Deep Learning Project – Project 2

Course Deep Learning

Professor:

Andrea Asperti, Tutor …..

Students:

Lorenzo Visentini, Master Degree in Management Engineering, ID: 0000….

Gianmarco Bolcato, Master Degree in Management Engineering, ID: 0000…

**Hardware**

The hardware we used for this project was:

*Lorenzo base hardware*

Laptop Acer Aspire …

CPU Intel I7-7700U

GPU Intel

RAM 8 GB

SSD Samsung Evo 1 TB

*Gianmarco base hardware*

Laptop …..

CPU ……

GPU ……

RAM …..

HDD ……

*Colab hardware*

Google Colab or “the Colaboratory” is a free cloud service hosted by Google to encourage Machine Learning and Artificial Intelligence research.

In particular, due to the low computational power of our base hardwares we decided to use Google Colab Jupyter Notebook, which notebooks allows you to combine executable code in Python and rich text in a single document.

The strength of this platform is to create your code as notebook, with cells that can be run once a time. At the same time, Colab offers specific settings including the possibility to run the code on a gpu hosted runtime

Here you can see the configuration of the CPU provided by Google colab during our project:

Architecture: x86\_64

CPU op-mode(s): 32-bit, 64-bit

Byte Order: Little Endian

CPU(s): 2

On-line CPU(s) list: 0,1

Thread(s) per core: 2

Core(s) per socket: 1

Socket(s): 1

NUMA node(s): 1

Vendor ID: GenuineIntel

CPU family: 6

Model: 63

Model name: Intel(R) Xeon(R) CPU @ 2.30GHz

Stepping: 0

CPU MHz: 2300.000

BogoMIPS: 4600.00

Hypervisor vendor: KVM

Virtualization type: full

L1d cache: 32K

L1i cache: 32K

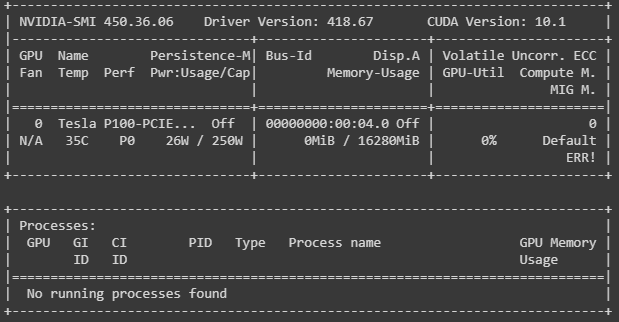
L2 cache: 256K

L3 cache: 46080K

NUMA node0 CPU(s): 0,1

Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ss ht syscall nx pdpe1gb rdtscp lm constant\_tsc rep\_good nopl xtopology nonstop\_tsc cpuid tsc\_known\_freq pni pclmulqdq ssse3 fma cx16 pcid sse4\_1 sse4\_2 x2apic movbe popcnt aes xsave avx f16c rdrand hypervisor lahf\_lm abm invpcid\_single ssbd ibrs ibpb stibp fsgsbase tsc\_adjust bmi1 avx2 smep bmi2 erms invpcid xsaveopt arat md\_clear arch\_capabilities

And here the GPU specs:



**Software**

*Python*

Programming language at high level, very used in the Artificial Intelligence (Machine Learning and Deep Learning) field, thanks to the import of specific libraries.

In our project, we used the *Keras* library API for everything related to models and KPI.

*Google Colab*

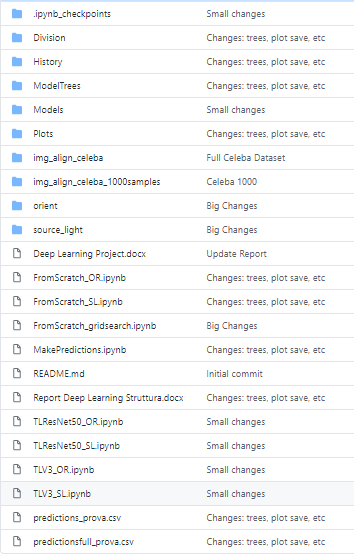
Describe early in this paper. Employed, as said before, in order to overcome all the hardware difficulties.

