Gender and Age classification using Convolutional Neural Networks

NEURAL NETWORKS & DEEP LEARNING
ASSIGNMENT 2

Salafranca Pardo Jaime N2303635K
Ponce Fernandez Alejandro N2303631L

Index

1. Introduction

In this assignment we will use Convolutional Neural Networks (CNN) to predict a person's gender and age from a picture. We will start by understanding the data set and preparing instances to be read by a CNN.

The whole process of convolutions for images will be discussed as well as the building of different models. Different hyperparameters will be tested and their results compared and evaluated to conclude.

2. Data preprocessing

2.1 Data set and attributes

In this assignment we will use the Adience data set [1]. It consists of a big table with data and a folder with all the images called ‘faces’. The table contains initially 19730 instances where the following attributes are given:

- user_id: This attribute is of type object. It contains the ID associated with the person of the image, which is the name of a folder inside the ‘faces’ folder where the pictures of this user are stored.
- original_image: This attribute is of type object. It contains the reference to the exact picture described in that row.
- face_id: This attribute is of type integer. It refers to the number of pictures of the same user.
- gender: This attribute is of type object. It indicates the gender of the person in the picture associated. It can have four different values: ‘f’, ‘m’, ‘u’, ‘nan’.
- age: This attribute is of type object. It indicates the age of the person in the picture associated. Its shape is both intervals and singular data, e.g. ‘(0-2)’ or ‘55’
- Size and angles values: This is not just one attribute but a group of non-relevant ones. Some information about the picture size and tilt of the faces is given which is non-relevant when predicting either gender or age.

2.2 Modifications and feature engineering

The first modification we introduced was joining face_id, user_id, and original_image under an attribute named image_path. We concatenated the attributes as

./Adience/faces/**user_id**/coarse_tilt_aligned_face.**face_id**.**original_image**

This will be the path in our working directory to find the exact picture referred to in a row. Then, we dropped the rows containing ‘nan’ values as they were not useful for either training or testing. Inside the gender attribute, we had some instances classified as ‘u’. We assume that this referred to unclassified data and as these were not useful we removed their associated rows. After dropping these values the total number of rows was: 18851 instances.

Regarding the age variable, as it was mentioned before, we were given both intervals and precise ages for the different instances. We decided to create new intervals as the intervals given were not precise and some of them overlapped others. We did as follow for **dataset 1**:

- "0-2" containing: ['(0, 2)', '2'] → Count: 2454
- "3-6" containing: ['(4, 6)', '3'] → Count: 2137

- "8-23" containing: ['(15, 20)', '(8, 12)', '22', '13', '23', '(8, 23)'] → Count: 4016
- "25-32" containing: ['(25, 32)', '(27, 32)', '29', '32'] → Count: 4728
- "33-37" containing: ['35', '36', '34'] → Count: 385
- "38-48" containing: ['(38, 43)', '(38, 48)', '45', '(38, 42)', '42', '46'] → Count: 2353
- "48-53" containing: ['(48, 53)'] → Count: 808
- "54-59" containing: ['55', '58', '57', '56'] → Count: 103
- "60-100" containing: ['(60, 100)'] → Count: 832

As it can be observed, the counts vary widely among the intervals. Although these intervals are precise they might be difficult to predict for the CNN as some will not have many training instances. Therefore, we will test both with the previous intervals and these new ones (stored in a different data set) that aim to have more balanced (in number of instances) classes for **dataset 2**:

- "0-6" containing: ['(0, 2)', '2', '(4, 6)', '3'] → Count: 4691
- "8-23" containing ['(15, 20)', '(8, 12)', '22', '13', '23', '(8, 23)']: → Count: 4016
- "25-35" containing: ['(25, 32)', '(27, 32)', '29', '32', '35'] → Count: 4995
- "36-54" containing: ['36', '34', '(38, 43)', '(38, 48)', '45', '(38, 42)', '42', '46', '(48, 53)'] → Count: 3279
- "54+" containing: ['55', '58', '57', '56', '(60, 100)'] → Count: 935

Lastly, we will create a mapping where we will map both of our target variables (age and gender) to numbers. In the case of gender, females will be 0 and males 1. For age, we will assign integers starting from 0 to each interval for the two different data sets.

2.3 Graphs

After the previous modifications there are graphs we can obtain with some relevant information. Firstly, we can check that the distribution of gender is almost equal.

