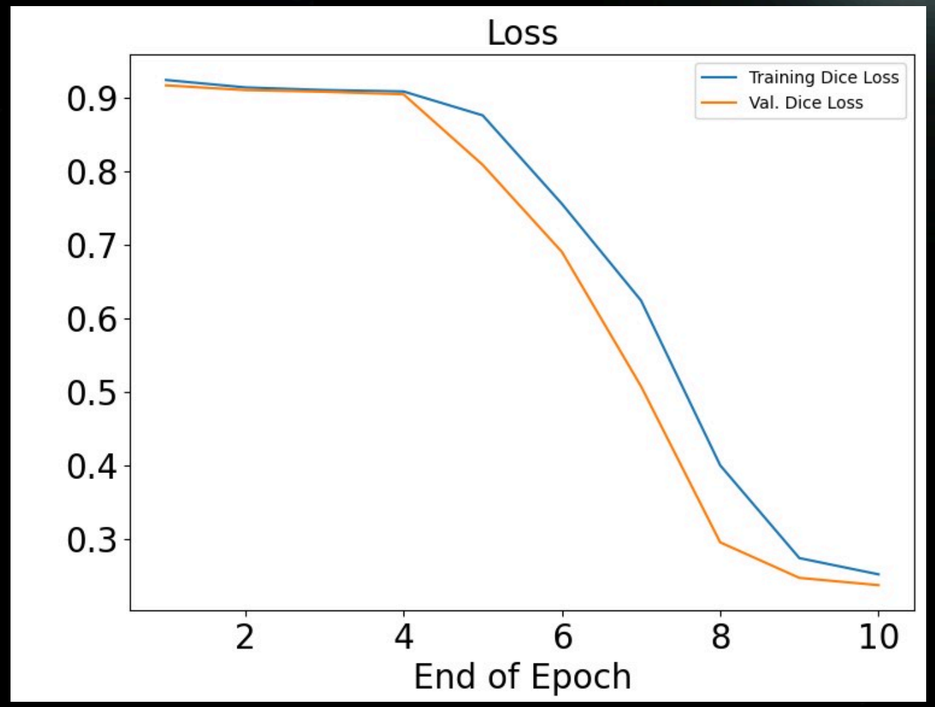
Training for 10 epochs with monitoring on validation set.

