DEEP LEARNING

MASTER DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS

Face Mask Classification :– “Are you wearing mask?”

Dataset divided in train, validation, and test folders accessible here.

Ana Marta Silva | m20200971

Gustavo Tourinho | m20180846

Lorenzo Pigozzi | m20200745

Salim Bouaichi | m20200547

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

Contents

1. Abstract

This work is an academic group project for the course of *Deep Learning*, Master in Data Science and Advanced Analytics at *Nova IMS*.
The idea that gave birth to this project comes from the particular situation that is impacting the world nowadays, the Coronavirus pandemic. The virus has changed our daily lives and habits, and one of the symbols of this new reality is represented by this new "cloth" that all of us should wear in public spaces: face masks.
Face-detection is a problem where Machine Learning and, in general, AI have developed, with significant results being achieved, such as high probabilities of recognition.
But what about if half of the face would be covered? Could we be able to help the authorities to discover if people are wearing the mask in public spaces, as the social restrictions require?
In other words: are people actually wearing masks? Can machine learning help on answering this?
With these questions and inspirations, we start the creation of our model.

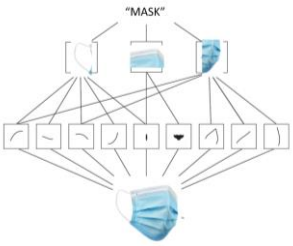
2. Problem Statement

We started by trying to create a model that when taking a picture as a whole would be able to classify it has with mask or without mask. To achieve the desired results, we built a Convolutional Neural Network(CNN).
Our specific problem is a binary classification challenge, in which we have 2 classes for the output: "mask" or "no mask".

Objective:

The challenge we want to complete is creating a model than can automatically classify if, given an image input, it is picture a picture of people wearing or not a mask.

The convolution operation



One of the key points identified for this classification problem's success was that our model would have enough and well trained filters capable to perceive the key characteristics that are present in a picture with and without mask.
As for evaluation measure, in order to assess the success of our model, we will use the metric accuracy which will indicate the capacity of the model in classifying correctly pictures with and without mask. Additionally, we will compare these results with the ones obtained when using a pre-trained model, one of the most famous in the computer vision field: the VVG 19.

3. Sources and data

Images MASK

Source website: https://www.kaggle.com/andrewmvd/face-mask-detection/metadata
This dataset contains images of a single person with a mask and others with a group of people wearing masks with different backgrounds and positions of the person/people in the image.
1411 images of people with masks are considered for this project.

