**Deep Learning for Satellite Image Classification**

**Description**

This project was about image classifications on low-resolution satellite images. It aimed to challenge a common real-world problem: extracting valuable features and insights from images that lack details and have special orientations. It held importance in the field of urban planning, environmental monitoring and agriculture.

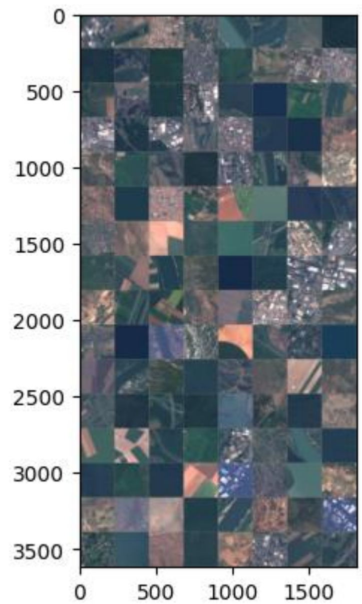
**Data Selection:**

The dataset used here was the EuroSAT dataset from the Torchvision documentation. It contained images with 64x64 pixels which were categorized in 10 classes: AnnualCrop, Forest, HerbaceousVegatation, Highway, Industrial, Pasture, PermanentCrop, Residential, River, and SeaLake. It offered a roughly balanced distribution with about 2700 samples for every class. Overall, the size of the dataset and the number of classes were suitable for a Capstone project.

**Data Preprocessing:**

- Splitting
  - The test set is 30% of the original dataset.
  - The ratio between the training set and the validation set was 70:30.
- Data Augmentation for the Training Set
  - Random Resizing and Cropping: The images in the training set were randomly cropped to 224×224 pixels with varying scales, which means the CNN sees slightly different portions and scales of the original image during each epoch, which improves its robustness by simulating diverse viewing conditions.
  - Random Horizontal Flip: The images in the training set were randomly flipped horizontally, which introduces further diversity, helping the model learn variances caused by different orientations.
- Normalization
  - All images in the dataset were standardized in a consistent way to help achieve model convergence and stable performance.

**Visualization before training:**




Above was a batch of training images. As we can see, many patches were predominantly green or brown, indicating farmland, vegetation, or forest areas, while some patches had lighter or grayish tones, indicating buildings, roads, or urban/industrial zones. A few patches might include bluish or whitish areas, which could represent water bodies or reflective surfaces.

Overall, the images were low-resolution patches with representations of different land-cover types and textures.

**Model Selection and Architecture:**

- Block Structure: the CNN had 3 convolutional blocks. Each block includes
  - Convolutional Layer, which gradually transforms the three-channel input image to 128 feature maps.
  - Max Pooling Layer, which decreases the spatial resolution.
- Output Layer: outputs the values for the 10 classes of EuroSAT.
- Activation Functions: ReLU (Rectified Linear Unit).
- Regularization Techniques
  - Batch Normalization: As mentioned in the proposal, this technique was particularly useful to low-resolution images that exhibit greater variability in pixel intensity distributions, because it helped stabilize and standardize the activations.
  - Dropout: the dropout layers helped prevent overfitting by randomly disabling a portion of neurons during each forward pass.

**Model Training:**

- Parameters
  - Learning rate: 0.01

    - Optimizer: SGD
    - Number of epochs: 25
- Early Stopping: if the validation loss did not improve within 5 epochs, the loop would automatically terminate.
- **Discussion**:
  During this phase, there were 2 models trained. One was the CNN model, the other was the pre-trained ResNet18 model, as I mentioned transfer learning within the proposal, because using models pre-trained on large, high-quality dataset would help compensate the lack of details of the satellite images.

  The CNN model achieved a best validation accuracy of 78.85% and a best validation loss of 0.6524, while the ResNet18 model achieved an accuracy of 97.57% which was very much perfect.