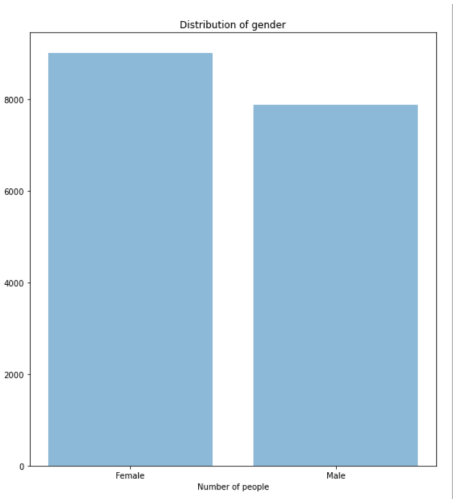
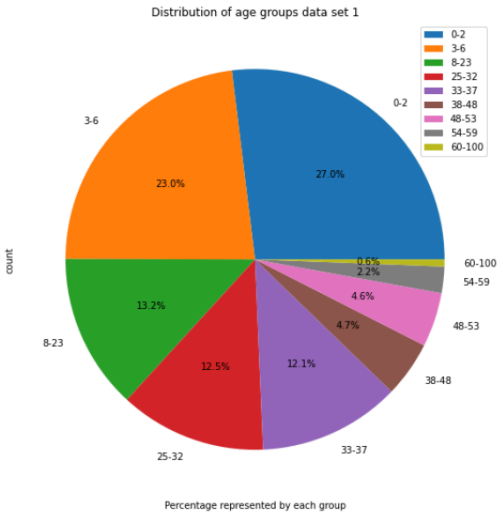
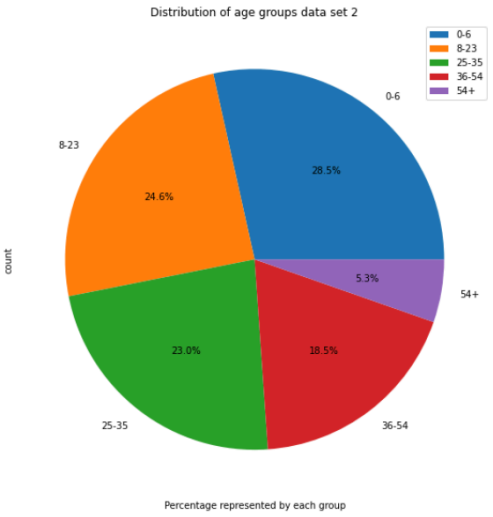


Figure 1. Distribution of data by gender.

We can also observe how the distribution of the age intervals created for both datasets is different. On the left graph below we can observe that the distribution is very uneven as it was mentioned before. This might lead to worse classification of the age. On the graph below on the right we can see 4 classes with a very even distribution and a 5th one with less instances. However, we believe that these intervals are optimal as the 54+ should be easier to classify as facial features are more distinctive.





-

Figure 2. Distribution of age for dataset 1 (on the left) and dataset 2 (on the right).

Lastly, we can check how the distribution of males and females is inside each age group. We can observe that inside each age group, we have very similar distributions of males and females, which is a positive thing as we will have training examples for everything:

Figure 3. Distribution of age for dataset 1 (on the left) and dataset 2 (on the right).

2.4 Preparation of auxiliary datasets

We will divide our dataset into 75% training data and 25% testing data. *X* will be assigned as just the variable 'image_path' and *y* as the 'gender' or 'age' (depending on what the model is for) as they are the variables we want to predict. We decided not to do k-fold cross-validation as we found the training for the CNN to be exhaustive enough with just normal validation.

2.5 Resizing of images

We transformed the *X* variable from being just a string containing the image path to the content a matrix for each row of size (227, 227, 3). This means that the image associated with each row was opened, resized to 227x227 and the three color values for each pixel were stored.

The 227x227 crop size is not arbitrary. After doing some research, we found a study [2] that compared the performance of 2D Convolutional Neural Networks with different image cropping sizes. They found experimentally that the ideal cropping size for face recognition is 227x227 [2, p.18].

3. Models

We needed to design a network to predict gender and/or age for this work. Deciding the structure and the number of convolutions and layers for our network was not easy, as we had to take into account both accuracy and training time (as we were not able to train super complex networks).

3.1 CNN Models for gender recognition

3.1.1 Traditional Models Performance (EfficientNetB5)

To determine how accurate our model should be we decided to study the performance of a classical CNN as a benchmark for the rest of the project. There is a wide variety of models to choose from such as VGG or Alex Net. For this part of the project, we decided to use a model that combines two regularization methods: Lasso and Ridge. This method (EfficientNet) combines the two penalty functions inside the same model [3].

We implemented an Efficient Net to predict the gender, specifically EfficientNetB5 from tensor flow and added some extra layers to adjust it well to our case. By adding a GlobalPooling followed by two dense layers separated by a dropout we were able to reduce the dimensionality. After training, 88.65% of accuracy was obtained for the validation data. We will use this result as a benchmark.

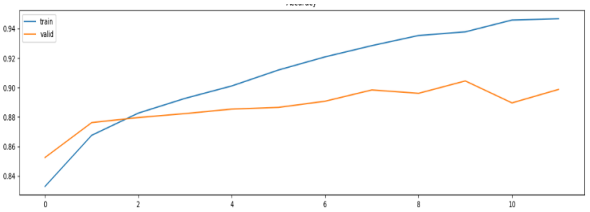


Figure 4. Accuracy of EfficientNetB5 model in gender prediction.

Along this project, we will try to achieve similar performance by using simple CNN networks to predict the gender of a person from a face picture.

3.1.2 Simplified model

The previous models had more than 200 layers so the aim is to find a simpler model with similar accuracy. From [4] we tried to apply a network that had 5 2D convolutions and 3 fully connected layers. However, the sizes proposed for the convolutions and layers were big which lead to very long and demanding training times that we could not deal with. In [5] they discussed 4 2D convolutions and 3 fully connected layers. This structure was less demanding but still the training time was too long. Finally, we found a structure [6] that only used 3 2D convolutions and 3 fully connected layers. All the structures discussed seem to agree in a number of convolutions followed by 3 fully connected layers, so we will test the structure that proposed 3 2D convolution layers as is the less demanding one.