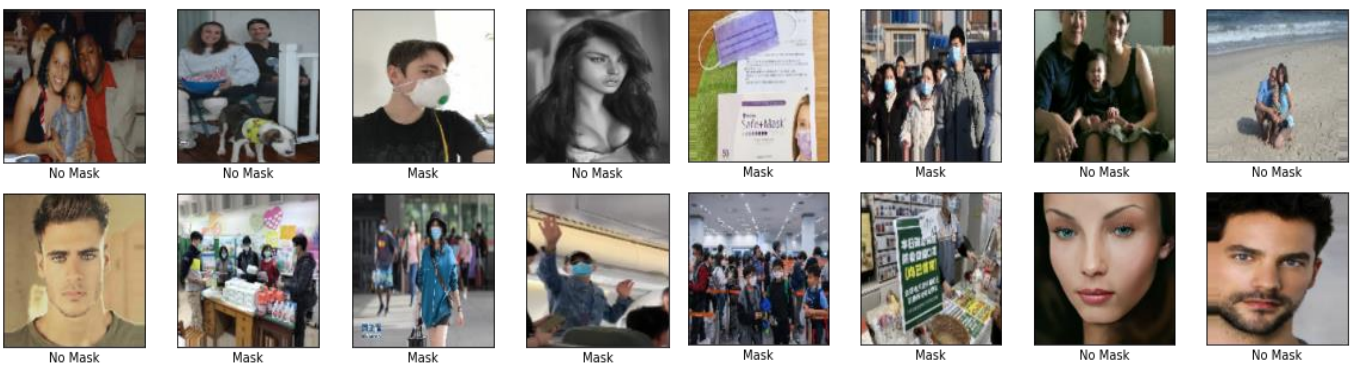
Images NO MASK

For images of people with no mask, we select images from 2 different datasets. The reason is simple; in order to match the complexity of the pictures with mask and achieve a similar level of noise compared with the pictures with mask. The first dataset has images of individual faces while the second dataset have pictures with more than one person per picture and with different backgrounds.
The source of the dataset with close-up faces of people without a mask is:
Source: https://www.kaggle.com/ashwingupta3012/human-faces?select=Humans
From this dataset, 658 images were selected.
The dataset with groups of people without masks is the following.
Source: http://chenlab.ece.cornell.edu/people/Andy/ImagesOfGroups.html

From this dataset, 753 images were randomly chosen.
Consequently, the total number of images of our dataset is 1411 images for the mask class and 1411 images for the no mask class, consisting of a total of 2822 pictures in the dataset.
Furthermore, to complete our analysis properly, we split the dataset into train, validation, and test sets. We chose the following percentages for the splitting:

- Train: 70 % 1978 images
- Validation: 15 % 422 images
- Test: 15 % 422 images

The final dataset already divided in the train, validation, and test folders can be found here.
An example of the type of pictures used can be seen in the figure below:



4. The modelling of the task

Intending to solve the image classification problem described above, we decided to apply a category of neural networks, the Convolution neural network (CNN).

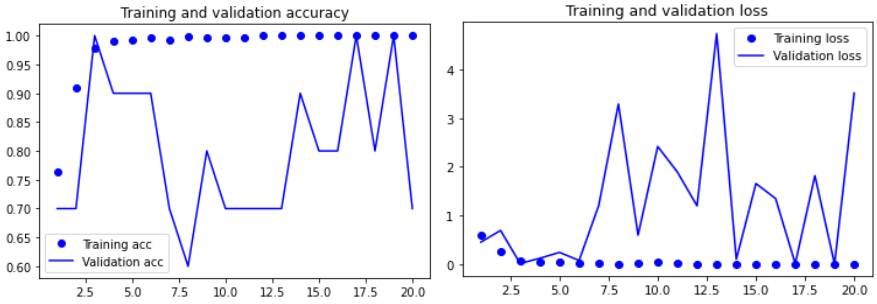
Firstly, we decided to design the simplest model possible. We started with one 2D convolutional layer consisting of 32 filters with a height and width of 3 and an input shape of 150 x 150 pixels and a channel (depth) of 3 because our pictures are colored. This layer will be responsible for extracting the features from the images. The activation function was the Rectified Linear Unit (Relu). Secondly, we added the pooling step consisting of the max spatial pooling with a stride of 2. We added a flattening layer and the fully connected layer, constituted by a layer of 32 neurons and as activation function the Relu. The output layer was constituted by one neuron, and the activation function was the sigmoid.

We used the optimizer Adam, the binary cross-entropy for the loss function, and the metric accuracy for compiling the model.

This architecture resulted in approximately 5 million trainable parameters.

Before running the model, we proceeded to rescale the pictures. For training the first model, we decided on 20 epochs, and 10 validation steps (both parameters will be kept equal to these values through all the tunning of parameters until the callback is applied) and 1978 steps per epoch,

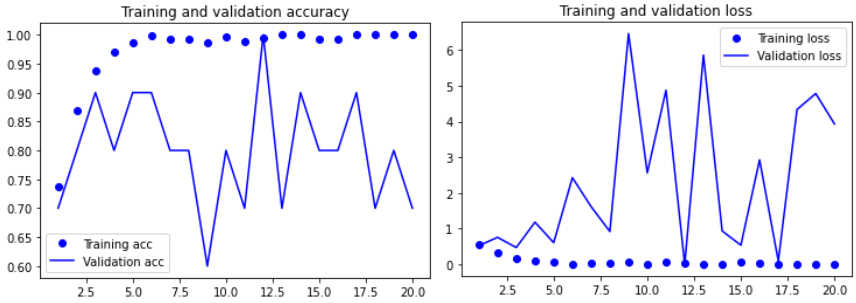
The first results obtained were a training accuracy of 1 and a validation accuracy of 0.84 in the last epoch. From this initial trial we identified complete overfit in the training dataset and many spikes in the validation accuracy and loss which can indicate a very diverse dataset.



