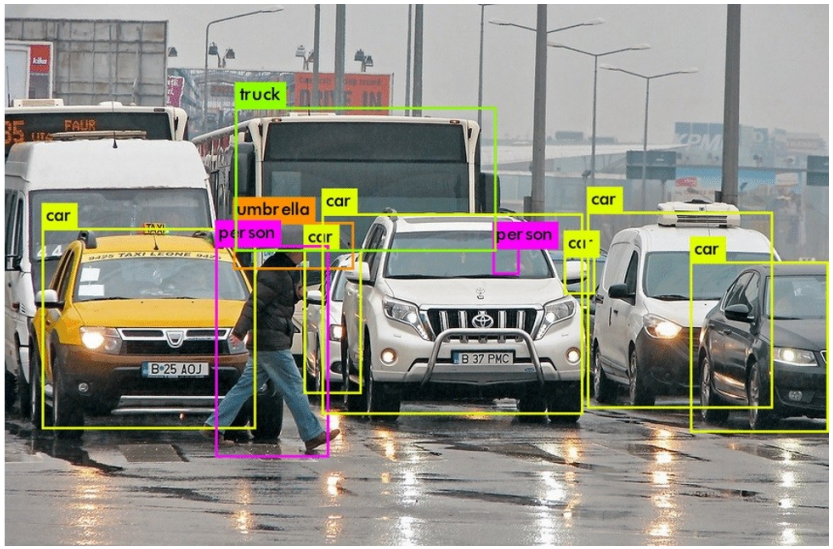
The usefulness of the VGG method in the problem of image classification

Matteo Di Stadio - 1794111
Marco Carfora - 1794568

SAPIENZA
UNIVERSITÀ DI ROMA

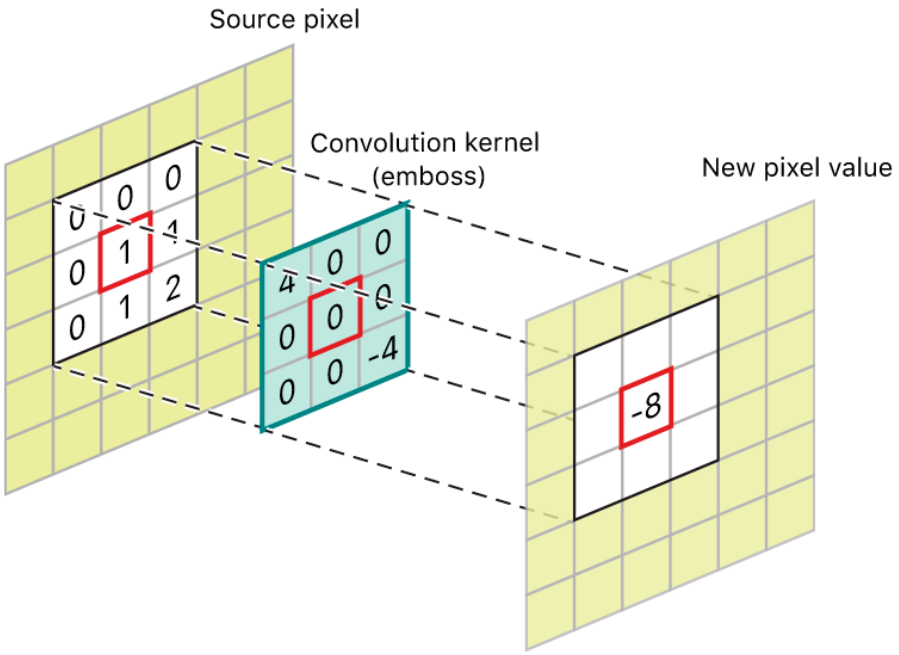
The problem of image recognition

- Humans take action based on the study of their surroundings, especially through vision.
- Computer vision is having the machine recognize the patterns of an object in order to discern what it can or is required to do in the given situation.
- Neural networks would be the most adequate available technique.



Convolutional neural networks

- Images can be seen as 3D tensors, leaving the convolutional neural networks as the best type of nets to deal with them.
- CNN make use of the mathematical operation of convolution, activation functions and aggregate functions to gain and exploit the main features of the tensor.
- ImageNet is a large visual database designed for use in visual object recognition software research, which helped defining many famous CNN models for computer vision.



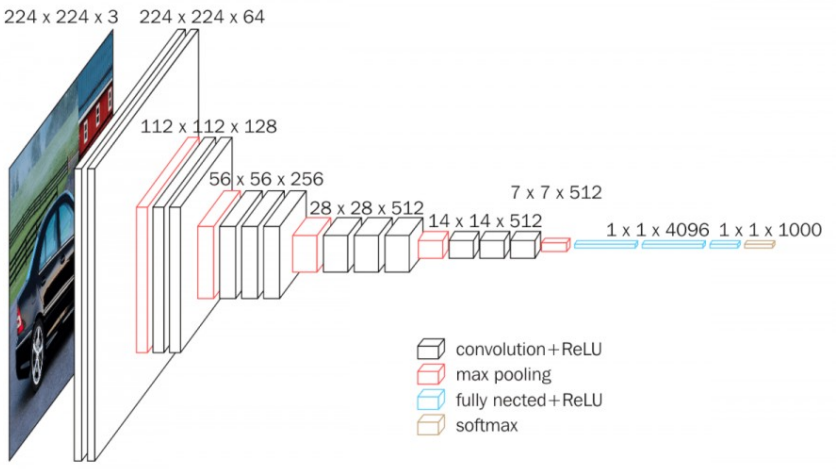
VGG method

- The VGG network architecture was first described in the 2014 paper ”Very Deep Convolutional Networks for Large Scale Image Recognition”
- A very simple architecture as it uses only 3x3 convolutional layers, followed by max pooling ones. At the end it also adds three fully connected layers.
- Six possible variations, all following the same basic structure, with the main difference being in their depths.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG16 and VGG19