The structure for this model is the following and was described and tested in [6, p. 37]:

- First 2D convolutional layer: consists of 96 layers of 7x7 kernel with a ReLu activation function followed by max pooling operation of 3x3 kernel.
- Second 2D convolutional layer: consists of 256 layers of 5x5 kernel and ReLu activation function followed by max pooling operation of 3x3 kernel.
- Third 3D convolutional layer: consists of 384 filters of 3x3 kernel and ReLu activation function followed by max pooling operation of 3x3 kernel.
- First fully connected layer: consists of 512 neurons with a ReLu activation function receiving the output of the last convolutional layer followers by a dropout layer with probability 0.5 (50% chance of making the neurons output 0).
- Second fully connected layer: consists of 512 neurons with a ReLu activation function that receive the output of the 512 neurons of the previous layer as input and is followed by a dropout layer with a probability of 0.5.
- Third fully connected layer: maps the previous layer output to the final classes of gender and is connected to a soft-max layer that assigns the corresponding probabilities.

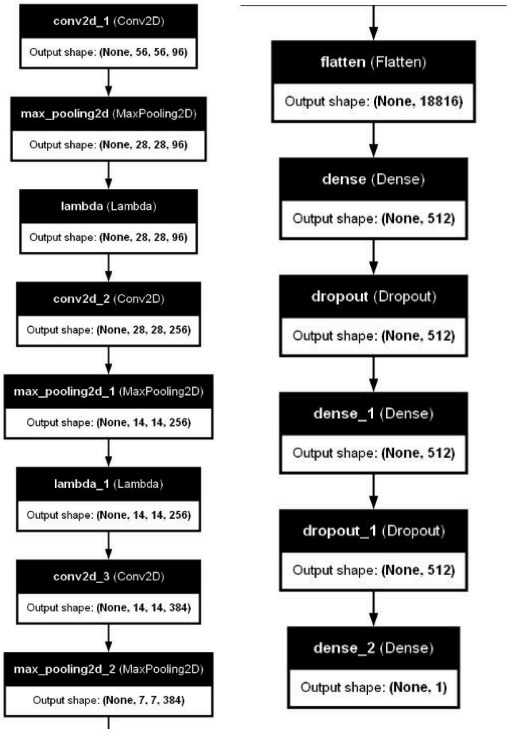


Figure 5. Visual representation of the model.

This model will be referred to as **simplified model**.In [6, p. 38] they discussed that the ideal batch size for the training is 50.Nevertheless, we came to disagree with this after training different models with different batch sizes and comparing their accuracies. We trained models with batch sizes 1, 5, 10 and 50:

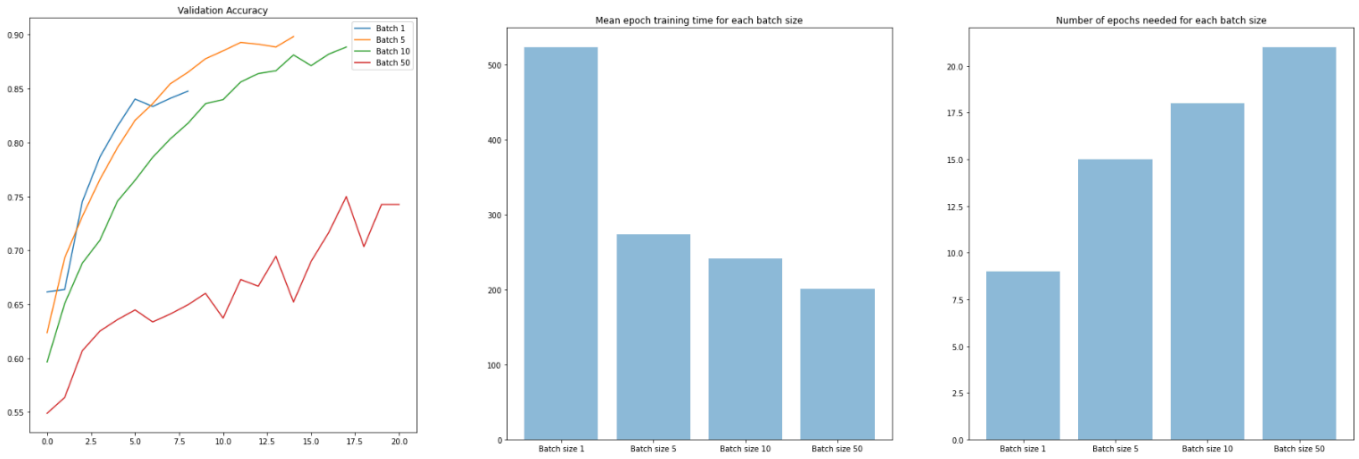


Figure 6. Accuracies, Mean Epoch Training time and number of epochs per batch size.

As we can observe batch size 50 gives very low validation accuracy results, therefore we conclude it is far from being the ideal batch size. Between 1, 5, and 10 we can see how they show very similar accuracy validation results. To choose an optimal batch size, we check training times and a number of epochs. We discard batch size 1 as it takes the longest, and between 5 and 10 we will take 5 as the optimal as gives more accuracy with similar training times. With batch size 5, this model reached a validation accuracy of 89.92%. During the rest of the project we will use **optimal batch size 5.**

3.1.2.1 Image preprocessing using filters

In [7] it was discussed that implementing image filters could lead to improvements when working with images. We decided to compare the accuracies when preprocessing the images with different filters:

- RGB: the standard processing we used in the previous section, where each pixel is given 3 values (red, blue and green).
- GreyScale: an image filter is applied changing it to grayscale where pixels are only given 1 value.
- RGB with Gaussian Blur: the standard RGB image we applied a Gaussian Blur which is used to reduce noise and detail. Here, again, each pixel will be given 3 values.
- GreyScale with Gaussian Blur: we applied a Gaussian Blur to the previous grayscale image. Only one value will be given to each pixel.

