Assignment 4
Advanced Machine Learning
University of Milano-Bicocca
M.Sc. Data Science

Federico Signoretta
ID number: 847343
f.signoretta@campus.unimib.it
November 25, 2019

Transfer Learning using a CNN pretrained on IMAGENET

The task of this assignment is Transfer Learning using a CNN pre-trained on IMAGENET. The suggested architecture is the VGG16, since AlexNet is not available in recent Keras versions. The CNN should be used as fixed feature extractor on a new task of your choice containing a number of classes in the range from 3 to 10.

1 Dataset

The images used to solve this task belong to the dataset called Intel Image Classification. This Data contains around 25k images of size 150x150 distributed under 6 categories. In particular, **buildings**: 0, **forest**: 1, **glacier**: 2, **mountain**: 3, **sea**: 4, **street**: 5.
The Train, Test and Prediction data is separated in each zip files. There are around 14k images in Train, 3k in Test and 7k in Prediction. This data was initially published on *datahack.analyticsvidhya.com* by Intel to host a Image classification Challenge.
For this scope, it was considered a sample of these Data: 1,998 images from the Train and 996 from the Test, with equal number of images for each class.

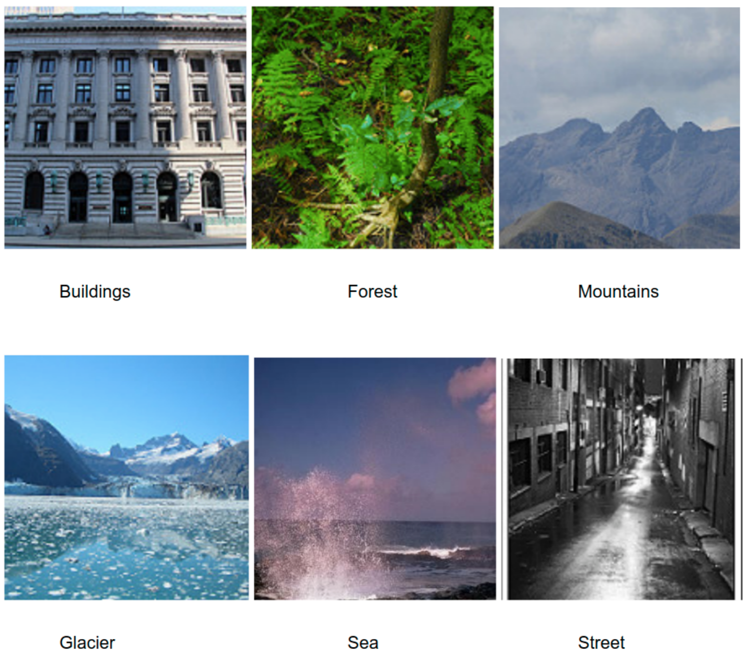


Figure 1: Example from the Intel Images Classification

2 Transfer Learning architecture

In order to solve this task it was used the Transfer Learning method. With transfer learning, instead of starting the learning process from scratch, it is possible to start from patterns that have been learned when solving a different problem. In computer vision, transfer learning is usually expressed through the use of pre-trained models (a model that was trained on a large benchmark dataset to solve a problem similar to the one that we want to solve). In this case, it was used the **VGG16**: is a CNN model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper *"Very Deep Convolutional Networks for Large-Scale Image Recognition"*. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. The main idea is to keep the VGG16 in its

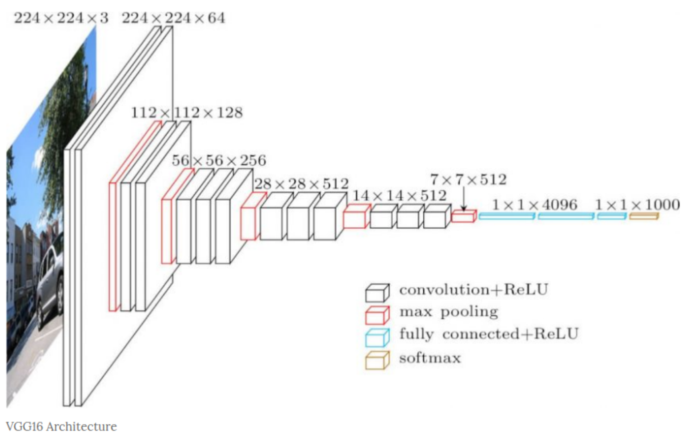


Figure 2: Example from the Intel Images Classification

original form and then use its outputs to feed the classifier. Using the pre-trained model as a fixed feature extraction mechanism it is possible to handle situation like short on computational power, small dataset, and/or pre-trained model solves a problem very similar to the one you want to solve.
In this experiment, it was used the **Logistic Regression** as classifier with the following parameters:

- solver='newton-cg'
- multi_class='multinomial'
- max_iter=200

3 Description of the experiments

Through the above "tools" above shown, several experiments were performed. In particular, it was cut the VGG16 on three different layer in order to use it as feature extractor:

1. First experiment: VGG16 up to block4_pool (MaxPooling2D)
2. Second experiment: VGG16 up to block5_pool (MaxPooling2D)
3. Third experiment: VGG16 up to fc1 (Dense)

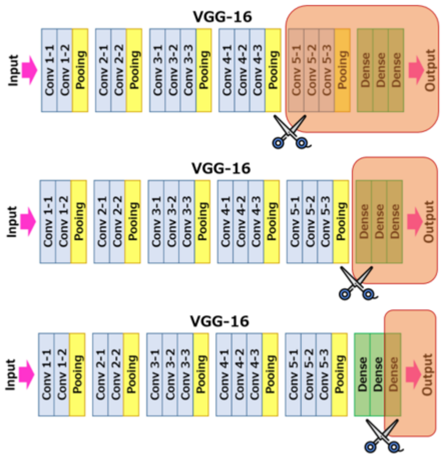
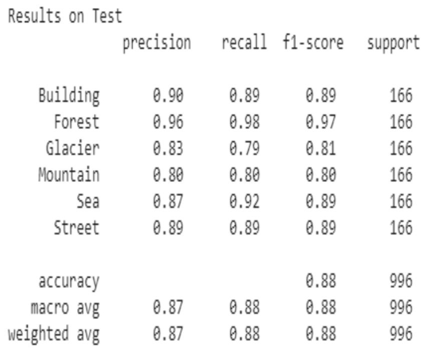


Figure 3: Experiments

In the following subsections, the results obtained will be presented.

3.1 First experiment

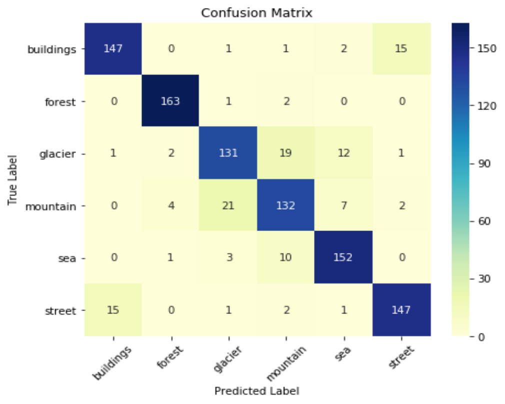
In this case, the VGG16 was cut on the fourth block, after the MaxPooling. The features extracted from the network are feature volumes that are collapsed in one dimension by a flatten function: in this case the number of features was 14x14x512=100,352. After that, it was used the Logistic Regression model previously shown. The result of this experiment are shown in the following figures.



```
Results on Test
              precision    recall  f1-score   support

    Building       0.90      0.89      0.89       166
      Forest       0.96      0.98      0.97       166
     Glacier       0.83      0.79      0.81       166
    Mountain       0.80      0.80      0.80       166
         Sea       0.87      0.92      0.89       166
      Street       0.89      0.89      0.89       166

    accuracy                           0.88       996
   macro avg       0.87      0.88      0.88       996
weighted avg       0.87      0.88      0.88       996
```



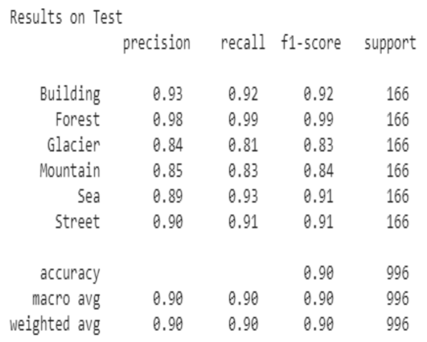
	buildings	forest	glacier	mountain	sea	street
buildings	147	0	1	1	2	15
forest	0	163	1	2	0	0
glacier	1	2	131	19	12	1
mountain	0	4	21	132	7	2
sea	0	1	3	10	152	0
street	15	0	1	2	1	147

Figure 4: Performance on the first experiment

From the results, it is possible to assert that cutting on the fourth block and using the Logistic Regression model the accuracy score on the Test images is equal to 0.88. Instead, from the confusion matrix it is possible to notice that the model makes some mistakes during the classification of mountains and glaciers.
Regarding the performance on the Train, the model correctly identifies all the images. So, the accuracy is equal to 100%. Therefore, there is an overfitting problem.