- Based on experimental and practical results, the most famous and used variants are the 16 and 19 one.
- Both VGG16 and VGG19 retain the same basic structure of the VGG method.
- The main difference lies in the weighted layers number, being a total of 16 and 19 respectively, but also in the filter sizes and number of times the ReLU activation function gets used.



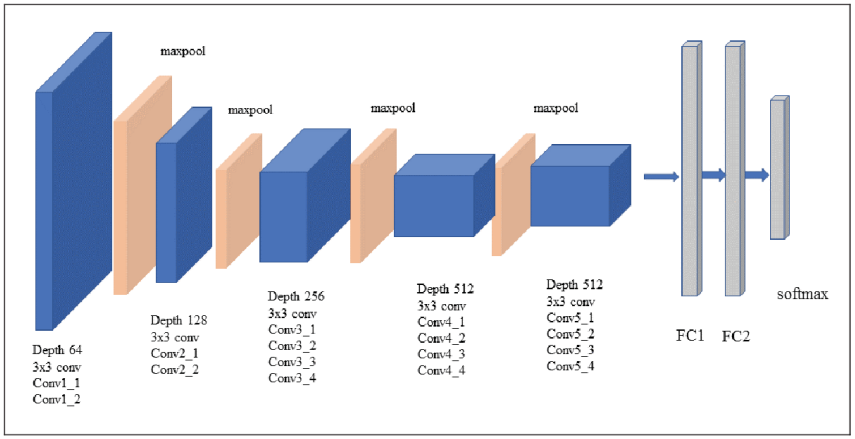


Fig. 3. VGG-19 network architecture

The dataset

- ARGOS is a surveilance system operating 24/7 in the city of Venice.
- Venice provides an incomparable environment, characterized by unique and challenging conditions like waves and reflections. Training on these terms can widely help with the study of intelligent surveillance systems.
- The gathered images are then split up in testing and validation sets.

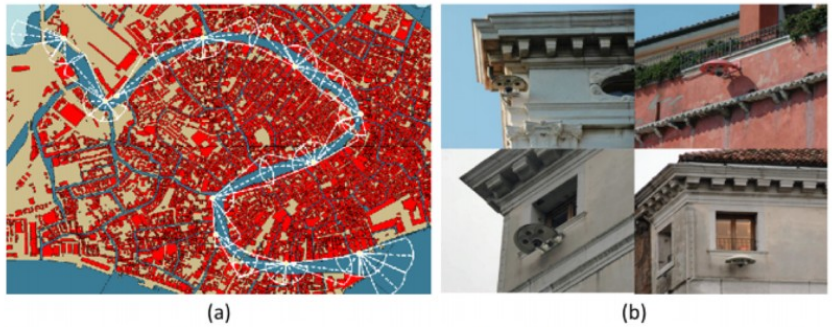


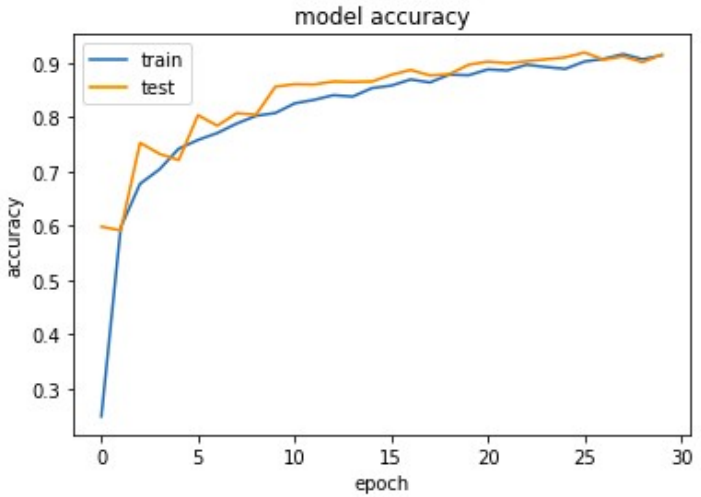
Figure 2. a) The waterway monitored by ARGOS system. b) The cell installed below the roof of buildings.