Figure 7. Comparison of the preprocessing filters for the images.The grayscale pictures are displayed with the 'viridis' colormap. The picture was randomly selected from the Adience data set.

We trained four different models and in each of them we did the training and testing with a different filter of the mentioned above. This were the results found:

The validations accuracies obtained were 89.92% for RGB, 87.67% for RGB Gaussian Blur, 86.91% for GrayScale and 87.59% for GrayScale Gaussian Blur. RGB was the best performing by a small difference.

When applying the Gaussian Blur filter for RGB and GrayScale pictures gave the same accuracy, and Gaussian Blur did improve the accuracy for GrayScale pictures.

While RGB was still the best accuracy, it is very interesting to know that GrayScale pictures perform well as well. This is because storing GrayScale pictures is way more efficient as they only have one value per pixel compared to three for RGB. Therefore, storage wise, it is three times more efficient.

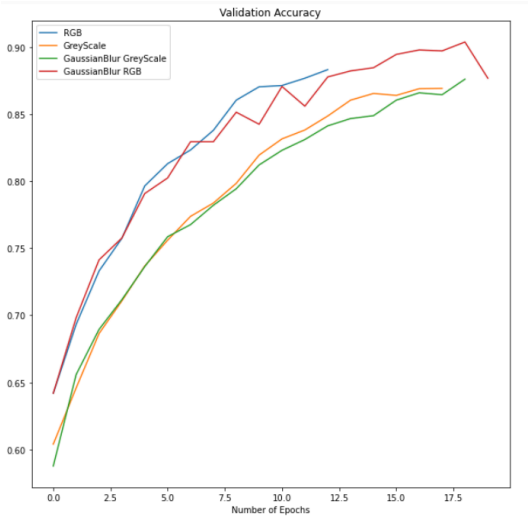
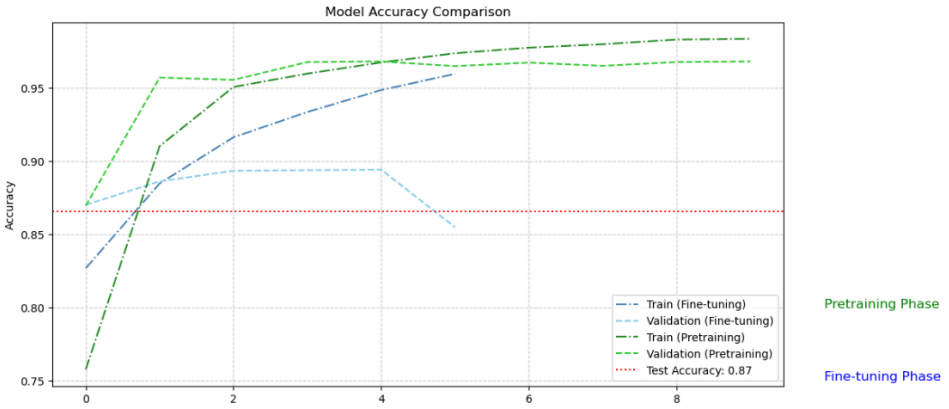


Figure 8. Validation accuracies for models trained with different filtered images.

3.1.2.2 Pretrained model using CelebA

One of the recommendations of the assignment was to pre-train the model with a dataset called CelebA [8]. CelebA contains more than 200,000 images of celebrities. Usually, the pretraining step uses more data than the training step. It is important to remember that the training set has a size of nearly 12,000 pictures and validation is 2,600. Therefore, we will randomly select in total 50,000, between training, testing and validation in order to pretrain the model for gender prediction.

Figure 9. Accuracies before fine-tuning and after.



We used the same model as in the two previous sections for gender prediction and obtained an accuracy of 96.67% for the validation set in CelebA. After, we re-trained the already model with the Adience dataset (fine-tuning). As we can observe, after retraining the model, the validation accuracy went down to below 87%.

Training time was widely reduced for fine-tuning which was an obvious advantage. However, accuracy was not maintained and went down to similar values to what we obtained in previous sections without fine-tuning. When comparing the fine-tuned and regular models, we believe that the first one is less overfitted as it was trained on a wider variety of images. Therefore, if our model was to be used for general gender prediction it would perform better if we used the fine-tuned one despite showing less accuracy.

3.1.2.3 Variations of the model

We decided to explore how different numbers of convolutions affect the accuracy of the model. We run models with 1, 2, 3, 4 and 5 convolutions. For the models with 1 and 2 convolutions, we just removed convolutions from the model described in *"3.1.2 Simplified model"*. For the models with 4 and 5, we added convolutions of 512 layers of 3x3 kernel and ReLu activation function followed by max pooling operation of 3x3 kernel.

As it can be observed the best accuracy was obtained for the model we had already tested with 3 convolutions. Two, three, and four convolutions seem to perform similarly and five convolutions give a good accuracy but take too many epochs to learn.

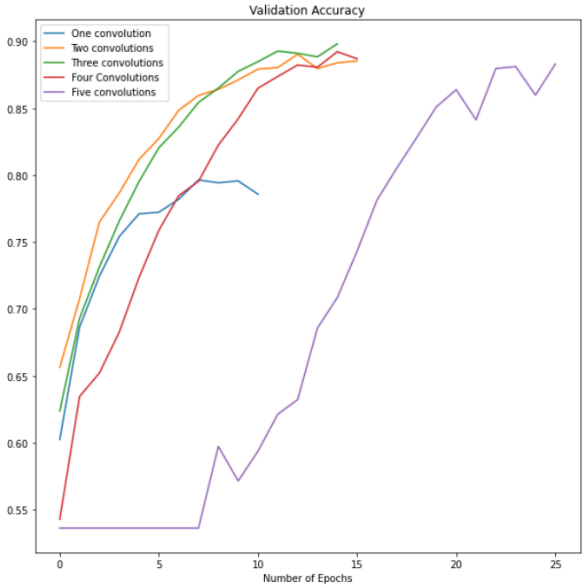


Figure 10. Validation accuracies for the different models.

