Satellite Images Classification
Final Presentation

Simone Zagaria (matr 2145389)
Gennaro De Cicco (matr 2128464)
Petra Udovicic (matr 2119592)
Sai Swaroop Chittoor (matr 1916247)
Ana Carina Branescu (matr 2125078)

Fundamentals of Data Science

2023/2024 Final Project

Dataset preparation

	image_path	label
0	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Cloudy
1	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Cloudy
2	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Cloudy
3	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Cloudy
4	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Cloudy
...	...	...
5626	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Water
5627	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Water
5628	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Water
5629	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Water
5630	C:\Users\nephr\Desktop\Uni Nuova\FDS\FDS_PROJE...	Water

5631 rows × 2 columns

- Firstly, we created a dataset from the downloaded Kaggle folder. This dataset includes, for each image, its path and the its corresponding category

- As expected, the dataset contains 1500 images for each category, with the exception of the ‘Desert’ category

```
{'Cloudy': 1500, 'Desert': 1131, 'Green_Area': 1500, 'Water': 1500}
```

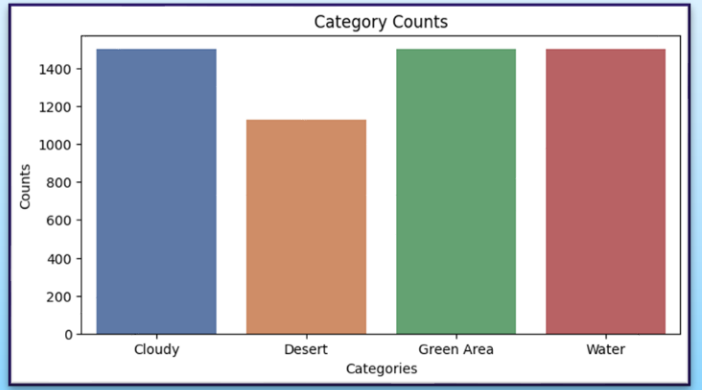
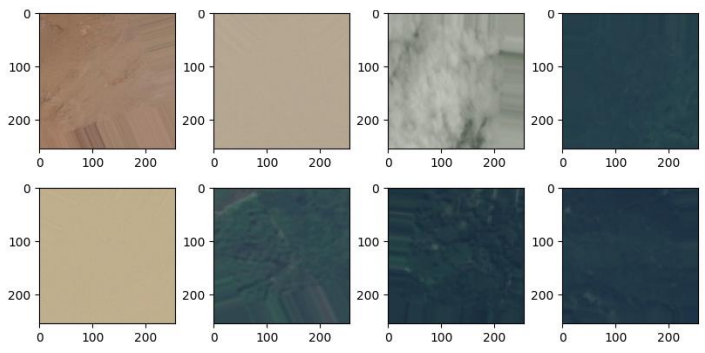


Image Pre-processing

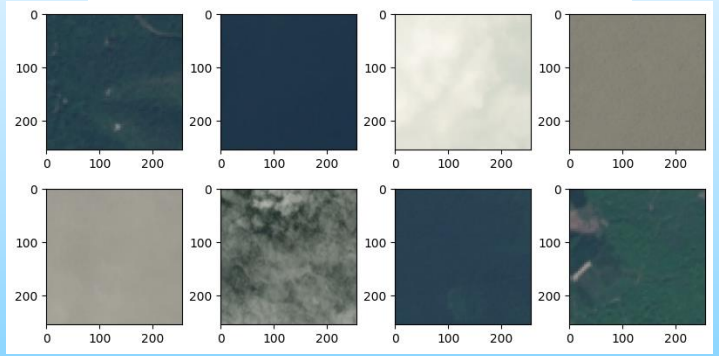
Pre-processed **Training** Data Samples



- The dataset was then divided in training and testing sets. We opted for a 80-20 split
- Images from the training test were zoomed, sheared, horizontally and vertically flipped, and rotated randomly to allow the model to learn new pattern and improve robustness

- Both the images from the training and testing set were resized to 256 x 256 pixel for normalization purposes
- Beyond resizing, we didn't make other alterations to testing images because as testing should be conducted on real-world images without modification