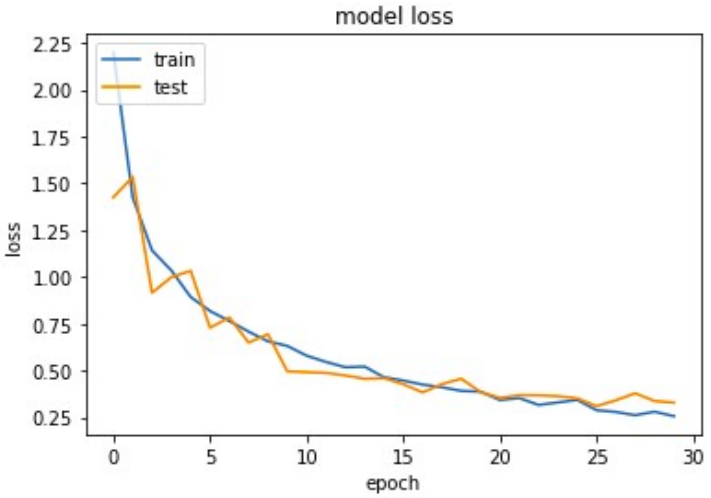
Figure 3. Automatic snapshot taken by the ARGOS system.

Metrics

- With the need of training different nets using different parameters and strategies, a way to reliably confront them all was needed.

- Accuracy
- Loss
- Precision
- Recall
- F1-Score
- Confusion Matrix





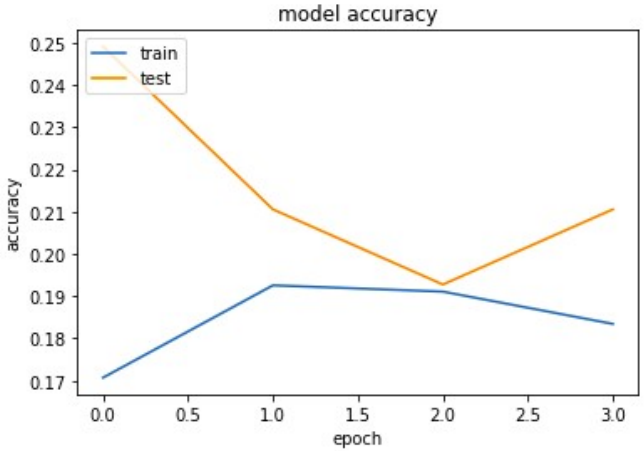
VGG16 – first experiment

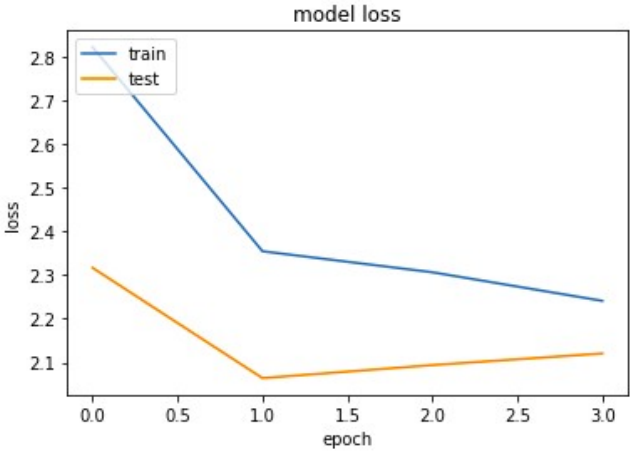
- First VGG16 was implemented following the specific directions found in the paper.
- An Adam optimizer was then applied. Chosen for its suitability for image classifiers, it updates the network weights iteratively using different rates for each parameter.
- An Early Stopping callback technique was also implemented to save on resources when it is needed and possible.

```
[[  0   0   0   0   0   0   0 420   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  19   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  22   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  51   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 355   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  59   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 284   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  74   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  15   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  19   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0  29   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0 325   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]]
```

VGG16 – first experiment results

	precision	recall	f1-score	support
Acqua	0.000	0.000	0.000	420
Alilaguna	0.000	0.000	0.000	19
Ambulanza	0.000	0.000	0.000	22
Barchino	0.000	0.000	0.000	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	0.000	0.000	0.000	3
Lancia	0.211	1.000	0.348	355
Motobarca	0.000	0.000	0.000	59
Motopontonerettangolare	0.000	0.000	0.000	3
MotoscafoACTV	0.000	0.000	0.000	1
Mototopo	0.000	0.000	0.000	284
Patanella	0.000	0.000	0.000	74
Polizia	0.000	0.000	0.000	15
Raccoltarifiuti	0.000	0.000	0.000	19
Sandoloaremi	0.000	0.000	0.000	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.000	0.000	0.000	29
VaporettoACTV	0.000	0.000	0.000	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.211	1686
macro avg	0.011	0.050	0.017	1686
weighted avg	0.044	0.211	0.073	1686