However, this cannot be used to declare 3 as the optimal number of convolutions as each convolution has parameters that can be modified that will lead to different results. For example, in [4] and [5] they use a different number of filters and kernel space.

In addition, we decided to explore what would happen if instead of regular convolutions we used dilated convolutions. These convolutions are characterized by a different kernel space where there will be space between the kernel elements [8]. We trained a model with the same structure as our simplified model but changed the last convolution to be dilated with dilatation 2. We obtained a validation accuracy of 90.27%, slightly higher than with all regular convolutions. This can be due to the convolutions finding relationships between a broader area and between pixels that were not initially tested together. We wanted to test different combinations of dilate convolutions but the complexity led to extremely long training times.

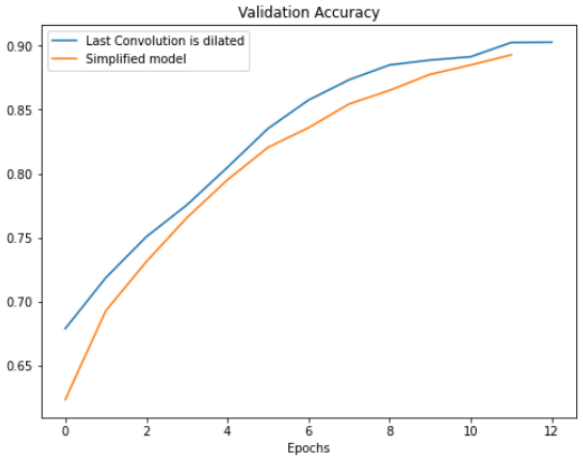


Figure 11. Validation accuracies for the simplified model and dilated convolution variation.

3.1.2.4 Visualization and understanding of the convolutional neural networks

During this project we found very interesting the process of convolution and pooling and how they modify images that are fed to the NN. We decided to visualize this process to understand better how predicting works. We started by visualizing how are filters created in the different layers (convolutions and max poolings):

As we can observe on the first convolution and max pooling details of the face are still conserved. We can see that some filters highlight the outlines and differentiate between the face and background. In the second convolution and max pooling detail starts to get lost and more geometrical shapes are

appearing (smaller filters). On the last convolutional and max pooling you can hardly tell that the filtered image is a face as you can only see geometric patterns.

Lastly, we decided to see what was the output fed from all these convolutions and poolings from the already trained simplified model to the fully connected layers for correctly classified men and women.

As we can observe in *Figure 13* (top row) the man it highlights the top left part of his forehead, we assume that it learned to associate receding hairline with men. On the other side for the woman, we see how it highlights the long hair so we assume it relates long hair to women. For the convolutional one (bottom row) it also highlights the forehead of the man, but this time the middle of it. For the woman, we can see again the importance to the long hair but also her nose and mouth. This might be the reason for the improved accuracy of the convolutional one as it gives importance to more parts of the face when classifying.

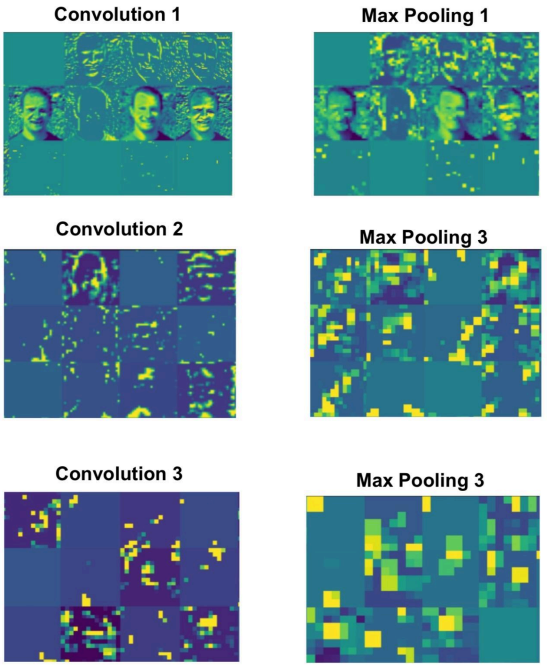


Figure 12. Sample of filters from every convolution and max pooling layer after running a male picture with normal convolutions..