*GitHub and GitHub Desktop*

GitHub is a hosting service for software projects and a version control using Git. It offers the possibility to work on repository that can be modified by all the members of the group with authorization.

We used GitHub Desktop as the desktop application of GitHub. The advantage is an easy and complete interface, provides [access control](https://en.wikipedia.org/wiki/Access_control) and several collaboration features such as [bug tracking](https://en.wikipedia.org/wiki/Bug_tracking_system), [feature](https://en.wikipedia.org/wiki/Software_feature) requests, [task management](https://en.wikipedia.org/wiki/Task_management), and [wikis](https://en.wikipedia.org/wiki/Wiki) for every project.

Our GitHub project’s repository:



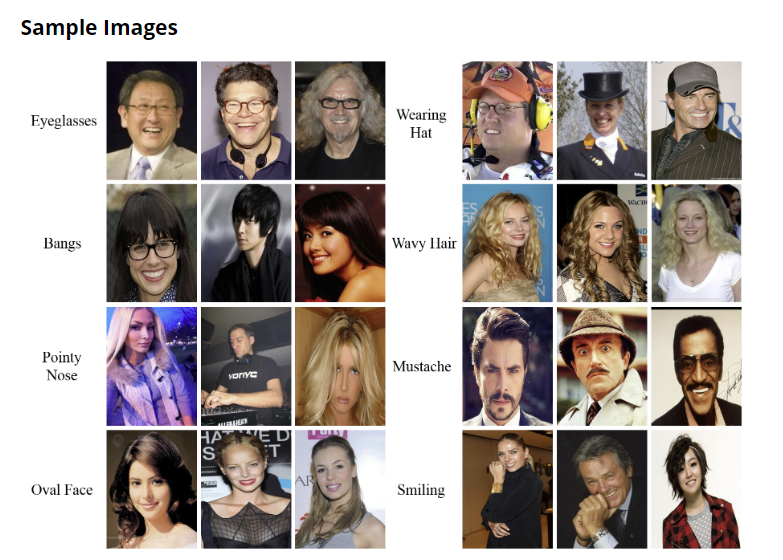
**Problem**

During the last few years Facial Attribute Classification (FAC) attracts a lot of attention for uses like face recognition, recommendation systems, etc. In other words, given a facial image, the task of FAC is to predict facial attributes.

Our project goal is to create a classifier using Neural Networks to predict the face orientation and the face light source origin of images from a very know in the Deep Learning field, called CelebA Dataset.

The **CelebFaces Attributes Dataset (CelebA)** is a large-scale face attributes dataset with more than 200’000 celebrity images, each with 40 attribute annotations. This dataset is great for training and testing models for face detection, particularly for recognizing facial attributes such as finding people with brown hair, are smiling, or wearing glasses. Images cover large pose variations, background clutter, diverse people, supported by a large quantity of images and rich annotations. This data was originally collected by researchers at MMLAB, The Chinese University of Hong Kong.

*Sample images*



*Face Orientation*

The first request is to code a classifier which aims to classify images taken as input based on the orientation of the faces (i.e. left, center, right).

*Source of light*

The second request is related to build a classifier which recognize where the source of light is in the photo (i.e. left, center, right).

**Steps adopted**

Firstly, before adopting the strategy “try, observe and refine”, we spent time building up the whole structure.

The approach is the same for both the problems.

Pipeline:

* Manual classification of the Celeba Dataset
* Focus on the transfer learning models and then finetune them
* Focus on the ‘From scratch’ model and following finetuning
* Test the best transfer learning model and the from scratch code
* Analyse the results and final comparation

Mettere meglio

The preparation of the datasets took a lot of time, a very onerous task.

Specifically, we built a python code to put the files in distinct folders. Strictly connected to the requests of the project, the folders were 3, all of these will be used for both the problems as we highlighted that the best composition of the starting dataset is divide manually the images in center, left and right.

With the other group, we decided to divide the task and ended to classify pictures based on source light orientation. 10’000 images for each dataset, orient and source light.