5. Tuning the Model

5.1. Convolutional steps

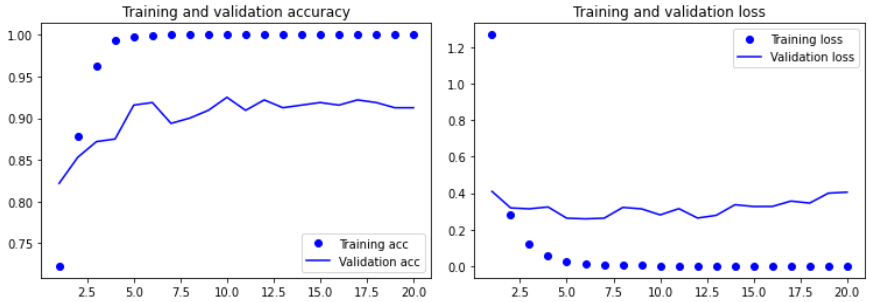
On top of the initial model that consisted of only one convolutional step, we decided to experiment the impact of adding one more convolutional layer and max pooling step, keeping all the above mentioned parameters equal, with exception that this time the convolutional layer would have 64 filters. The results obtained were a training accuracy of 0.99 and a validation accuracy of 0.78 which was lower than the one of the model with only one convolutional step. Moreover, this architecture showed a more erratic behavior in the graphs of the accuracy and loss, therefore we decided to continue our tunning endeavor with one convolutional step.



5.2. Understanding the Spikes – Batch Sizes

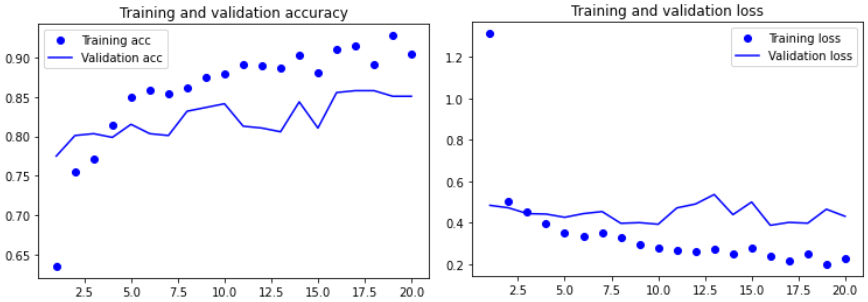
As mentioned, we identified in our initial graphs a very high number of spikes in the validation loss and accuracy. Consequently, we tried to solve this problem by changing the size of the batch. We considered the following batches: 23, 32, and 46. We chose bigger batch sizes because it is associated with higher accuracy.[1]

From these trials, we concluded that the best batch size is 46 based on the behavior of the model in the loss and accuracy graphs as it was the one that produced the most smoothing effect in the spikes that we wanted to solve. This new defined batch size of 46 will have implications in the steps per epoch which will become lower at 43, and the validation steps which will be approximately 9.