**Hyperparameter Tuning:**

- Ranges of hyperparameters
    - Learning rate: [0.001, 0.01]
    - Optimizer: SGD or Adam
    - Dropout rates:
- **Discussion**:
  During this phase, a grid search was conducted to determine the best combination of hyperparameters for the CNN model. The experiment log was as below:

```
----- Experiment Log -----
LR: 0.001, Optimizer: SGD, Dropout: 0.3 --> Val Acc: 0.8353
LR: 0.001, Optimizer: SGD, Dropout: 0.5 --> Val Acc: 0.8383
LR: 0.001, Optimizer: Adam, Dropout: 0.3 --> Val Acc: 0.7924
LR: 0.001, Optimizer: Adam, Dropout: 0.5 --> Val Acc: 0.7672
LR: 0.01, Optimizer: SGD, Dropout: 0.3 --> Val Acc: 0.7788
LR: 0.01, Optimizer: SGD, Dropout: 0.5 --> Val Acc: 0.7326
LR: 0.01, Optimizer: Adam, Dropout: 0.3 --> Val Acc: 0.1079
LR: 0.01, Optimizer: Adam, Dropout: 0.5 --> Val Acc: 0.4520
```

  As we could see, the SGD optimizer performed significantly better than the Adam optimizer, while the learning rate of 0.001 was more preferable than 0.01. The effect of dropout rate was not fully investigated during the grid search.

  The best combination determined was: learning rate=0.001, optimizer=SGD, dropout rate=0.5. The validation accuracy in this case was 83.83% which was better than the original 78.85%, but was still outperformed by the 97.57% of the ResNet18 model.

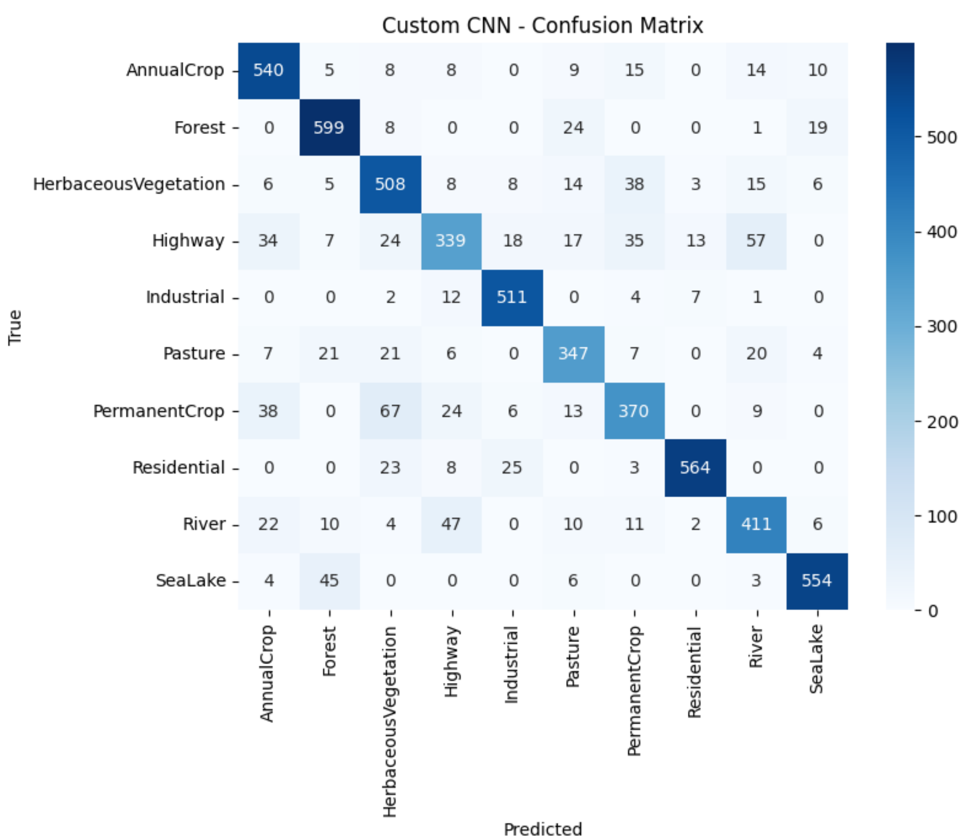
**Evaluation:**

- CNN model
  - Validation Accuracy: 83.65%
  - Classification Report:

Classification Report for Custom CNN:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| AnnualCrop | 0.83 | 0.89 | 0.86 | 609 |
| Forest | 0.87 | 0.92 | 0.89 | 651 |
| HerbaceousVegetation | 0.76 | 0.83 | 0.80 | 611 |
| Highway | 0.75 | 0.62 | 0.68 | 544 |
| Industrial | 0.90 | 0.95 | 0.92 | 537 |
| Pasture | 0.79 | 0.80 | 0.79 | 433 |
| PermanentCrop | 0.77 | 0.70 | 0.73 | 527 |
| Residential | 0.96 | 0.91 | 0.93 | 623 |
| River | 0.77 | 0.79 | 0.78 | 523 |
| SeaLake | 0.92 | 0.91 | 0.91 | 612 |
| | | | | |
| accuracy | | | 0.84 | 5670 |
| macro avg | 0.83 | 0.83 | 0.83 | 5670 |
| weighted avg | 0.84 | 0.84 | 0.83 | 5670 |