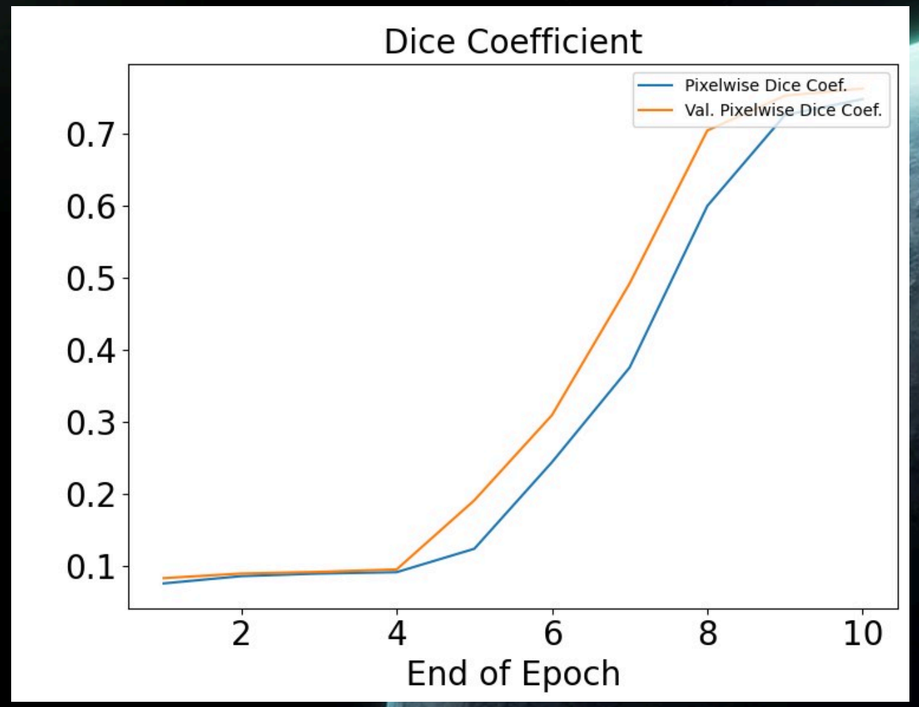
Performance

Training and validation loss (Weighted Pixelwise Crossentropy).

Dice Coefficient, Mean IoU, Recall, and Precision.

U-net with ResNet50 (Data Augmentation)





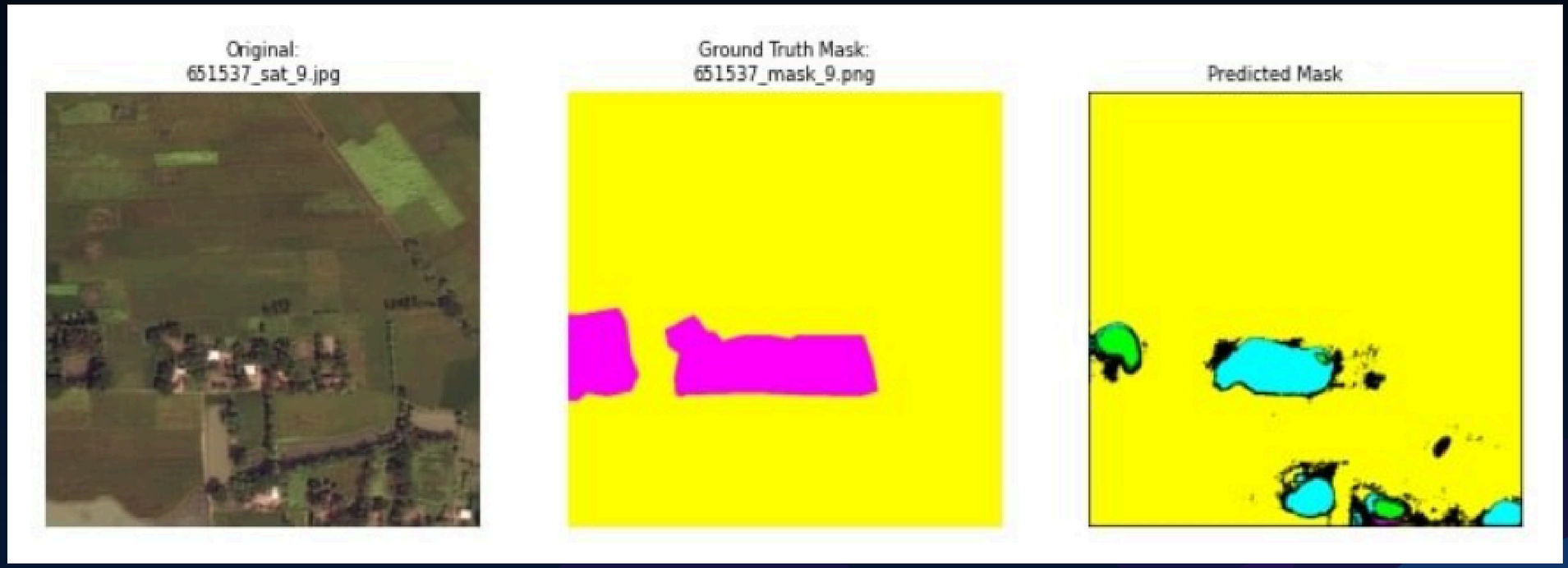
Here’s the final model with data augmentation.
(Dice_Coeff: 0.745 & Loss: 0.255).