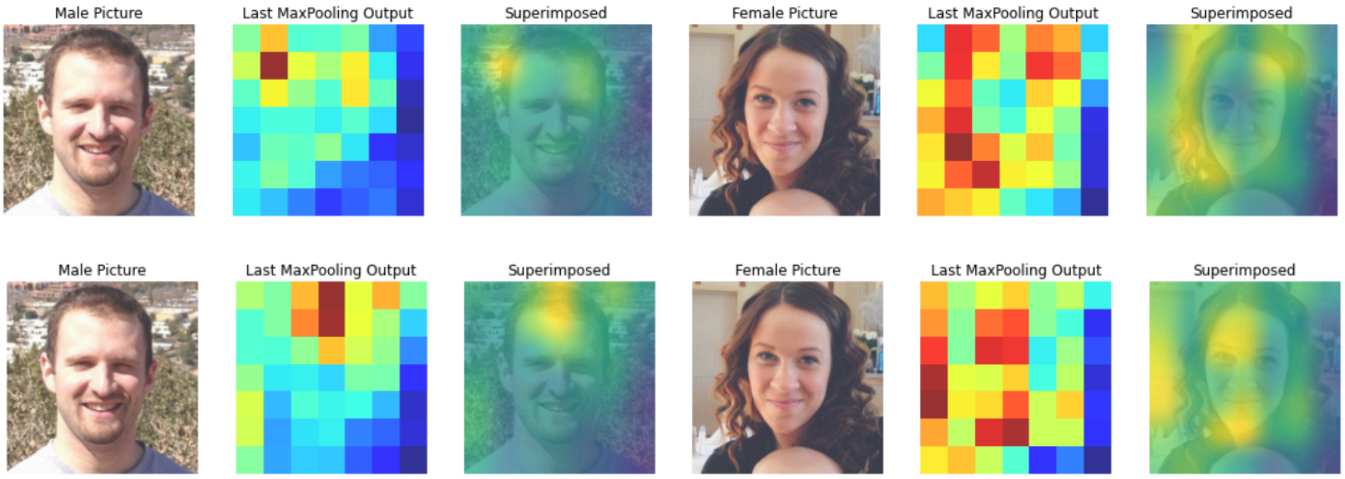


Figure 13. Picture, Last max pooling output, and superimposed of both a man and woman face pictures which were correctly classified. The top row is by the pre-trained simplified model with regular convolution and the bottom row is with the last convolution being dilated.

3.2 CNN Models for Age Recognition

3.2.1 Traditional Model Performance (EfficientNetB5)

In this section we aimed to perform classification for age instead of gender. Again, we decided to use EfficentNetB5 to perform a benchmark for our future models.

We used dataset 1 as the class for the age and obtained a validation accuracy of 64.25%. EfficientNet comes from its innovative approach to balancing model size (number of parameters) and accuracy [3], achieving state-of-the-art performance with relatively fewer parameters compared to other architectures.

However, the complexity of this model is still high and we will try to reach this benchmark with a similar approach as for the gender section.

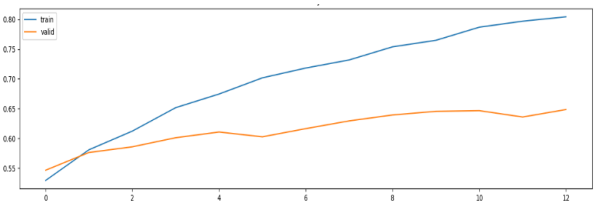


Figure 12. Sample of filters from every convolution and max pooling layer after running a male picture with normal convolutions..

3.2.2 Testing the Simplified Model

Our first approach was to reuse the simplified model by replacing the binary predicting layer with a sparse categorical cross-entropy layer. We ran the model in both data sets, with different batch sizes:

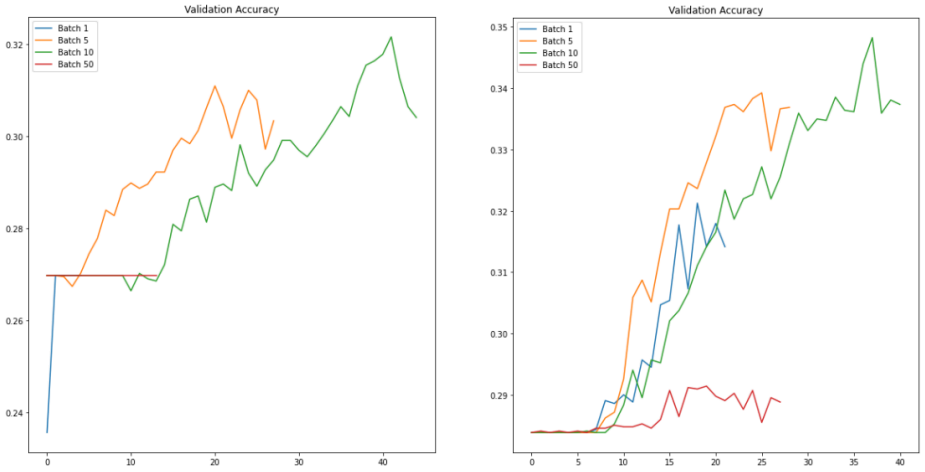
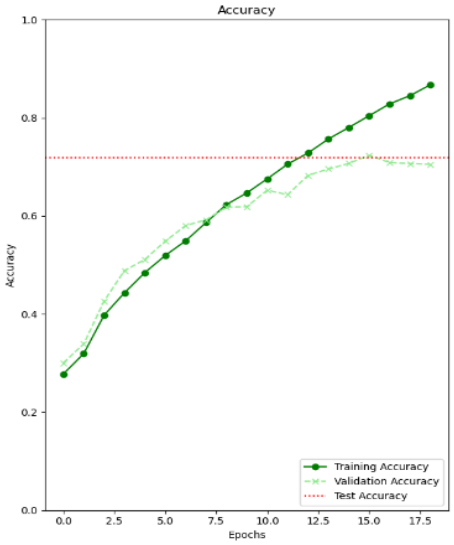


Figure 13. Validation accuracies for age prediction using simplified model for dataset 1 (left) and dataset 2 (right).

As it can be observed the accuracies obtained were low for every batch size in both datasets. We ran some predictions and saw where most of the errors were. Ages from 23-54 showed a lot of errors as they are probably more difficult to classify.