5.2 Data Augmentation

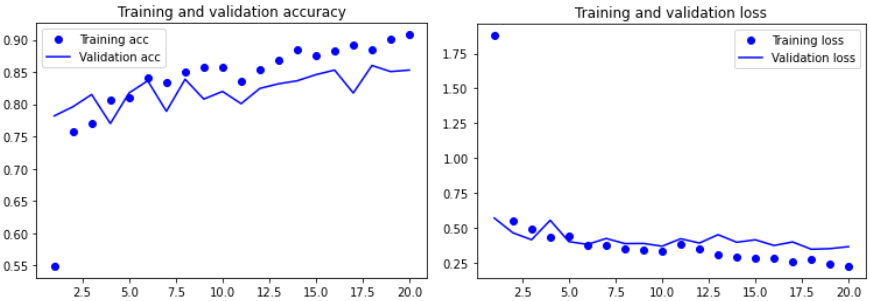
In order to solve the clear problem of overfitting we decided to try the regularization technique of data augmentation in order to increase the diversity of the training set. We defined parameters rescale to 1/255, shear which distort the angle of the picture to 0.2, zoom to 0.2 which means the zoom will be 0.8 and the zoom out will be 1.2; horizontal flip = True which will reverse the input horizontally. [2] From this tentative we obtained a training and validation accuracy of respectively: 0.90 and 0.87. These results and the graphs showed that this technique allowed to reduce the problem of overfitting (the gap between the validation loss and the training loss became smaller).



5.3. Filters

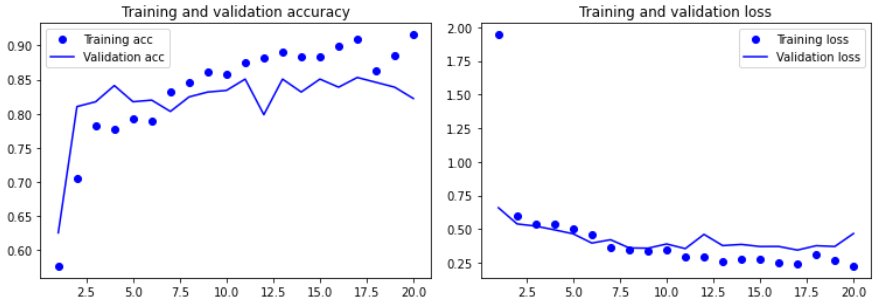
Following, in our tunning activity, we decided to try to increase the number of filters in the model to boost the number of features extracted in the training of the model and impact the accuracy of our model. We tried the following numbers: 24, 64, and 96. We concluded that 96 filters were the best option as it allowed for the highest validation accuracy at 0.85 and a training accuracy of 0.90.

Filters	**24**	**64**	**96**
Training Accuracy	0.88	0.88	0.91
Validation Accuracy	0.82	0.81	0.85



5.4. Dropout

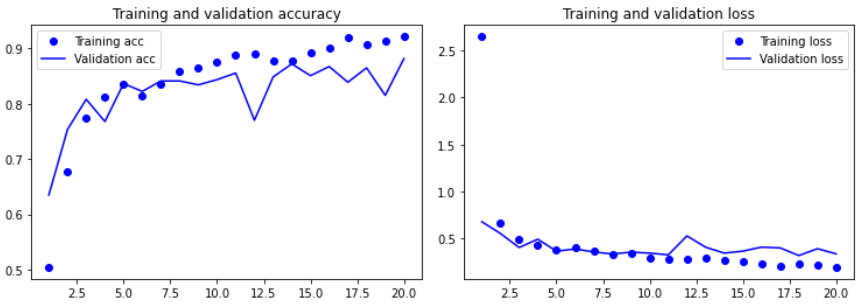
However, we were still not satisfied with the accuracy in the training, which was still excessive, consequently, we decided to try the regularization technique of dropout after the flattening layer. For this trial, we applied different probabilities values for probability of the outputs not being considered, such as: 0.20, 0.25, 0.30, and 0.50. After analyzing all the options, we concluded that the best dropout probability was 0.3 because of the behavior of the curves in the accuracy and loss graphs which showed us that it allowed for the lower overfitting and the highest validation accuracy through all epochs.