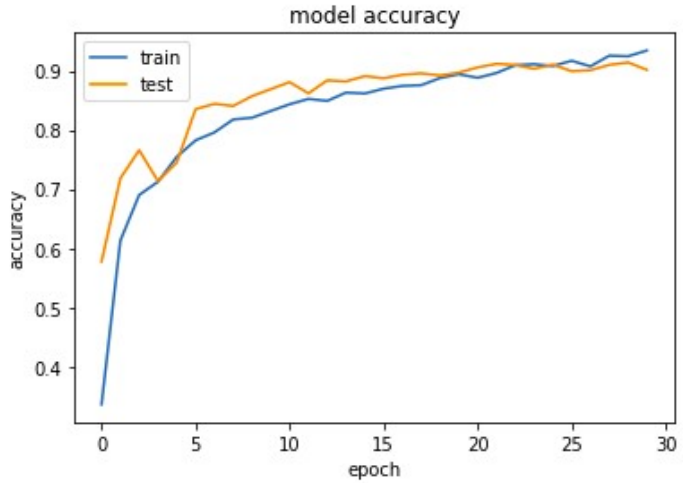
VGG16 – second experiment

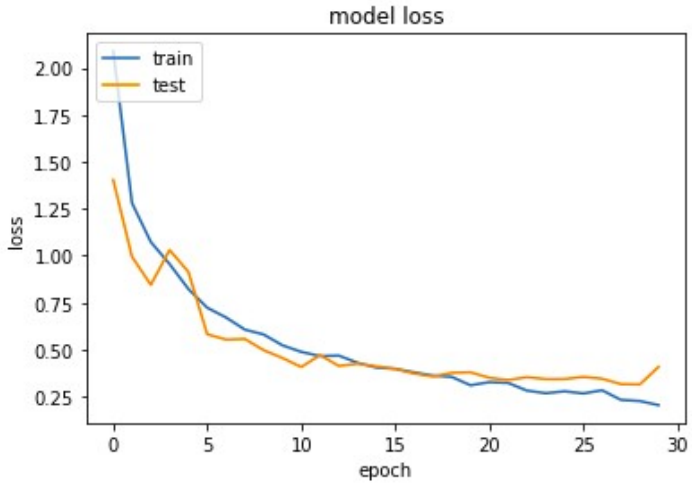
- The results of the first experiment are quite lackluster.
- A possible explaination can be seen in the dataset lack of images among the two sets.
- To compensate it was then introduced the concept of data augmentation.

```
[[414   0   0   0   0   0   0   2   2   0   0   0   1   0   0   0   0   0   1   0]
 [  0  17   1   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  0   0  20   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   1]
 [  3   0   0  18   2   0   0   7   0   0   0   0  18   0   0   0   1   1   0   1]
 [  0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0]
 [  2   0   1   0   0   0   0 345   2   0   0   0   2   2   0   0   0   0   0   1]
 [  0   0   2   0   0   0   0   5  39   0   0   7   2   1   1   0   0   1   0   1]
 [  0   0   0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  1   0   7   0   0   0   1   4  22   0   0 244   3   0   0   0   0   1   1   0]
 [  1   0   0   3   0   0   0   6   0   0   0   0  61   0   0   0   0   2   0   1]
 [  0   0   1   1   0   0   0   4   0   0   0   0   0   9   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   1   0   0   0   0   0  17   0   0   0   0   0]
 [  0   0   0   1   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  2   0   0   4   0   0   0   3   0   0   0   0  14   0   0   0   0   6   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 325   0]
 [  0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

VGG16 – second experiment results

	precision	recall	f1-score	support
Acqua	0.979	0.986	0.982	420
Alilaguna	1.000	0.895	0.944	19
Ambulanza	0.606	0.909	0.727	22
Barchino	0.621	0.353	0.450	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	0.333	0.333	0.333	3
Lancia	0.910	0.972	0.940	355
Motobarca	0.582	0.661	0.619	59
Motopontonerettangolare	1.000	1.000	1.000	3
MotoscafoACTV	0.500	1.000	0.667	1
Mototopo	0.968	0.859	0.910	284
Patanella	0.598	0.824	0.693	74
Polizia	0.750	0.600	0.667	15
Raccoltarifiuti	0.944	0.895	0.919	19
Sandoloaremi	0.000	0.000	0.000	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.545	0.207	0.300	29
VaporettoACTV	0.994	1.000	0.997	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.902	1686
macro avg	0.567	0.575	0.557	1686
weighted avg	0.904	0.902	0.898	1686





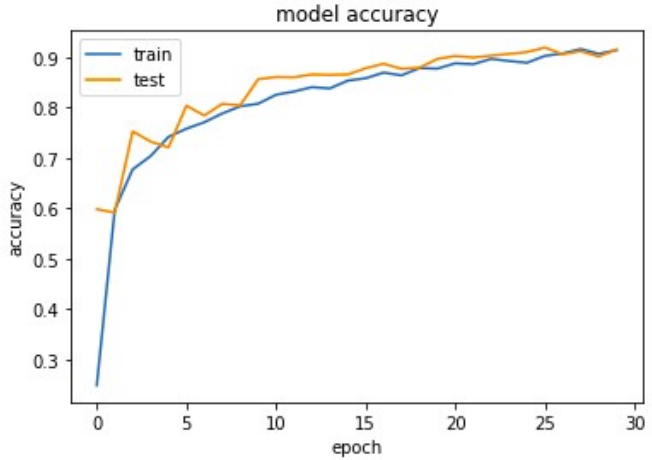
VGG16 – third experiment

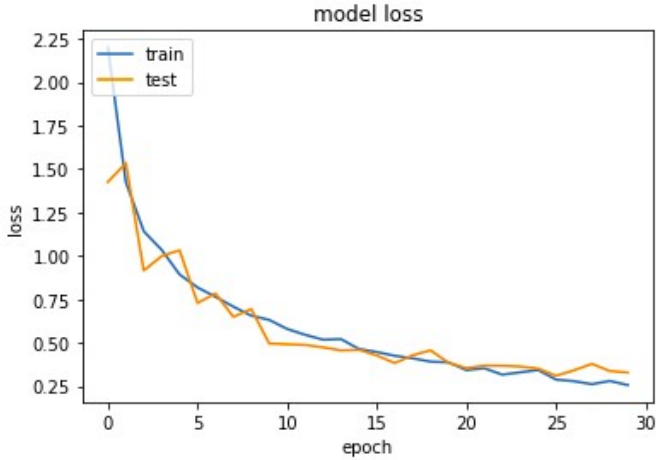
- More optimizations are possible to get even better outcomes.
- An improvement can be given by Dropout, a cheap and effective regularization method to reduce overfitting and improve generalization error.
- The idea is to use a single model to simulate having a large number of different network architectures by temporarily randomly dropping out nodes during training.