  - Confusion Matrix



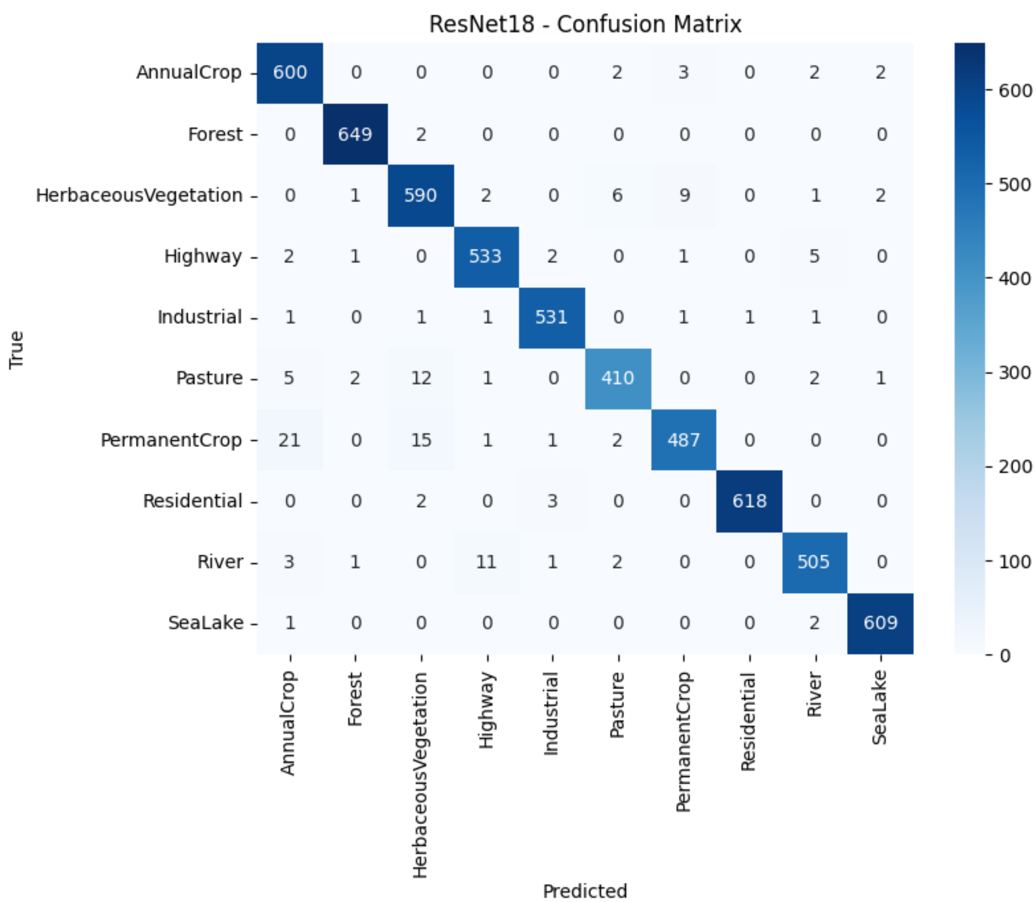
- Pre-trained ResNet18 model:
  - Validation Accuracy: 97.57%
  - Classification Report:

```
Classification Report for ResNet18:
                      precision    recall  f1-score   support

          AnnualCrop       0.95      0.99      0.97       609
              Forest       0.99      1.00      0.99       651
HerbaceousVegetation       0.95      0.97      0.96       611
             Highway       0.97      0.98      0.98       544
          Industrial       0.99      0.99      0.99       537
             Pasture       0.97      0.95      0.96       433
       PermanentCrop       0.97      0.92      0.95       527
         Residential       1.00      0.99      1.00       623
               River       0.97      0.97      0.97       523
             SeaLake       0.99      1.00      0.99       612

            accuracy                           0.98      5670
           macro avg       0.98      0.97      0.97      5670
        weighted avg       0.98      0.98      0.98      5670
```

  - Confusion Matrix:



- **<u>Discussion:</u>**
  Based on the classification report and the confusion matrix of the CNN model, we could see that the model had trouble identifying the Farmland categories, such as AnnualCrop, HerbaceousVegetation and PermanentCrop. Also, it got confused a lot with the urban-related classes, such as Highway. All these signs were suggesting that the model was not enough to learn the subtle differences among similar categories.