5.5. Dense Layer

Moreover, in the consisting pursuit to improve the validation accuracy of the model, we decided to try a different design for the fully connected layers of the neural network. We decided to try with 2 hidden layers, instead of only one has had been the case until now. The first layer would have 64 neurons and the second one 32 neuron. Additionally, we also tried different values for the number of neurons when considering only one hidden layer, specifically 64 and 128. We concluded that the best design was to keep only one hidden layer with 128 neurons as it was the trial that awarded us with the highest validation accuracy score.

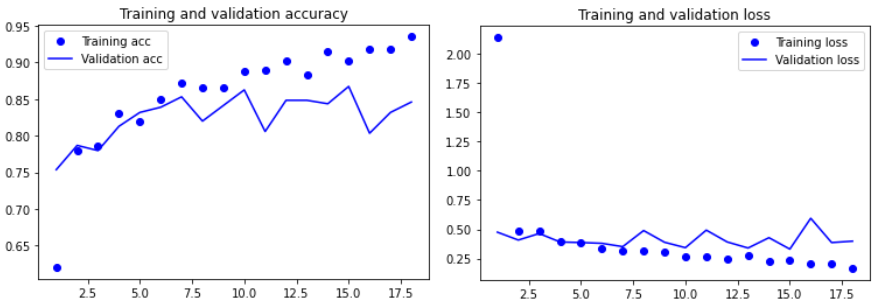
Dense Layer Neurons	**2 layers 64 & 32**	**One hidden layer with 64**	**One hidden layer with 128**
Training Accuracy	0.94	0.95	0.92
Validation Accuracy	0.85	0.86	0.88



5.6. Callback

Furthermore, in order to improve the efficiency of our training, we decided to include a call back earlystopping that would stop the model from training when the validation loss would stop decreasing, limiting in this way the overfitting. We tried many values for the parameter patience (2, 3, 5 and 7) because of our knowledge of the high noise existent in our dataset. This patience would allow the model to continue training even if the validation loss increases for a few number of epochs. We concluded that the most appropriate patience number was 3 because it was the one that provided the lowest validation loss at the stopping epoch. This trial stopped the model at 18 epochs, which will become the number of epochs used in the subsequent trainings of the model.

Callback Early Patience	**2**	**3**	**5**	**7**
Validation Loss	0.44	0.39	0.46	0.40



5.7. Optimizer

Following, we decided to test the impact of another optimizer in our model. We decided to try the RMSprop optimizer which is also an adaptive learning rate method.[3] We concluded that the Adaptive Moment

Estimation (adam) was the best optimizer for our model because it awarded us with the highest validation accuracy.

Optimizer	RMSprop	Adam
Training Accuracy	0.92	0.93
Validation Accuracy	0.85	0.87

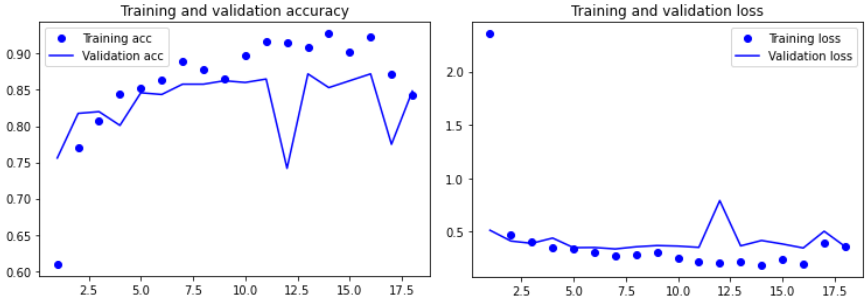
6.Best Model & Error analysis

After all these tunning efforts, we concluded that the best model was the one with the following parameters:

- Number of filters: 96
- Dropout of 0.30
- 1 hidden layer with 128 neurons
- Batch size: 46
- Number of epochs: 18
- Optimizer as Adam

In order to assess the performance of this final model, we defined 2 different approaches: testing in on unseen data and using transfer learning. We started with the first approach by running the best model with the final parameters and a checkpoint callback in order to save the model that allowed for the highest accuracy in the validation set. We obtained a training accuracy of 0.84 and validation accuracy of 0.85. We should remember that our first model awarded us with a training accuracy of 1 and a validation accuracy of 0.84. So it is clear the improvement, specifically in the treatment of the overfitting.
Following, we loaded this best model and made the predictions on the test set. We achieved a test accuracy of 0.83. This high result is backed by the values of true positives and true negatives (in green) that were achieved in the confusion matrix seen below.