3.2 Second experiment

In this case, the VGG16 was cut on the fifth block, after the MaxPooling. The features extracted from the network are feature volumes that are collapsed in one dimension by a flatten function: : in this case the number of features was 7x7x512=25,088. After that, it was used the Logistic Regression model previously shown. The result of this experiment are shown in the following figures.



```
Results on Test
              precision    recall  f1-score   support

    Building       0.93      0.92      0.92       166
      Forest       0.98      0.99      0.99       166
     Glacier       0.84      0.81      0.83       166
    Mountain       0.85      0.83      0.84       166
         Sea       0.89      0.93      0.91       166
      Street       0.90      0.91      0.91       166

    accuracy                           0.90       996
   macro avg       0.90      0.90      0.90       996
weighted avg       0.90      0.90      0.90       996
```

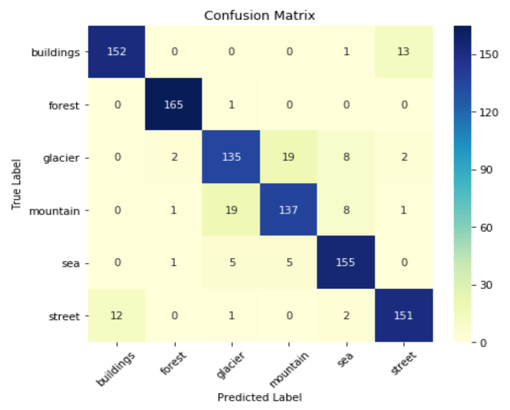
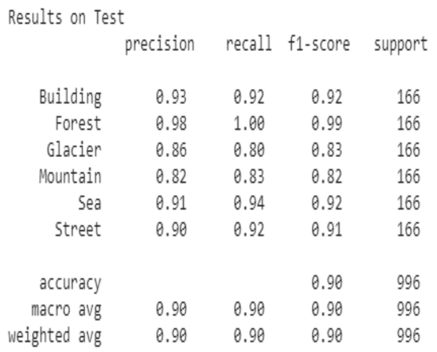


Figure 5: Performance on the second experiment

From the results, it is possible to assert that cutting on the fifth block and using the Logistic Regression model the accuracy score on the Test images is equal to 0.90. Instead, from the confusion matrix it is possible to notice that the model makes some mistakes during the classification of mountains and glaciers.
Regarding the performance on the Train, the model correctly identifies all the images. So, the accuracy is equal to 100%. Therefore, there is an overfitting problem.

3.3 Third experiment

In this last case, the VGG16 was cut on the first fully connected layer, after the flatten Layer. The features extracted from the network was already flatted, where the dimension is equal to 4,096. After that, it was used the Logistic Regression model previously shown. The result of this experiment are shown in the following figures.



```
Results on Test
              precision    recall  f1-score   support

    Building       0.93      0.92      0.92       166
      Forest       0.98      1.00      0.99       166
     Glacier       0.86      0.80      0.83       166
    Mountain       0.82      0.83      0.82       166
         Sea       0.91      0.94      0.92       166
      Street       0.90      0.92      0.91       166

    accuracy                           0.90       996
   macro avg       0.90      0.90      0.90       996
weighted avg       0.90      0.90      0.90       996
```

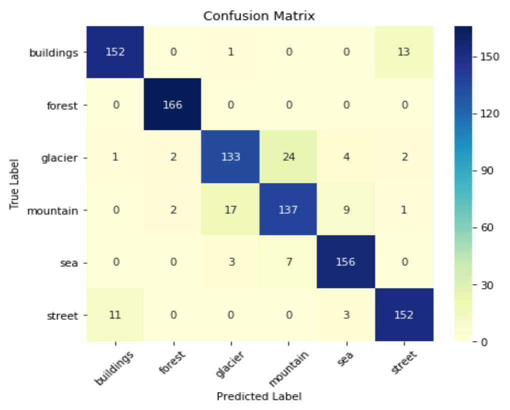


Figure 6: Performance on the first experiment

From the results, it is possible to assert that cutting on the first fully connected layer and using

the Logistic Regression model the accuracy score on the Test images is equal to 0.90. Instead, from the confusion matrix it is possible to notice that the model makes some mistakes during the classification of mountains and glaciers.
Regarding the performance on the Train, the model correctly identifies all the images. So, the accuracy is equal to 100%. Therefore, there is an overfitting problem.

4 Conclusions

In conclusion, it is possible to assert that the choice of where to cut the network conditions the performance of the model, but the most important aspect is the computational time used to extract the feature from the VGG16: in fact, if the network is cut earlier, the computational cost increases. Viceversa, if the network is cut later (fully connected level) the computational cost decreases. In this experiment, the best choice is to cut the network later. The reason is that probably the images used to train the VGG16 were very similar to the images used in this experiment.