We decided to try again, reducing the age classes to four. To make each class the same size we create. *baby* (0-6), *teen* (8-23) from teenager, *young* (23-35) and *adult* (35+). This split will be referred to as **dataset 3**. The class sizes remain the same. After applying the simplified model for gender prediction and optimal parameters, we obtained a significantly better accuracy. As shown in Figure 14, we obtain a test accuracy of around 71%. This accuracy is really interesting as it matches the predictions done in [6]. Nonetheless, this is achieved by reducing the range of precision which is not ideal. We could imagine that with a better data set and a bigger amount of data, we could achieve with the same model a much better accuracy for more classes.



Another possible solution, that we didn't approach in this assignment could be using shifting cropping and image deformation to create a bigger data set for some range of ages

Figure 14. Validation accuracy for the simplified model predicting four age classes (dataset 3).

3.2.3 Model for gender and age recognition

Seeing the results of age prediction the only way to create a combined model is using dataset 3. Using a dataset as CelebA as pre training could also be interesting but this data set doesn't contain any age variable. We could have created two output-dense layers for computing these predictions but we preferred to combine both variables under one by creating one with two digits. The first digit is binary and represents the gender while the second one represents the age class.

Running the simplified model with the optimal parameters we obtain a similar accuracy than for age, around 70%. As it is displayed later in conclusions some predictions have no error in age and others have errors in both age and gender predictions. This means that if we looked at the variables separately we might have better accuracies. However, merging the two variables in one enables the model not to overfit the gender variable and adjust more to the age one. It would be interesting to compare the results with a model that doesn't merge the two variables in one.

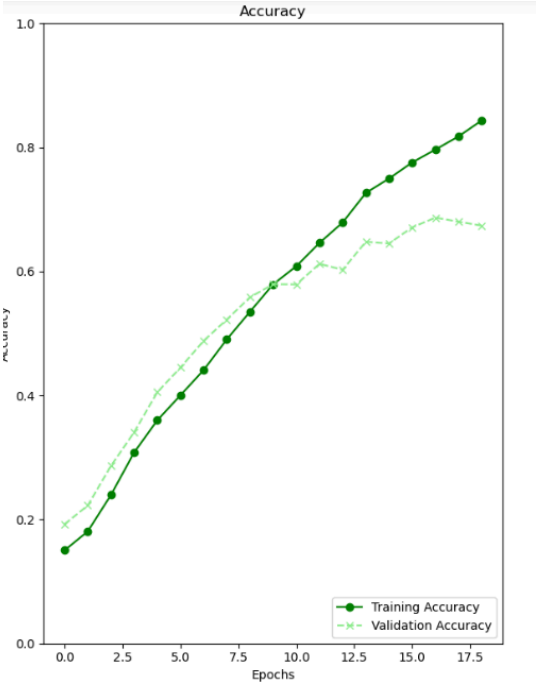


Figure 15. Validation accuracy for the simplified model predicting both age and gender.

3.2.4 ViT Model.

One of the main characteristics of ViT is that the model has to be trained with an enormous amount of data. The data should be of the order of 100 million. The use of openIA has increased exponentially. ChatGPT is a well-known tool of the classifier. For us creating a ViT model from scratch was nonsense as the computer could not handle this amount of data. That is why we decided to implement the Open AI model used for Dalli [10].. This is a practical way to use ViT and understand how it works. We will use their zero-gradient classifiers.

The model has already been trained with a large amount of data from very uncorrelated datasets. The data consists of text and images that are associated with each other. The prompts for the images are usually created by writing *"This is a photo of …",* The sentence is completed with a description of the image. That is why we write eight prompts separated by gender and 4 groups of age. We run the model that we imported. The models can be selected from some of them. The options consist of ResNets that are not interesting for us, and the ViT L for large and B for a smaller size of ViT. We select 32 as the patch size. We measure the accuracy and see the results. We can see how powerful this model is but the difference between our model implemented is that the pretraining consists of more than 100 million images.

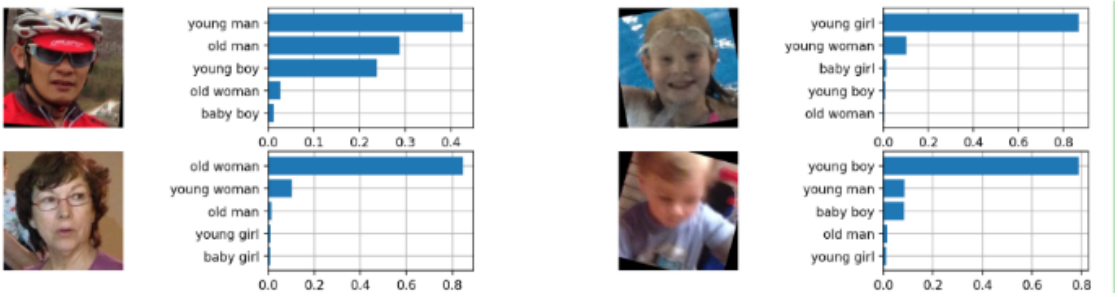


Figure 16. Examples of some predictions done by the ViT model using random Adience pictures.