Evaluation on Validation and Test Sets

Validation Results

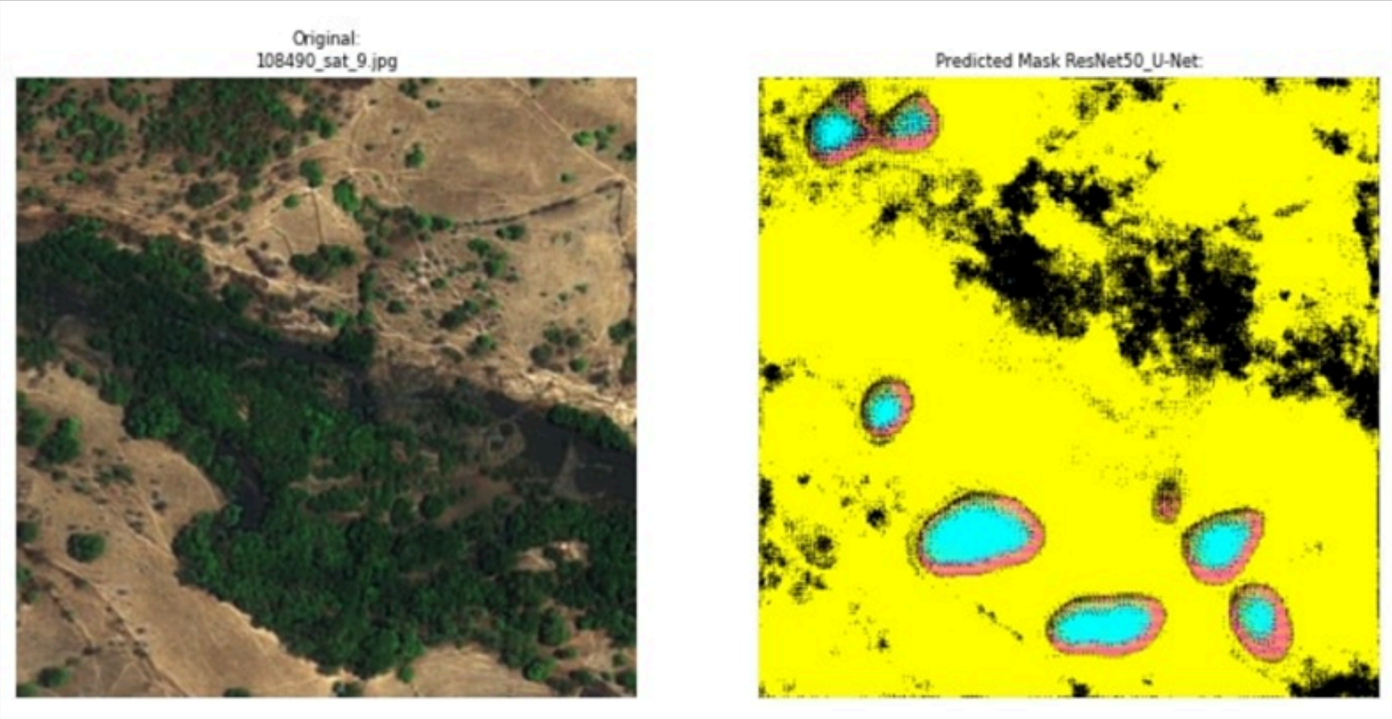
Color Labels:

- Urban Land --> Cyan
- Agriculture Land --> Yellow
- Rangeland --> Purple
- Forest Land --> Green
- Water Bodies --> Blue
- Barren Land --> White
- Unknown --> Black



