**Labeling and Classification on Celeba**

Marco Acerbis and Nicolaas Ruberg

Deep Learning Course (A. A.19/20)

Prof. Andrea Asperti

Exam Session 2020

Table of Contents

[Abstract 4](#_Toc45806941)

[Labeling and Classification on Celeba 5](#_Toc45806942)

[Introduction 5](#_Toc45806943)

[Experiment setup 5](#_Toc45806944)

[Face Classifiers 6](#_Toc45806945)

[Face Orientation 7](#_Toc45806946)

[Implementation: VGG19 7](#_Toc45806947)

[1st experiment with VGG-19 configuration 8](#_Toc45806948)

[Training 10](#_Toc45806949)

[Implementation: Mobile Net + Clustering 11](#_Toc45806950)

[Implementation: Convolutional Neural Network 11](#_Toc45806951)

[CNN configuration 12](#_Toc45806952)

[Training 14](#_Toc45806953)

[Light Orientation 15](#_Toc45806954)

[Implementation: VGG19 16](#_Toc45806955)

[VGG-19 configuration 16](#_Toc45806956)

[Training 16](#_Toc45806957)

[CNN configuration 17](#_Toc45806958)

[Training 17](#_Toc45806959)

[[Heading 1] 19](#_Toc45806960)

[[Heading 2] 19](#_Toc45806961)

[[Heading 3] 19](#_Toc45806962)

[References 20](#_Toc45806963)

[Footnotes 21](#_Toc45806964)

[Tables 22](#_Toc45806965)

[Figures 23](#_Toc45806966)

Abstract

[The abstract should be one paragraph of between 150 and 250 words. It is not indented. Section titles, such as the word Abstract above, are not considered headings so they don’t use bold heading format. Instead, use the Section Title style. This style automatically starts your section on a new page, so you don’t have to add page breaks. To apply any text style in this document with just a tap, on the Home tab of the ribbon, check out Styles.]

Keywords: [Tap here to add keywords.]

Labeling and Classification on Celeba

[The body of your paper uses a half-inch first line indent and is double-spaced. APA style provides for up to five heading levels, shown in the paragraphs that follow. Note that the word Introduction should not be used as an initial heading, as it’s assumed that your paper begins with an introduction.]

# Introduction

[The first two heading levels get their own paragraph, as shown here. Headings 3, 4, and 5 are run-in headings used at the beginning of the paragraph.]

# Experiment setup

For our experiment we used Google™ Colaboratory or Google™ Colab [1]. Colab is a free platform that executes python code through the browser and is especially design for machine learning, data analysis and education. Using the principle of Jupyter Notebooks, Colab relies on Notebooks for storing and sharing python code.

Next, we present the configuration of the Virtual provided from Google™ Colab during our experiments:

1. !lscpu
2. Architecture:        x86\_64
3. CPU op-mode(s):      32-bit, 64-bit
4. Byte Order:          Little Endian
5. CPU(s):              4
6. On-line CPU(s) list: 0-3
7. Thread(s) per core:  2
8. Core(s) per socket:  2
9. Socket(s):           1
10. NUMA node(s):        1
11. Vendor ID:           GenuineIntel
12. CPU family:          6
13. Model:               63
14. Model name:          Intel(R) Xeon(R) CPU @ 2.30GHz
15. Stepping:            0
16. CPU MHz:             2300.000
17. BogoMIPS:            4600.00
18. Hypervisor vendor:   KVM
19. Virtualization type: full
20. L1d cache:           32K
21. L1i cache:           32K
22. L2 cache:            256K
23. L3 cache:            46080K
24. NUMA node0 CPU(s):   0-3
25. 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

The Graphic adapter configuration:

1. !nvidia-smi
2. Fri Jul 10 12:30:47 2020
3. +-----------------------------------------------------------------------------+
4. | NVIDIA-SMI 450.36.06    Driver Version: 418.67       CUDA Version: 10.1     |
5. |-------------------------------+----------------------+----------------------+
6. | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
7. | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
8. |                               |                      |               MIG M. |
9. |===============================+======================+======================|
10. |   0  Tesla P100-PCIE...  Off  | 00000000:00:04.0 Off |                    0 |
11. | N/A   44C    P0    29W / 250W |      0MiB / 16280MiB |      0%      Default |
12. |                               |                      |                 ERR! |
13. +-------------------------------+----------------------+----------------------+
15. +-----------------------------------------------------------------------------+
16. | Processes:                                                                  |
17. |  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
18. |        ID   ID                                                   Usage      |
19. |=============================================================================|
20. |  No running processes found                                                 |
21. +-----------------------------------------------------------------------------+

# Face Classifiers

Facial Attribute Classification (FAC) attracts a lot of attention since it can be used in several applications like face recognition, micro-expression recognition, recommendation systems etc. Given a facial image the task of FAC is to predict facial attributes. Our goal is to predict the face orientation and the face light source.