Pre-processed **Testing** Data Samples



Model Creation and Training

```python
def create_model():
    model = keras.models.Sequential([
        # First convolutional layer with 32 filters, each of size (2,2)
        layers.Conv2D(32, (2,2), activation='relu', input_shape=(255, 255, 3)),
        layers.MaxPooling2D(2,2),
        # Second convolutional layer with 64 filters, each of size (2,2)
        layers.Conv2D(64, (2,2), activation='relu'), #second convolutional layer
        layers.MaxPooling2D(2,2),
        # Third convolutional layer with 64 filters, each of size (3,3)
        layers.Conv2D(64, (3,3), activation='relu'),
        layers.MaxPooling2D(2,2),
        layers.Flatten(),
        layers.Dropout(0.1),
        layers.Dense(512, activation='relu'),
        layers.Dropout(0.1),
        # Output layer with 4 neurons
        layers.Dense(4, activation='softmax')
    ])

    # Model compilation
    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
    return model
```

- We defined a function for the creation of the CNN model, containing various layers, such as:
  - Convolutional layers
  - Pooling layers
  - Fully connected layers
  - Dropout layers

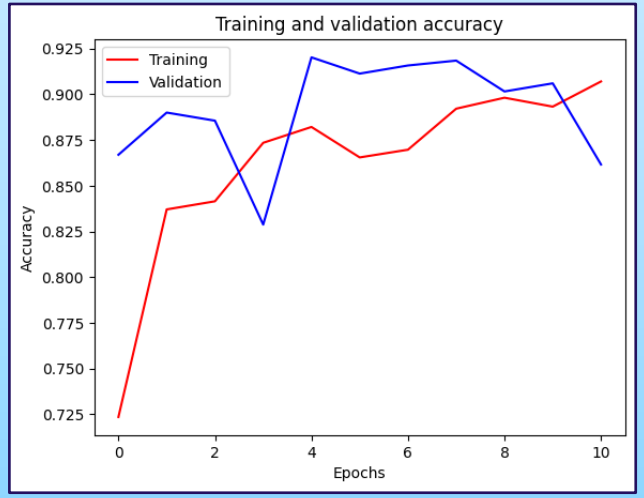
```python
# Creating an instance of the untrained model
model = create_model()

# Training the creared model
history = model.fit(train_generator,
                    epochs=11,
                    verbose=1,
                    validation_data=test_generator)
```

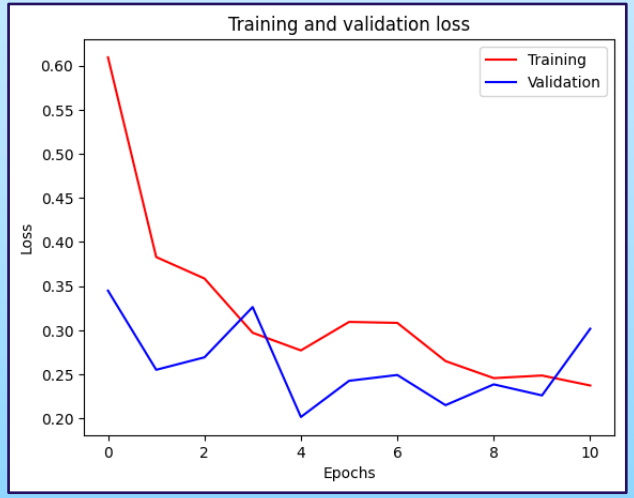
- We trained the model using the pre-processed training images generator. We decided to choose 11 epochs since it's generally considered an optimal number of iteration for training a dataset

Accuracy and Loss

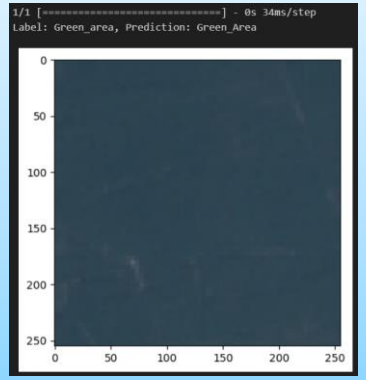
- Accuracy graph:

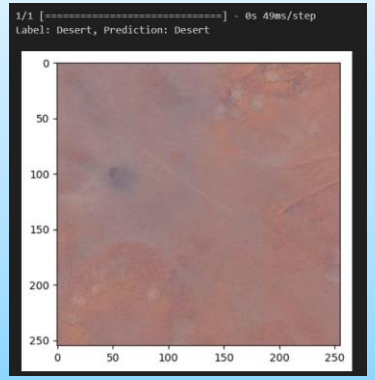


- Loss graph:



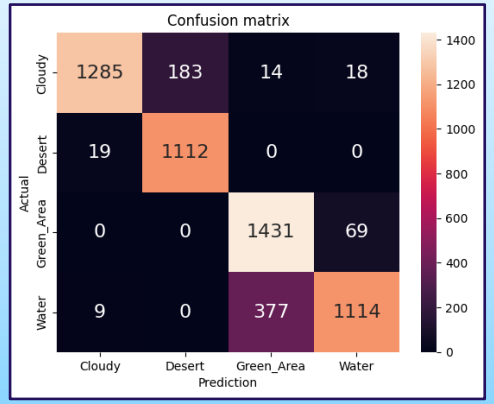
Predictions and Confusion Matrix

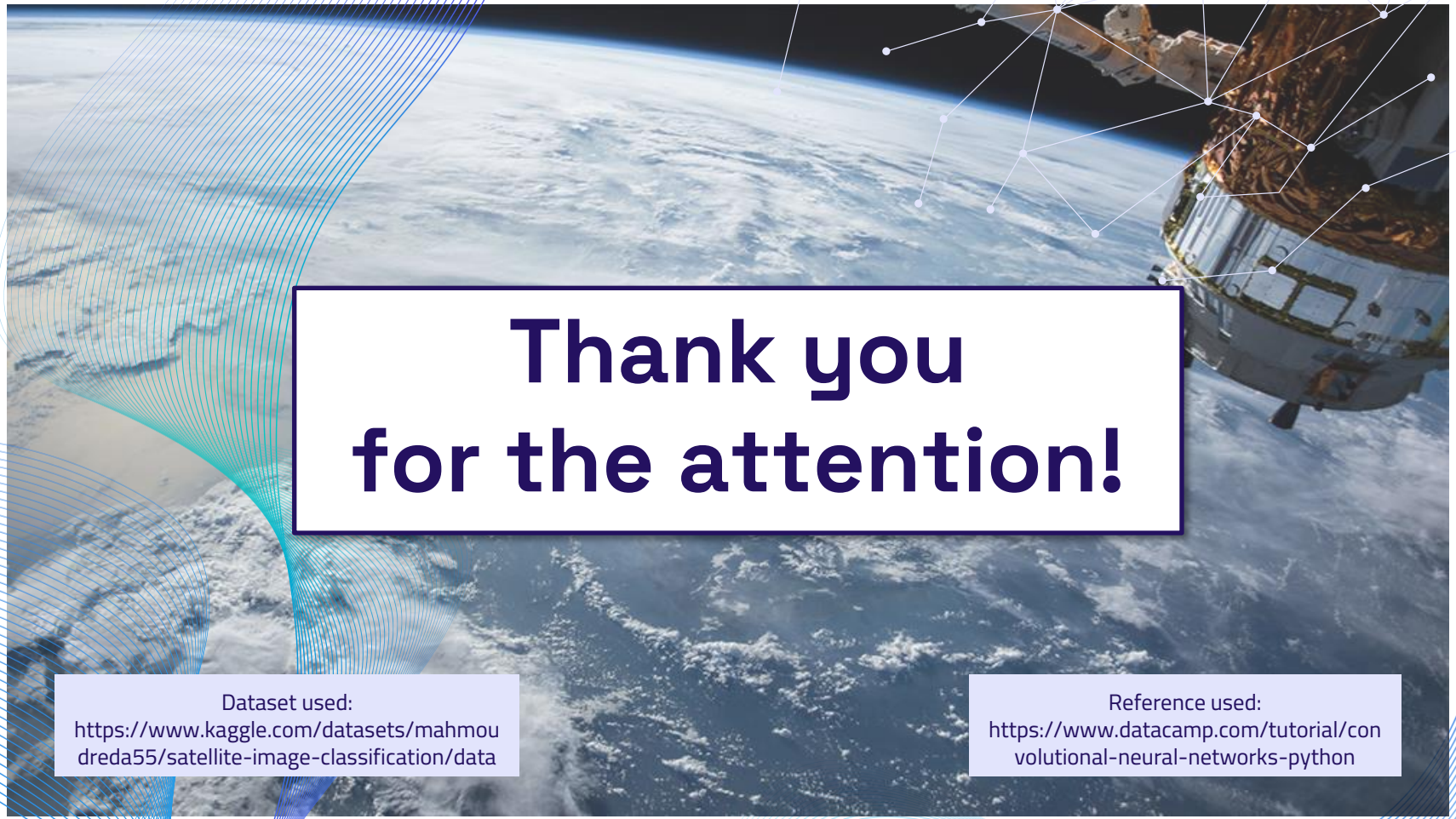




- Next, we used the trained model to generate predictions on randomly selected images from the dataset
- All the predictions so far were correct

- Lastly, we constructed a confusion matrix to test the accuracy of the predictions on the entire dataset.





Thank you for the attention!

Dataset used:
https://www.kaggle.com/datasets/mahmoudreda55/satellite-image-classification/data

Reference used:
https://www.datacamp.com/tutorial/convolutional-neural-networks-python