```
[[414   0   0   2   0   0   0   2   0   0   0   0   1   0   0   0   0   0   1   0]
 [  0  18   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0  18   0   0   0   0   1   0   0   0   0   2   1   0   0   0   0   0   0]
 [  1   0   0  22   0   0   0   2   0   0   0   0  22   0   0   0   0   4   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   1   0   0   0   0   0   1   0   0]
 [  4   0   1   0   1   0   0 334   1   0   0   0  12   2   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1  28   0   0  14  11   0   5   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  1   0   0   0   0   0   0   1   1   0   0 278   1   0   0   0   0   1   1   0]
 [  0   0   0   2   0   0   0   0   1   0   0   0  71   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   2   0   0   0   0   4   8   0   0   0   0   0   0]
 [  1   0   0   0   0   0   0   0   1   0   0   0   0   0  17   0   0   0   0   0]
 [  0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   2   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   5   0   0   0   0   0   0   0   2  14   0   0   1   0   7   0   0]
 [  1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 324   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]]
```

VGG16 – third experiment results

	precision	recall	f1-score	support
Acqua	0.981	0.986	0.983	420
Alilaguna	1.000	0.947	0.973	19
Ambulanza	0.947	0.818	0.878	22
Barchino	0.667	0.431	0.524	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	0.000	0.000	0.000	3
Lancia	0.974	0.941	0.957	355
Motobarca	0.848	0.475	0.609	59
Motopontonerettangolare	1.000	1.000	1.000	3
MotoscafoACTV	1.000	1.000	1.000	1
Mototopo	0.939	0.979	0.959	284
Patanella	0.504	0.959	0.660	74
Polizia	0.727	0.533	0.615	15
Raccoltarifiuti	0.773	0.895	0.829	19
Sandoloaremi	0.000	0.000	0.000	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.467	0.241	0.318	29
VaporettoACTV	0.994	0.997	0.995	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.915	1686
macro avg	0.591	0.560	0.565	1686
weighted avg	0.920	0.915	0.911	1686





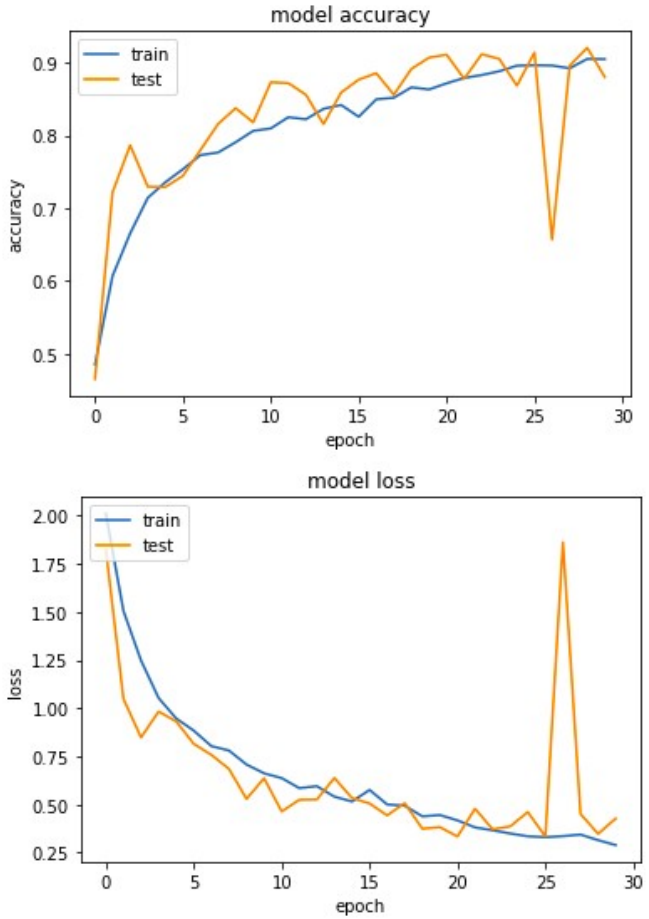
VGG16 – fourth experiment

- One can beg the question if something more can still be applied.
- Batch Normalization is a possible further improvement. A technique that was actually invented after the initial formulation of VGG.
- Batch Normalization standardizes the inputs to a layer for each mini batch, stabilizing the learning task and reducing the number of training epochs.

