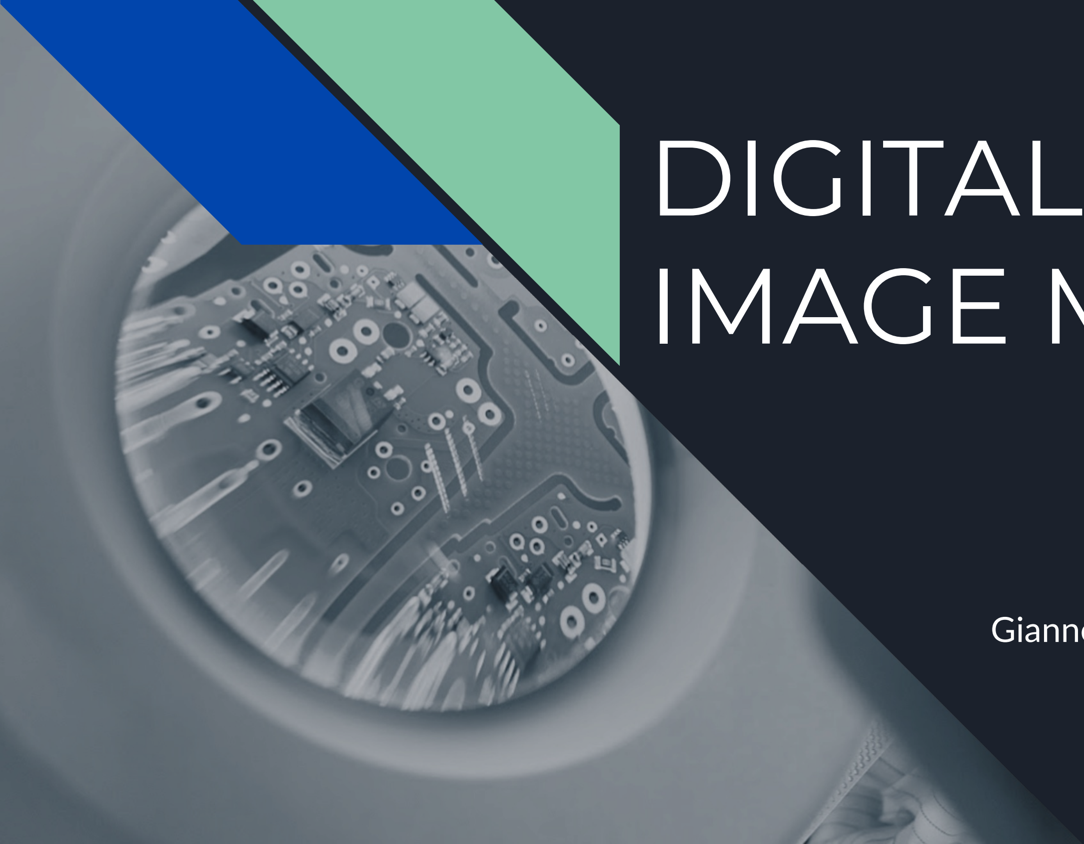


DIGITAL SIGNAL & IMAGE MANAGEMENT

Giannelli Alessio Imbonati Lorenzo Valoti Davide

SUMMARY

Step 1: Image Classification -

FGVC Aircraft 100 Dataset

using Densenet201

Step 2: Image Retrieval -

Feature Re-Weighting in CBIR

Dataset & Pre-Processing

Info about Dataset

Dataset Size: 2.76 GB

Number of classes: 100

Number of Instances per class: 100

Splitting

TRAIN (60%)	VALIDATION (30%)	TEST (10%)
6000 instances	3000 instances	1000 instances

Cropping

Model Architecture

Transfer Architecture DenseNet201

```python
base_net = keras.applications.DenseNet201(input_shape=(224,224,3),weights='imagenet', include_top=False)
base_net.trainable = True
```

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, 224, 224, 3)]	0
tf.math.truediv_2 (TFOpLambda)	(None, 224, 224, 3)	0
tf.nn.bias_add_1 (TFOpLambda)	(None, 224, 224, 3)	0
tf.math.truediv_3 (TFOpLambda)	(None, 224, 224, 3)	0
densenet201 (Functional)	(None, 7, 7, 1920)	18321984
average_pooling2d_1 (AveragePooling2D)	(None, 3, 3, 1920)	0
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 1920)	0
dense_3 (Dense)	(None, 1024)	1967104
batch_normalization_2 (BatchNormalization)	(None, 1024)	4096
dropout_2 (Dropout)	(None, 1024)	0
dense_4 (Dense)	(None, 512)	524800
batch_normalization_3 (BatchNormalization)	(None, 512)	2048
dropout_3 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 100)	51300

```
Total params: 20,871,332
Trainable params: 20,639,204
Non-trainable params: 232,128
```