  On the other hand, the ResNet18 model was doing great on all the classes with a few mistakes on PermanentCrop.

**Fine-Tuning:**

Based on the evaluation results from the previous section, the essentiality of a more sophisticated model was identified. In this case, the original architecture was modified by adding 1 more convolutional block and 1 more convolution layer to each block.

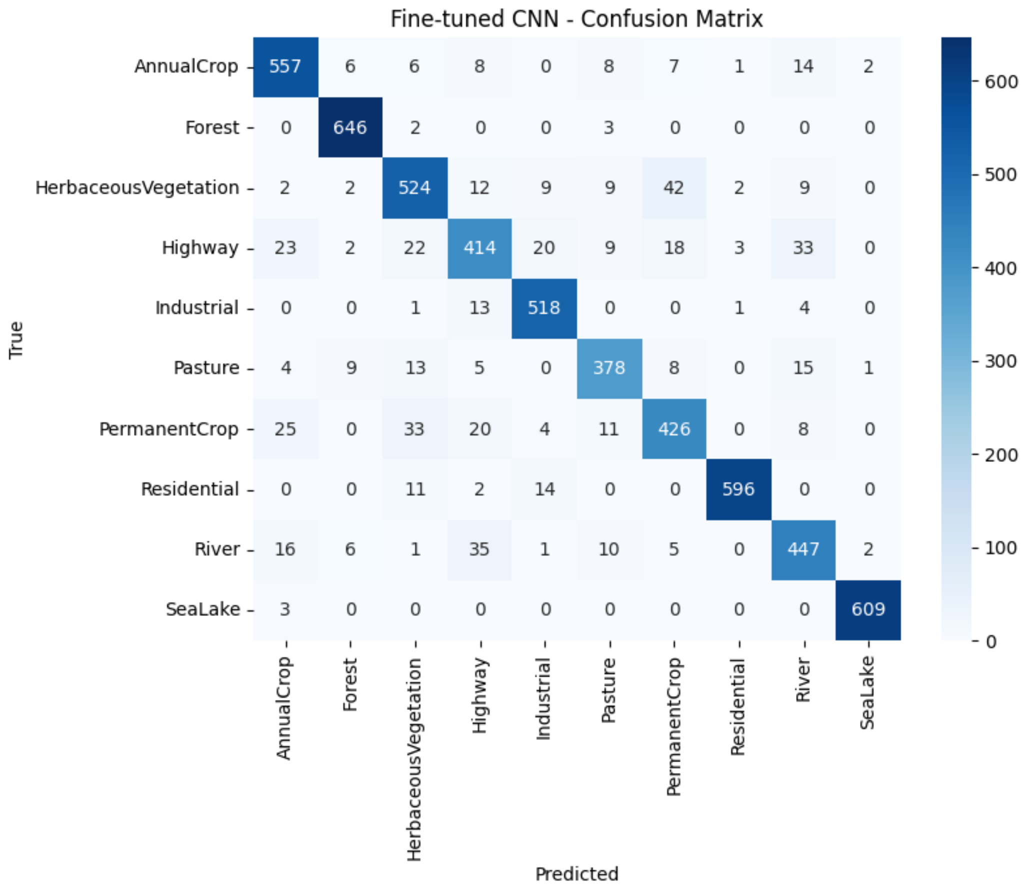
The fine-tuned model was trained with the best combination of hyperparameters and evaluated. Its validation accuracy was 90.21%. Its classification report was as below:

```
Classification Report for Fine-tuned CNN:
                      precision    recall  f1-score   support

          AnnualCrop       0.88      0.91      0.90       609
              Forest       0.96      0.99      0.98       651
HerbaceousVegetation       0.85      0.86      0.86       611
             Highway       0.81      0.76      0.79       544
          Industrial       0.92      0.96      0.94       537
             Pasture       0.88      0.87      0.88       433
       PermanentCrop       0.84      0.81      0.82       527
         Residential       0.99      0.96      0.97       623
               River       0.84      0.85      0.85       523
             SeaLake       0.99      1.00      0.99       612

            accuracy                           0.90      5670
           macro avg       0.90      0.90      0.90      5670
        weighted avg       0.90      0.90      0.90      5670
```