```
[[412   0   0   1   3   0   0   2   0   0   0   0   1   0   0   0   0   0   1   0]
 [  0  19   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  1   0  19   0   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0]
 [  0   0   0  36   0   0   0   1   0   0   0   0  13   1   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   0   1   0   0   1   0   0   0   0]
 [  3   0   0   1   0   0   0 338   1   0   0   0   2  10   0   0   0   0   0   0]
 [  0   0   0   0   1   0   0   7  37   0   0   4   7   1   1   1   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  1   0   5   1   0   0   0   5  28   0   0 220  18   1   1   2   0   2   0   0]
 [  0   0   0  12   1   0   0   2   0   0   0   0  59   0   0   0   0   0   0   0]
 [  0   0   0   0   1   0   0   2   0   0   0   0   0  12   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   1   3   0   0   0   0   0  14   0   0   0   0   0]
 [  0   0   0   2   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   8   1   0   0   1   1   0   0   0  15   0   0   1   0   2   0   0]
 [  0   4   0   0   0   0   0   2   8   0   0   0   0   0   0   0   0   0 311   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0]]
```

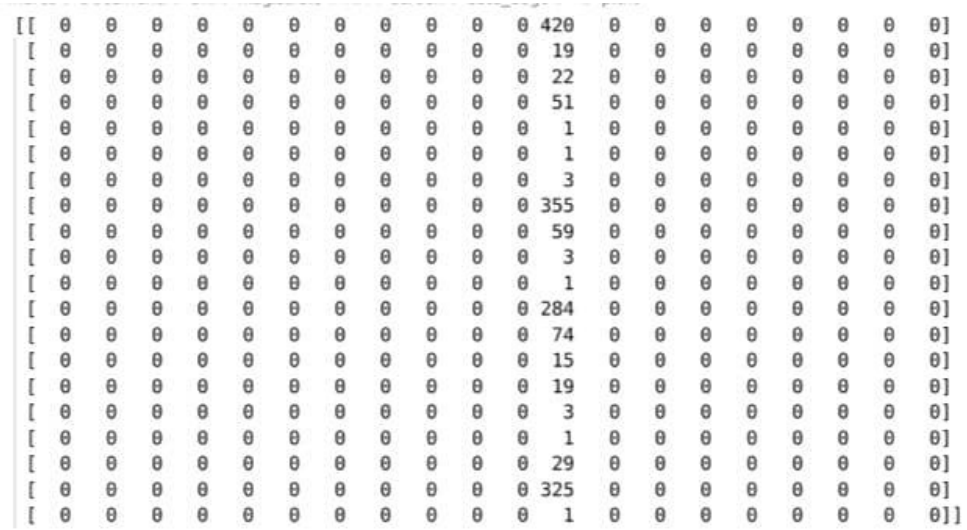
VGG16 – fourth experiment results

	precision	recall	f1-score	support
Acqua	0.988	0.981	0.984	420
Alilaguna	0.826	1.000	0.905	19
Ambulanza	0.792	0.864	0.826	22
Barchino	0.562	0.706	0.626	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	0.000	0.000	0.000	3
Lancia	0.929	0.952	0.940	355
Motobarca	0.474	0.627	0.540	59
Motopontonerettangolare	1.000	1.000	1.000	3
MotoscafoACTV	1.000	1.000	1.000	1
Mototopo	0.982	0.775	0.866	284
Patanella	0.509	0.797	0.621	74
Polizia	0.444	0.800	0.571	15
Raccoltarifiuti	0.875	0.737	0.800	19
Sandoloaremi	0.167	0.333	0.222	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.500	0.069	0.121	29
VaporettoACTV	0.997	0.957	0.976	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.880	1686
macro avg	0.552	0.580	0.550	1686
weighted avg	0.900	0.880	0.882	1686



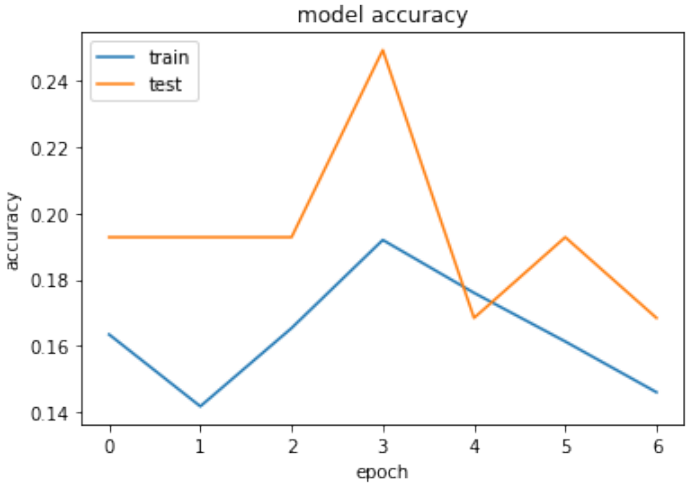
VGG19 – first experiment

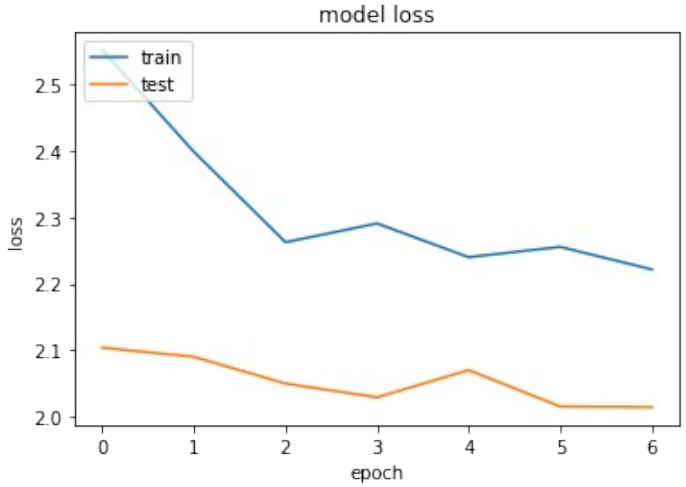
- Considering VGG19 instead, the modalities are almost similar to the ones we previously saw.
- We make again use of both the Adam optimizer and the Early Stopping callback, but we also implement Dropout from the start this time around.
- Like before, the first experiment replicates the standard formulation of VGG19.



VGG19 – first experiment results

	precision	recall	f1-score	support
Acqua	0.000	0.000	0.000	420
Alilaguna	0.000	0.000	0.000	19
Ambulanza	0.000	0.000	0.000	22
Barchino	0.000	0.000	0.000	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	0.000	0.000	0.000	3
Lancia	0.000	0.000	0.000	355
Motobarca	0.000	0.000	0.000	59
Motopontonerettangolare	0.000	0.000	0.000	3
MotoscafoACTV	0.000	0.000	0.000	1
Mototopo	0.168	1.000	0.288	284
Patanella	0.000	0.000	0.000	74
Polizia	0.000	0.000	0.000	15
Raccoltarifiuti	0.000	0.000	0.000	19
Sandoloaremi	0.000	0.000	0.000	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.000	0.000	0.000	29
VaporettoACTV	0.000	0.000	0.000	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.168	1686
macro avg	0.008	0.050	0.014	1686
weighted avg	0.028	0.168	0.049	1686





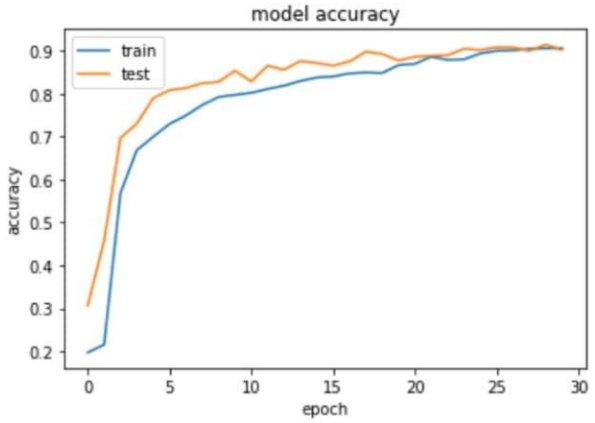
VGG19 – second experiment

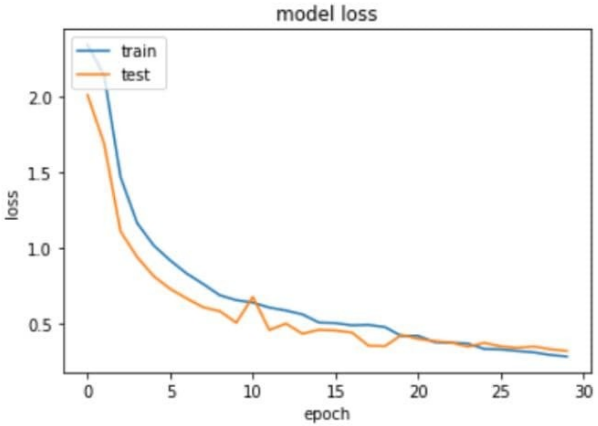
- Just like what we did with VGG16 we can now add a good level of data augmentation in order to gain better results.