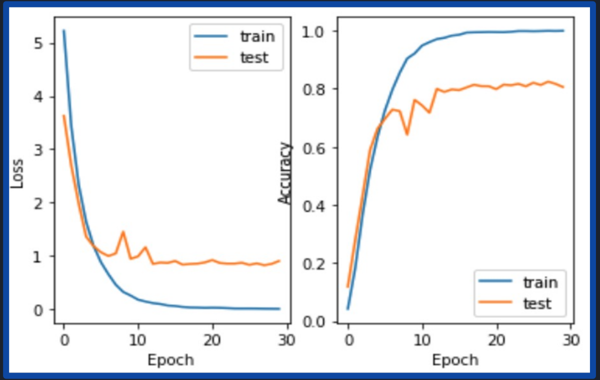
Transfer Learning

DenseNet201 Performance

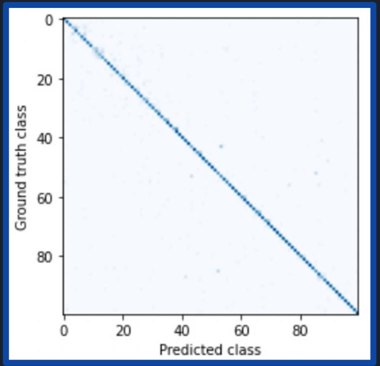
Classification Report

	precision	recall	f1-score
0	0.86	0.97	0.91
1	0.93	0.85	0.89
2	0.96	0.74	0.83
19	0.65	0.52	0.58
20	0.73	0.94	0.82
21	0.65	0.73	0.69
22	0.82	0.70	0.75
23	0.97	0.91	0.94
accuracy			0.81
macro avg	0.82	0.81	0.80
weighted avg	0.82	0.81	0.80

Loss & Accuracy Trend



Confusion Matrix



The performance of the model is very good as it achieves 80% accuracy on the validation set.

The trend shows some performance fluctuations but in general it is quite stable and the validation follows the growth of the training set.

DenseNet201 Evaluation

on Test Set

Test loss	Test accuracy
0.8971	80.30%

on Web Image

Distribution Probability

```
previsione

array([[8.45307895e-08, 8.59294147e-09, 2.28450489e-07, 1.04331457e-05,
        5.10848849e-07, 9.47635385e-07, 9.56761141e-08, 1.16615745e-04,
        1.19382069e-01, 6.69421115e-06, 1.71462332e-07, 1.86927124e-07,
        3.83096435e-08, 1.61001561e-04, 8.33525717e-01, 1.18540612e-03,
```

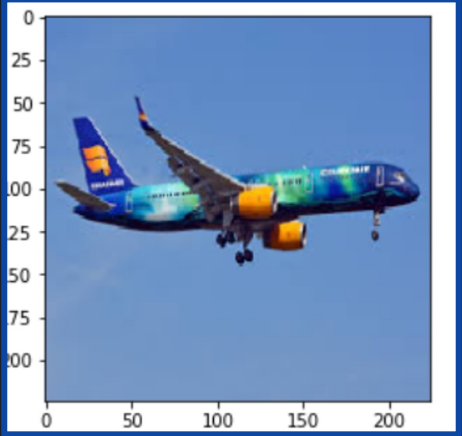
Image Prevision

```
np.argmax(previsione)

14
```

```
train.class_names[14]

'757-200'
```



Feature Re-Weighting in CBIR

Implementation of the following paper

Feature Re-weighting in Content-Based Image Retrieval

Gita Das[1], Sid Ray[1], and Campbell Wilson[2]

[1] Clayton School of Information Technology
Monash University
Victoria 3800, Australia
{Gita.Das, Sid.Ray}@csse.monash.edu.au
[2] Caulfield School of Information Technology
Monash University
Victoria 3800, Australia
Campbell.Wilson@csse.monash.edu.au

Main concepts:

- Use of the previous neural network as feature extractor
- Use of weighted Minkowski distance as similarity measure
- Update of the query results according to user preferences

Feature Extraction

Load Task2 Model

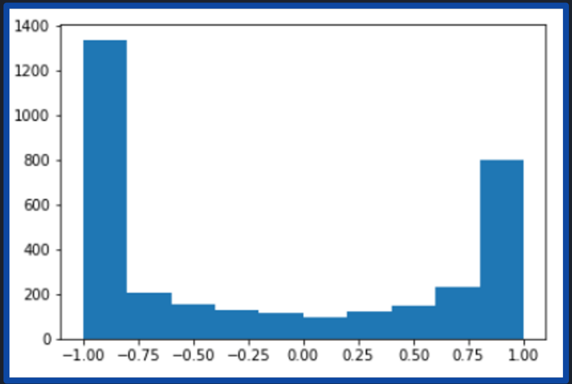
```python
temp = keras.models.load_model('Model/densenet201_final_task2.h5')

layer_name = 'dense_1'
newmodel = Model(inputs=temp.input, outputs=temp.get_layer(layer_name).output)
newmodel.summary()
```