4. Conclusions

Model	Validation Accuracy	Classification Use	Data Split
EfficentNetB5	88.65%	Gender	All
Simplified Model RGB	89.92%	Gender	All
Simplified Model RGB Gaussian Blur	87.67%	Gender	All
Simplified Model GrayScale	86.91%	Gender	All
Simplified Model GrayScale Gaussian Blur	87.59%	Gender	All
Simplified Model Pretrained	87.03%	Gender	All
Simplified Model 1 Convolution Variation	88.54%	Gender	All
Simplified Model 2 Convolution Variation	78.57%	Gender	All
Simplified Model 4 Convolution Variation	88.71%	Gender	All
Simplified Model 5 Convolution Variation	88.30%	Gender	All
Simplified Model Dilated Convolution	90.27%	Gender	All
EfficentNetB5	64.25%	Age	Dataset 1
Simplified Model	30.26%	Age	Dataset 1
Simplified Model	33.71%	Age	Dataset 2
Simplified Model	70.63%	Age	Dataset 3
Simplified Model	67.4%	Age&Gender	Dataset 3

Table 1. Validation accuracies for all the models tested using batch size 5.

In *Table 1* we can see the validation accuracies of all the models tested in this work using batch size 5, as we found it to be optimal. For predicting age, the best performing model was the variation of our simplified model where we change the last convolution to be dilated, with dilation rate 2. In *3.1.2.4* we highlighted how we can see that the dilated convolution used different parts to classify as men or women which can be the cause of the slight improvement with respect to the simplified regular model.

For age prediction the best accuracy obtained was when using the simplified model with only 4 classes for the age variable (dataset 3). Our previous attempts with dataset 1 and dataset 2 were not successful, so we decided to reduce the number of classes. However, despite the increase in accuracy this reduced drastically the precision with which the model predicted age as the intervals were now bigger. Lastly, for the combination of age and gender we obtained a similar accuracy than in [6] so we conclude our work to be successful with some predictions shown in *Figure 17.*

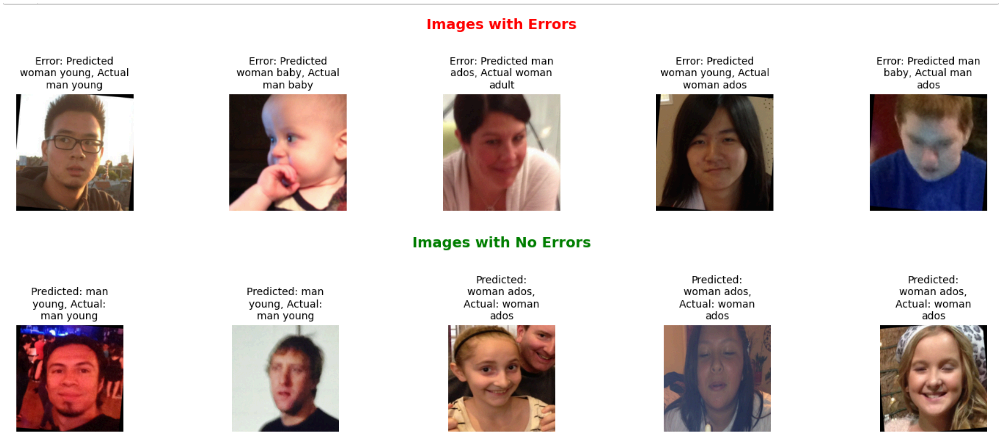


Figure 17. Correct and incorrect predictions of the simplified model for age and gender.

Finally, we wanted to reflect on how we found this project to be limited by computational power and time. Other papers we read worked with much larger data sets as well as more complex models that our computers could not handle. Despite this, we are happy with our results as we were able to predict gender and age.

References

[1] Levi, G., Hassncer, T.: Age and gender classification using convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 34–42. IEEE, Boston (2015) doi: https://doi.org/10.1109/CVPRW.2015.7301352

[2] Li, J.; Qiu, T.; Wen, C.; Xie, K.; Wen, F.-Q. Robust Face Recognition Using the Deep C2D-CNN Model Based on Decision-Level Fusion. Sensors 2018, 18, 2080. doi: https://doi.org/10.3390/s18072080

[3] A. Sarkar, “Understanding EfficientNet — The most powerful CNN architecture,” Medium, May 08, 2021. https://arjun-sarkar786.medium.com/understanding-efficientnet-the-most-powerful-cnn-architecture-eaeb40386fad

[4] Chatfield, Ken, et al. “Return of the Devil in the Details: Delving Deep into Convolutional Nets.” Arxiv.org, 14 May 2014, arxiv.org/abs/1405.3531.

[5] Zhu, Zhiyu, et al. When Residual Learning Meets Dense Aggregation: Rethinking the Aggregation of Deep Neural Networks. https://arxiv.org/abs/2004.08796

[6] Levi, Gil, and Tal Hassner. Age and Gender Classification Using Convolutional Neural Networks. https://openaccess.thecvf.com/content_cvpr_workshops_2015/W08/papers/Levi_Age_and_Gender_2015_CVPR_paper.pdf

[7] Agarwal, Prerak. “Age Detection Using Facial Images: Traditional Machine Learning vs. Deep Learning.” Medium, 2 July 2020, towardsdatascience.com/age-detection-using-facial-images-traditional-machine-learning-vs-deep-learning-2437b2feeab2.

[8] S. Yang, P. Luo, C. C. Loy, and X. Tang, "From Facial Parts Responses to Face Detection: A Deep Learning Approach", in IEEE International Conference on Computer Vision (ICCV), 2015

[9] Adri Priadana, et al. “A Facial Gender Detector on CPU Using Multi-Dilated Convolution with Attention Modules.” 2022 22nd International Conference on Control, Automation and Systems (ICCAS), 27 Nov. 2022, https://doi.org/10.23919/iccas55662.2022.10003722

[10] “CLIP: Connecting Text and Images.” Openai.com, openai.com/research/clip