```
[[414   0   0   1   0   0   0   2   1   0   0   0   1   0   0   0   0   0   1   0]
 [  0  17   1   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  0   1  14   0   0   0   0   0   1   0   0   1   0   0   0   0   0   0   0   5]
 [  2   0   0  24   3   0   0   2   1   0   0   0  13   1   0   0   0   5   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   0]
 [  4   0   0   0   0   0   0 339   3   0   0   0   3   5   0   0   0   0   0   1]
 [  0   1   0   0   0   0   0   3  32   0   0   8   6   1   8   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   2   0   0   0   0   1   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0  24   0   0   0 254   4   0   0   0   0   1   1   0]
 [  0   0   0  11   1   0   0   1   0   0   0   0   0  58   1   0   0   0   2   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   1   0  13   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0  18   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   1   0   0   0   0   1   0   0   0   0   1   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  1   0   0   2   0   0   0   1   1   0   0   1  13   0   0   0   0  10   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0 324   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]]
```

VGG19 – second experiment results

	precision	recall	f1-score	support
Acqua	0.983	0.986	0.985	420
Alilaguna	0.895	0.895	0.895	19
Ambulanza	0.933	0.636	0.757	22
Barchino	0.600	0.471	0.527	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	1.000	0.333	0.500	3
Lancia	0.969	0.955	0.962	355
Motobarca	0.500	0.542	0.520	59
Motopontonerettangolare	1.000	0.667	0.800	3
MotoscafoACTV	0.500	1.000	0.667	1
Mototopo	0.958	0.894	0.925	284
Patanella	0.563	0.784	0.655	74
Polizia	0.619	0.867	0.722	15
Raccoltarifiuti	0.667	0.947	0.783	19
Sandoloaremi	0.000	0.000	0.000	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.500	0.345	0.408	29
VaporettoACTV	0.994	0.997	0.995	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.902	1686
macro avg	0.584	0.566	0.555	1686
weighted avg	0.910	0.902	0.903	1686





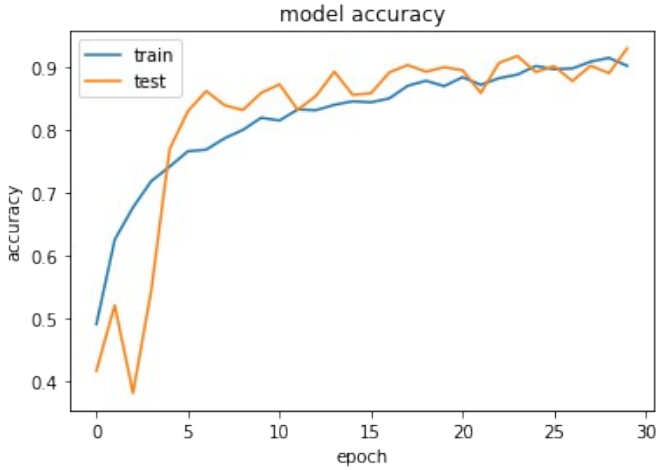
VGG19 – third experiment

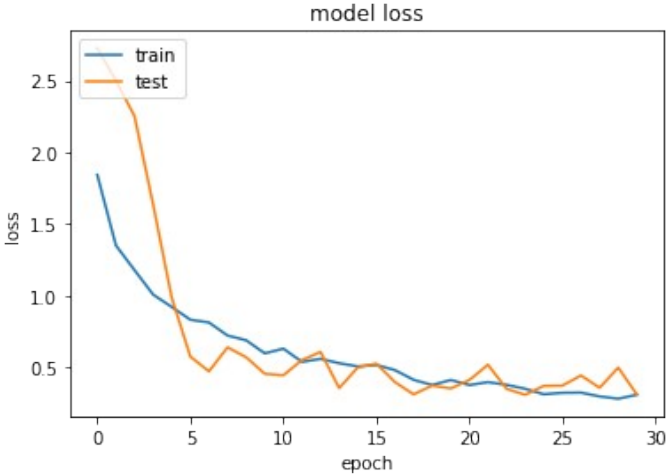
- We then try to unite the VGG19 method with the Batch Normalization, providing the same specifics as before.
- We can see that it finds itself really confortable if paired with VGG19.
- These results are actually the best we have gotten.