Evaluation on Validation and Test Sets

Test Set Evaluation



Deployment

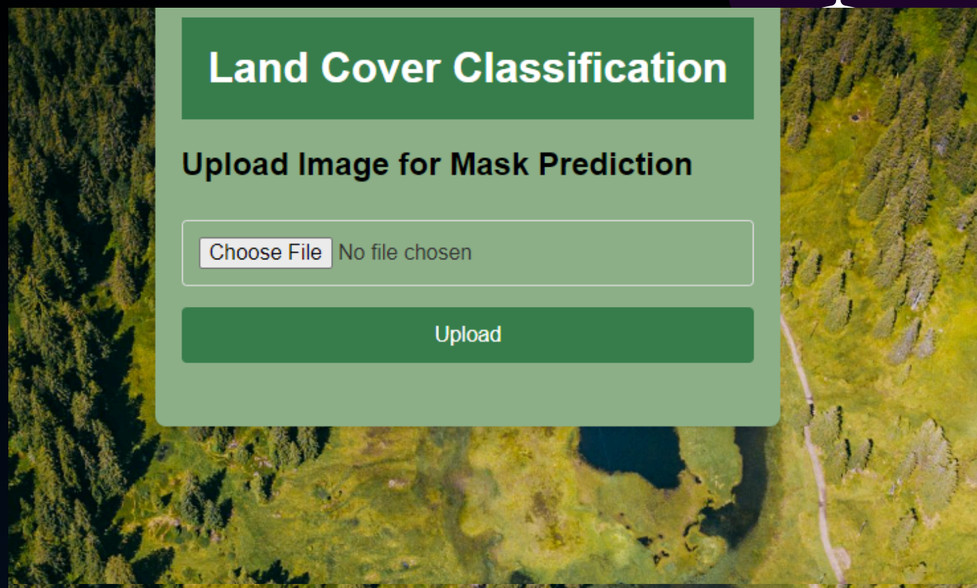
This Flask application provides a web interface for uploading images and visualizing their predicted masks. It leverages the model to perform the mask prediction.