## Face Orientation

To generate our classifier for the Face Orientation we conducted our experiment with three different implementations: VGG19, MobileNet+ Clustering and CNN with 4 convolution layers.

We ran our experiments with a training set of 8.383 images with a validation set of 1.181 images, generating three classifications: pose left, pose front and pose right. We also verify our model with a test set of 1.005 images.

Here are some examples of the pose images:

|  |  |  |
| --- | --- | --- |
| A close up of a person  Description automatically generated | A person wearing glasses and smiling at the camera  Description automatically generated | A person smiling for the camera  Description automatically generated |
| **Left Pose** | **Center Pose** | **Right Pose** |

## Implementation: VGG19

Our first implementation of the Facial Classifier uses Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG-19) on the Celeba Images. VGG19 is a model that uses transfer learning. We developed three experiments to better tune the hyperparameters of the VGG19 approach.

For our experiment we use the whole image resolution (218x178) as the input of the Network.

VGG-19 configuration – Next we present the configuration of out Neural Network.

In our experiment, in order to avoid overfitting, we had to vary the Dropout on the inner layers. We reached good results with a Drop out of 0.7.

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Training – We trained our model for 20 epochs with batch size of 32 images, compiled with *categorical\_crossentropy* loss function with *Adam* optimizer. We executed this experiment with three different VGG19 setups.

* First experiment is using the VGG19 configuration presented in the previous section;
* Second experiment is using the same configuration but with data augmentation. This is the setup for the image data generator (Rotation=0, Width shift=0.08, Shear=0.3, Height shift=0.08 and Zoom=0.08); and
* The third experiment we change the VGG19 structure, with a different transfer learning technique, we added two dense network layers to the VGG19 network, setting these final layers to be trainable.

Figure 1 presents the measured accuracy for our training and validation, for the three scenarios:

|  |  |  |
| --- | --- | --- |
|  |  | Tela de computador com texto preto sobre fundo branco  Descrição gerada automaticamente |
| **VGG19** | **VGG19 with data augmentation** | **VGG19 – 2 dense layers added** |

Figure - VGG19 accuracy - 20 epochs

Figure 2 presents the loss for our training and validation set:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **VGG19** | **VGG19 with data augmentation** | **VGG19 – two dense layers added** |

Figure - VGG19 loss - 20 epochs

Results: Finally, we run our experiments with our test set, we achieved the following results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| VGG19 | | VGG19 with data augmentation | | VGG19 – two dense layers added | |
| Accuracy | Loss | Accuracy | Loss | Accuracy | Loss |
| 82,96% | **0.555** | 71% | 2.22 | 76% | 0.7 |

## Implementation: Mobile Net + Clustering

## Implementation: Convolutional Neural Network

﻿Our third implementation of the Facial Classifier uses Convolution Neural Network on the Celeba Images. In the work of James Le[2]**,** experiments were conducted to generate a classifier for Fashion MNIST dataset, with images of 28x28 pixels. In his work he assed four solutions: three with CNNs varying the number of Convolutional Layers (one, three and four), and a VGG-19 pre-trained model.

Inspired in the James Le experiment[2]we devised our CNN with four Convolutional layers For our experiment we uses the whole image resolution (218x178) as the input of the Network.

In our experiment, in order to avoid overfitting, we had to vary the Dropout on the inner layers. We reached good results with a Drop out of 0.7.

CNN configuration – Next we present the configuration of out CNN

![A screenshot of a cell phone

Description automatically generated]()

Figure - CNN4 configuration

Training – As in the experiment with VGG19, we trained our model for 20 epochs with batch size of 32 images, compiled with *categorical\_crossentropy* loss function with *Adam* optimizer.

We executed this experiment with two different CNN setups.

* First experiment is using the CNN configuration presented in the previous section; and
* Second experiment is using the same configuration but with data augmentation. This is the setup for the image data generator (Rotation=8, Width shift=0.08, Shear=0.3, Height shift=0.08 and Zoom=0.08).

Figure 2 presents the measured accuracy for our training and validation.

|  |  |
| --- | --- |
| A close up of text on a black background  Description automatically generated |  |
| **CNN** | **CNN with data augmentation** |

Figure - CNN and CNN with data augmentation accuracy

Figure 6 - presents the loss for our training and validation set.

|  |  |
| --- | --- |
|  |  |
| **CNN** | **CNN with data augmentation** |

Figure - presents CNN and CNN with data augmentation loss

Results: Finally, we run our experiments with our test set, we achieved the following results:

|  |  |  |  |
| --- | --- | --- | --- |
| CNN | | CNN with data augmentation | |
| Accuracy | Loss | Accuracy | Loss |
| 89.31% | **0.22** | 89.61% | 0.233 |

## Light Orientation

The computation of light source directions from images is an ill-posed problem, since many possible solutions can be observed in an image [3]. Moreover, generate a classifier with few categories for a 360° possible light source generates dubious classifications. Also, several light sources increase the difficulty on the classification task since some clues to the light source origin are tempered (some clues includes, shadows, image intensity, etc.)