```
[[413   0   0   2   0   0   0   2   0   0   0   0   2   0   0   0   0   0   1   0]
 [  0  19   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0  22   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  1   0   0  26   3   0   0   0   0   0   0   0  13   0   0   0   0   8   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0]
 [  3   0   0   1   0   0   0 339   1   0   0   0   5   5   1   0   0   0   0   0]
 [  1   0   1   0   0   0   0   2  34   0   0  12   8   0   0   0   0   0   1   0]
 [  0   0   0   0   0   0   0   0   0   3   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   3   0   0   0 276   4   0   0   0   0   1   0   0]
 [  0   0   0   8   0   0   0   1   0   0   0   0  62   2   0   0   0   1   0   0]
 [  0   0   0   0   0   0   0   3   0   0   0   0   1  11   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   0   2   0   0   1   0   0  15   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   1   0   1   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0]
 [  1   0   0   2   0   0   0   0   0   0   0   2   3   0   0   0   0  21   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 325   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0]]
```

VGG19 – third experiment results

Acqua	0.986	0.983	0.985	420
Alilaguna	1.000	1.000	1.000	19
Ambulanza	0.957	1.000	0.978	22
Barchino	0.634	0.510	0.565	51
Cacciapesca	0.000	0.000	0.000	1
Caorlina	0.000	0.000	0.000	1
Gondola	1.000	0.333	0.500	3
Lancia	0.969	0.955	0.962	355
Motobarca	0.919	0.576	0.708	59
Motopontonerettangolare	1.000	1.000	1.000	3
MotoscafoACTV	1.000	1.000	1.000	1
Mototopo	0.945	0.972	0.958	284
Patanella	0.614	0.838	0.709	74
Polizia	0.579	0.733	0.647	15
Raccoltarifiuti	0.938	0.789	0.857	19
Sandoloaremi	1.000	0.333	0.500	3
Sanpierota	0.000	0.000	0.000	1
Topa	0.636	0.724	0.677	29
VaporettoACTV	0.994	1.000	0.997	325
VigilidelFuoco	0.000	0.000	0.000	1
accuracy			0.931	1686
macro avg	0.708	0.637	0.652	1686
weighted avg	0.935	0.931	0.930	1686





Conclusions

- VGG16 with the right conditions can still perform really well even on the tougher conditions of ARGOS.
- VGG19 presents the best overall outcomes considering the same initial conditions. It can also work better with Batch Normalization.
- We managed also to highlight the importance of data augmentation and other optimizers and tricks like Adam, Dropout and Early Stopping in the computer vision environment.