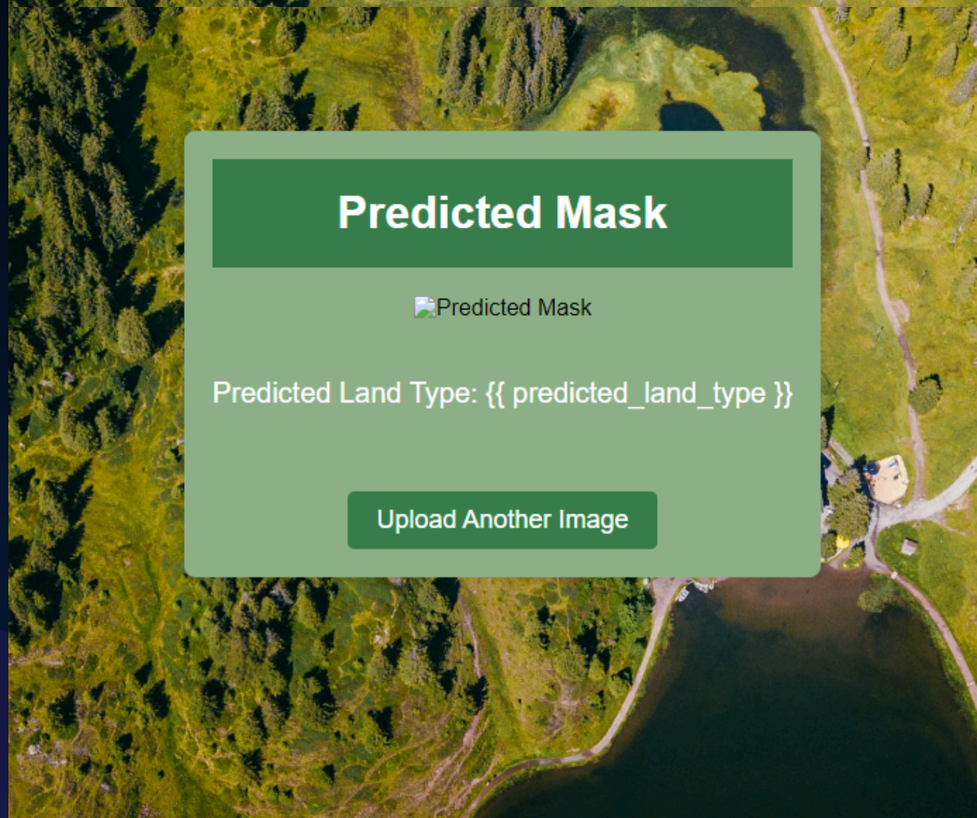
<u>Key Components:</u>

1. Flask App:
    - Creates a web server to handle HTTP requests.
    - Defines routes for handling specific actions, such as image upload and mask display.

2. HTML Templates:
    - Upload Page: Contains an HTML form with an input field for selecting an image file.
    - Upon submission, the uploaded image is sent to the Flask app.

- Result Page: Displays the predicted mask.





06

Challenges & Improvements

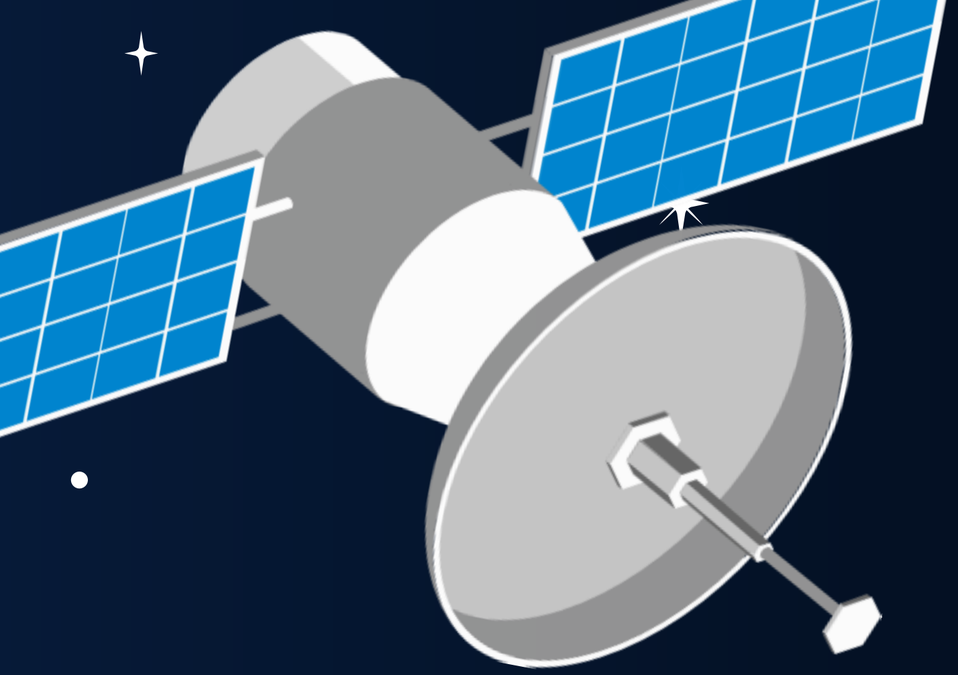
Challenges and Improvements

Challenges

- Training time and memory constraints with large models.
- Handling class imbalance in land cover data.
- Finding a good custom Loss Function methods
- Increasing the measuring metrics as much as possible.

Potential Improvements

- Data augmentation to increase robustness.
- Experimenting with deeper architectures or ensemble models.
- Use of post-processing techniques for cleaner segmentation results.



07
Conclusion

Conclusion

- Deep learning for computer vision effectively segment satellite imagery.
- DeepGlobe dataset provides valuable training data.
- System offers accurate land cover classification.
- Applications include environmental monitoring, urban planning, and disaster management.
- Improves efficiency and reduces costs compared to traditional methods.
- Contributes to sustainable development and addresses environmental challenges.

THANKS!