To generate our classifier for the Light Orientation we conducted the same experiments that we did for the Face orientation feature (with three different implementations: VGG19, MobileNet+ Clustering and CNN with 4 convolution layers).

We ran our experiments with a training set of 1.1000 images with a validation set of 1.000 images, generating three classifications: left light source, front/center light source and right light source. We also verify our model with a test set of 1.000 images.

Here are some examples of the light orientation images:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Left Light Source** | **Center Light Source** | **Right Light Source** |

## Implementation: VGG19

In our experiment VGG19 for light source orientation, we have taken our best performance VGG19 implementation on the face pose classification and used it to generate the classifier for our light source orientation.

As in the face orientation experiment, we use the whole image resolution (218x178) as the input of the Network.

VGG-19 configuration – Since the classification over the light source is more sensible, we used a drop out of 0.5.

Training – We trained our model for 20 epochs with batch size of 32, compiled with *categorical\_crossentropy* loss function with *Adam* optimizer.

Figure 8 presents the measured accuracy for our training and validation.

|  |  |
| --- | --- |
|  |  |
| **Accuracy** | **Loss** |

Figure - VGG19 accuracy and loss for Light Source orientation

Results: We can observe that the curves for accuracy and loss do not indicate a good model for our classifier. Moreover, with our test set, we get very poor results:

|  |  |
| --- | --- |
| Accuracy | Loss |
| 53.7% | 1.09 |

CNN configuration – As in the previous experiment we have a drop out of 0.7.

Training – We trained our model for 20 epochs with batch size of 32, compiled with *categorical\_crossentropy* loss function with *Adam* optimizer.

Figure 10 presents the measured accuracy for our training and validation.

|  |  |
| --- | --- |
|  |  |
| **Accuracy** | **Loss** |

Figure - CNN4 accuracy and loss for Light Source orientation

Results: We have some improvement with the CNN solution over the VGG19 one, but we still observe that the curves for accuracy and loss do not indicate a perfect model for our classifier. That being sad, with our test set, we get reasonable results:

|  |  |
| --- | --- |
| Accuracy | Loss |
| 80.6% | 0.6 |

# [Heading 1]

[The first two heading levels get their own paragraph, as shown here. Headings 3, 4, and 5 are run-in headings used at the beginning of the paragraph.]

## [Heading 2]1

[To update the table of contents (TOC), apply the appropriate heading style to just the heading text at the start of a paragraph and it will show up in your TOC. To do this, select the text for your heading. Then, apply the style you need.]

[Heading 3]. [Include a period at the end of a run-in heading. Note that you can include consecutive paragraphs with their own headings, where appropriate.]

[Heading 4]. [When using headings, don’t skip levels. If you need a heading 3, 4, or 5 with no text following it before the next heading, just add a period at the end of the heading and then start a new paragraph for the subheading and its text.] (Last Name, Year)

[Heading 5]. [Like all sections of your paper, references start on their own page, as shown on the page that follows. The body of the References section uses the Bibliography style. For more detailed information on formatting references, see the APA Style Manual, 6th Edition. (Last Name, Year)

# Bibliography

|  |  |
| --- | --- |
| [1] | G. Colab, "https://research.google.com/colaboratory/faq.html," [Online]. [Accessed 10 07 2020]. |
| [2] | J. Le, "Towards Data Science," 7 10 2018. [Online]. Available: https://towardsdatascience.com/the-4-convolutional-neural-network-models-that-can-classify-your-fashion-images-9fe7f3e5399d. [Accessed 10 07 2020]. |
| [3] | F. M. Last Name, "Article Title," *Journal Title,* pp. Pages From - To, Year. |
| [4] | F. M. Last Name, Book Title, City Name: Publisher Name, Year. |

Footnotes

1[Add footnotes, if any, on their own page following references. For APA formatting requirements, it’s easy to just type your own footnote references and notes. To format a footnote reference, select the number and then apply the Footnote Reference. The body of a footnote, such as this example, uses the Normal text style. (Note: If you delete this sample footnote, don’t forget to delete its in-text reference as well.)]

Tables

Table 1

[Table Title]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column Head | Column Head | Column Head | Column Head | Column Head |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |

Note: [Place all tables for your paper in a tables section, following references (and, if applicable, footnotes). Start a new page for each table, include a table number and table title for each, as shown on this page. All explanatory text appears in a table note that follows the table, such as this one. Use the Table/Figure style to get the spacing between table and note. Tables in APA format can use single or 1.5 line spacing. Include a heading for every row and column, even if the content seems obvious. To insert a table, on the Insert tab, tap Table. New tables that you create in this document use APA format by default.]

Figures



Figure 1. [Include all figures in their own section, following references (and footnotes and tables, if applicable). Include a numbered caption for each figure. Use the Table/Figure style for easy spacing between figure and caption.]

For more information about all elements of APA formatting, please consult the APA Style Manual, 6th Edition.