The light, especially in images with medium-low resolution, is not easy to detect as the reflection can create issues about where the source is. Adding to this, lights sometimes provoked misunderstanding due to noise of the background and artificial setup made by photographers.

The importance of the input of a convolutional neural network is very high and consequently it’s indispensable to clean data as well as perform very accurate manual classification. And since detecting the source light is not an exclusive data task – light can come from different position and, as said before, reflections can cause some issues – we adopted rules enhancing the robustness of this stage.

Firstly, the classes decided are only three, avoiding mixed classes such as center-left, center-right. The division is led by the portion of the image – or portion of the face - that has mostly of the light.

Secondly, some transformation to Celeba images has been performed: HSV (Hue, Saturation, Value) color model which describes colors in terms of their shade, pixel-value based threshold pointing out only pixels’ values that are high than 170 – founded by trying – and traditional image.

Example:

Immagine che contiene fotografia, persona, posando, guardando

Descrizione generata automaticamente

Hence, we applied some operations similar to a pre-processing stage.

During the manual classification, many pictures have been discarded since the categorization was nearly impossible, despite the efforts.

**Transfer Learning**

Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. It consists of taking features learnt on one, first, problem and apply them on a new second, but related, problem.

In Convolutional Neural Networks field, an existent neural network, pre-trained on a similar problem, is transposed to the main problem of interest.

The benefits of the transfer learning approach are many and often related to the request of the project. Usually, the starting datasets are not large, not big enough to achieve important results in terms of accuracy and loss. The use of data augmentation can help but at the same time can also introduce bias that get the things worse – in this paper we will talk about data augmentation. In addiction to this, the risk of overfitting is very high due to the implementation of a complex neural network, with millions of parameters.

Hence, the Transfer Learning comes in handy allowing the use of pre-trained neural networks. This approach overcome, also, the issue of the expensiveness of the training since it takes a lot of time and requires dedicated hardware.

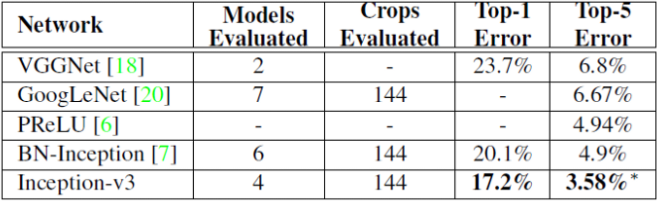
On the other hand, transfer learning is not always the best solution. It necessary to study in depth the project and decide if this is similar to the problem solved by state-of-art CNNs.

For the project we opted for two very known CNNs, GoogLeNet InceptionV3 and ResNet50.

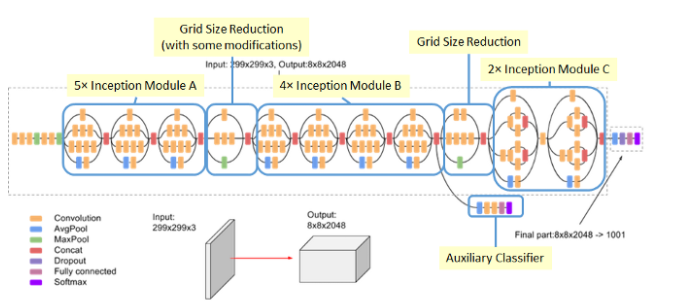
The reason why the choice of these two state-of-art CNNs relies on that both have introduced particular modules or news.

**InceptionV3**

With 42 layers, lower error rate is obtained and make it become the 1st Runner Up for image classification in ILSVRC (ImageNet Large Scale Visual Recognition Competition) 2015.



Architecture



InceptionV3 (why, tree, layers added, plots, tables with accuracy)

**ResNet50**

ResNet50 (why, tree, layers added, plots, tables with accuracy)

Best TL\_CNN accuracy on whole dataset and csv predictions

FromScratch

Advantages and disadvantages

Tree

Why used some types of layers

Results on train and test with plots and accuracy

Results on whole dataset

Final considerations

Bibliography and sitography