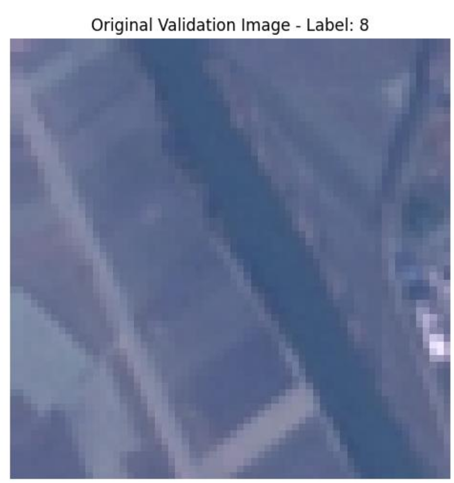
Its confusion matrix was as below:



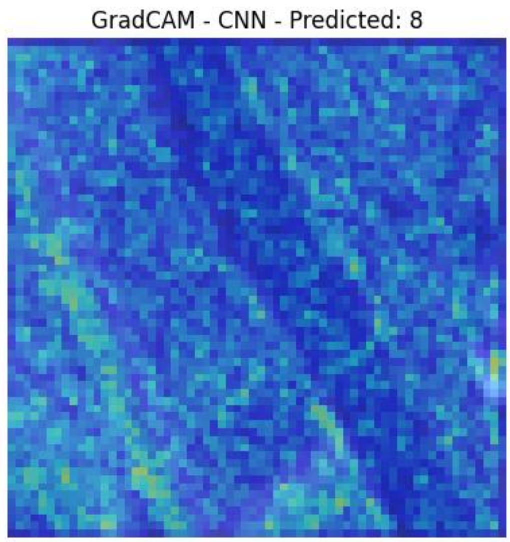

Based on the results above, we could see that its performance was significantly better than the original CNN model, though it still did not outperform the pre-trained ResNet18 model.

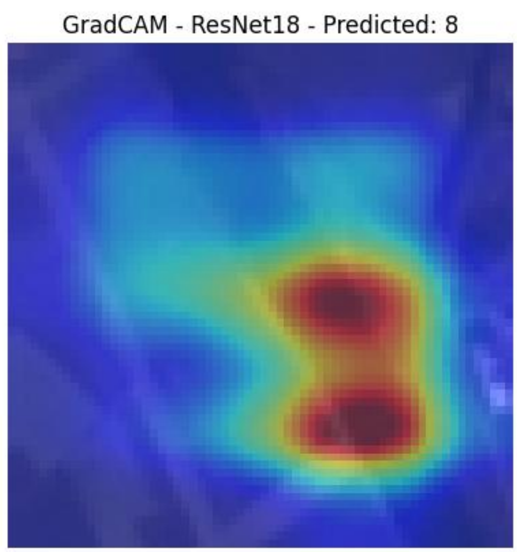
**GradCAM:**

The GradCAM technique was also used to analyze what regions the model is paying attention to. In this case, a raw image of "River" as below was selected:



The visualization of GradCAM results of the fine-tuned CNN model and the pre-trained ResNet18 model were also plotted:




According to the plots above, we could see that, the CNN model was paying more attention to the edges of the rivers and different land textures, while the ResNet18 model was focusing more on the river itself.

**Final Model Testing:**

The fine-tuned CNN model was tested on the held-out test dataset and achieved a test accuracy of 89.95%. Since the number was close to its validation accuracy, no signs of overfitting or underfitting was observed.

The ResNet18 model was tested on the held-out test dataset and achieved a test accuracy of 97.16%. Since the number was close to its validation accuracy, no signs of overfitting or underfitting was observed.

**Conclusion:**
Overall, after several steps of tuning, the CNN model was performing very well on the chosen dataset, yet it was not able to beat the pre-trained ResNet18 model which was doing too well on classifying the satellite images, proving that models pre-trained on large, high-quality dataset would help compensate the lack of details of the low-resolution images.

For actionable future improvements, we could either keep tuning the CNN model by building more complex architectures and searching over wider ranges of hyperparameters, or fine tune the ResNet18 model a little bit to make it work even better. Furthermore, we could test some totally different neural network architectures, such as autoencoders, on the dataset.

**Github Page:**
https://github.com/fiozzt/6147---Project.git