Splitting

TRAIN
6000 instances

TEST
1000 instances



Features normalization

$$f_i' = \frac{f_{i,org} - \mu_i}{3\sigma_i}$$

$$f_i = \frac{f_i' + 1}{2}$$

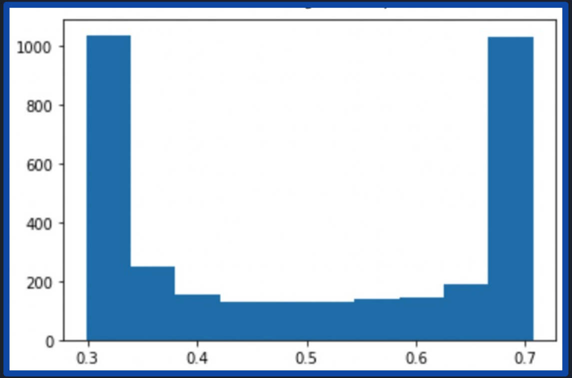


Image Retrieval (Query)

Manhattan similarity measure

$$D(I,Q) = \sum_{i=1}^{M} w_i * |f_{iI} - f_{iQ}|$$

Weights are constant for the first round of retrieval

Top20 Precision on test set
77,56%

Images Similarity to the test image

Rebalancing type 1

Update weights formula Type 1

$$weight - type1 : w_i^{k+1} = \frac{\epsilon + \sigma^k_{N_r,i}}{\epsilon + \sigma^k_{rel,i}}, \epsilon = 0.0001$$

New weight for the i-th feature is equal to the division between the standard deviation over the 20 retrieved images and the standard deviation over the relevant images at the previous round

$w^{k+1} = 0.9 * w^k + 0.1 * w^{k+1}$

Round number	Top20 Precision
Round 0	77.56
Round 1	83.94
Round 2	84.56
Round 3	85.10
Round 4	85.41
Round 5	**85.54**

Rebalancing type 2

Update weights formula Type 2

$$weight - type2 : w_i^{k+1} = \frac{\delta_i^k}{\epsilon + \sigma_{rel,i}^k}$$

$$\delta_i^k = 1 - \frac{\sum_{l=1}^{k} |\psi_i^{l,U}|}{\sum_{l=1}^{k} |F_i^{l,U}|}$$

New weight for the i-th feature is equal to the division between the sigma quantity defined in the second formula, that depends on the **dominant range,** and the standard deviation over the relevant images at the previous round

$w^{k+1} = 0.9*w^k + 0.1*w^{k+1}$

Round number	Top20 Precision
Round 0	**77.56**
Round 1	61.70
Round 2	58.84
Round 3	59.91
Round 4	60.09
Round 5	60.53

Rebalancing type 3

Update weights formula Type 3

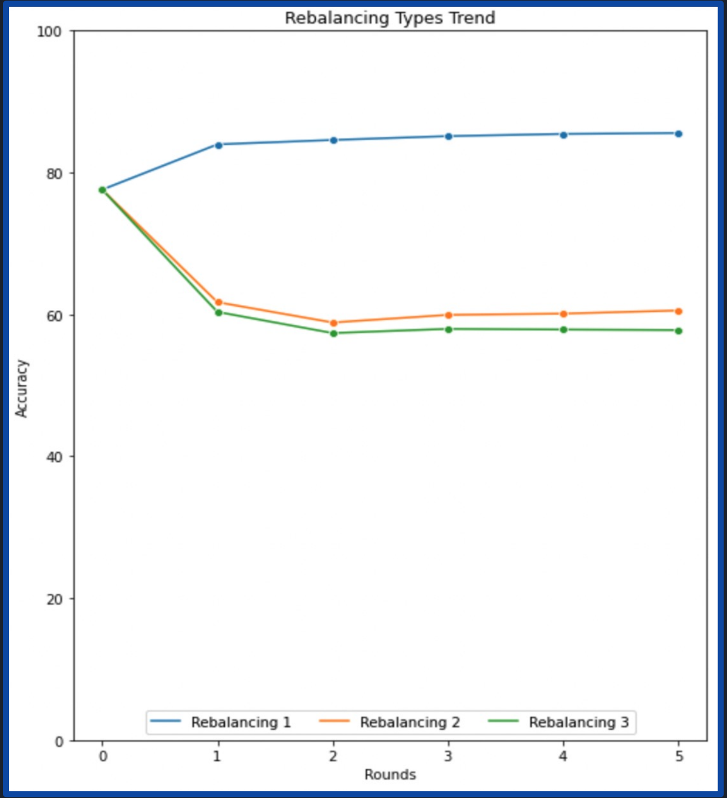
$$weight - type3 : w_i^{k+1} = \delta_i^k * \frac{\epsilon + \sigma_{N_r,i}^k}{\epsilon + \sigma_{rel,i}^k}$$

New weight for the i-th feature is equal to the the delta value defined in the previous slide by the weights of type 1

$$w^{k+1} = 0.9*w^k + 0.1*w^{k+1}$$

Round number	Top20 Precision
Round 0	**77.56**
Round 1	60.33
Round 2	57.35
Round 3	57.94
Round 4	57.85
Round 5	57.77

Rebalancing Types Trend



Type 1 rebalancing is definitely the best since it shows increasing growth.

The other two types of rebalancing do not produce any improvement.

Let's leave room for the demo...