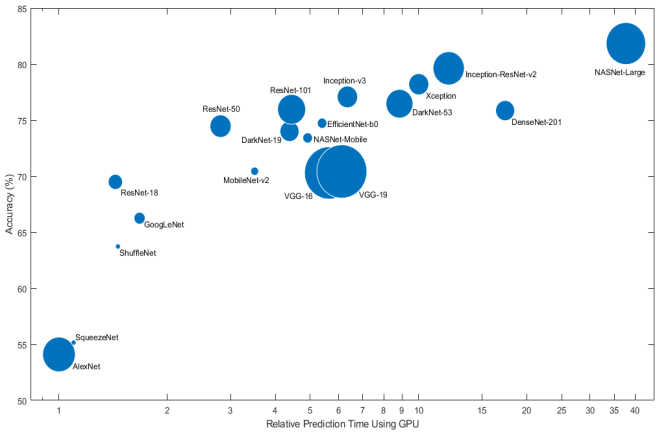
	Predicted - No Mask	Predicted - Mask
Actual - No Mask	196	15
Actual - Mask	56	155



From the analysis of the graphs we can see that the train and validation accuracy are very close and have an upward direction with few exceptions that we justify as being relates to the nosiness of the dataset. On the other hand, considering the loss graph, the loss in the training and in the validation have a decreasing tendency through the epochs and then tend to stabilize with a small gap between them, with exception to a few points which again we explain by the noisy that exists in the dataset of images with masks. These results indicate that the model is not overfitting, instead this proximity between the line reveals the good generalization ability of our model.

7.Transfer Learning - Feature Extraction

Next, as already mentioned, we decided to compare our model created from scratch with the results of using transfer learning with feature extraction. It consisted in utilizing a pre-trained model in our convolutional steps and keeping our design for the fully connected neural network. We aimed with this to benefit from the representations learned of a model trained in a much bigger dataset. The pre-trained model used was the VGG 19 which is a convolutional neural network with 19 layers, which was trained in more that one million images from the ImageNet database and can classify images in 1000 categories.[4] This model has a balanced performance in terms of speed and accuracy as it is visible in the picture below, that is why we selected it.[5] The results obtained with 5 epochs (with callback earlystopping) were: training accuracy of 0.999, a validation accuracy of 0.957 and a test accuracy of 0.96 The result on the test represents a 13 percentage point improvement (16% better) comparing with our best model test accuracy. However, this model suffers from complete overfitting.



8.Conclusion

All in all, the results prove that the model that we designed to solve the image classification problem of mask or no mask is able to classify correctly with 83% of probability. Besides, we were able to solve the problem of overfitting by applying several regularization techniques, which awarded us with a model with high generalization power. Comparing with the feature extraction experiment, our model continues to hold as a good option, considering it is only 16% less competent in classifying than the pre-trained model.

Nevertheless, we experienced some challenges regarding the dataset due to the small availability of real pictures of people wearing mask and the ones found were very diverse in terms of background, number of people in the image and position of elements.

Likewise, we are certain that if we had more computational resources available we would have run each model multiple times and average the results in order to obtain more robust results.

On the other hand, we experienced a steep learning curve in this project, we tried a total of 23 different models, having become surprised with the great capability of the convolutional neural network as we were not expecting so high levels of accuracy from the beginning considering the diversity of pictures that we provided and the fact that it has only one convolutional step.

Consequently, in future works we aim to transform the classification problem proposed into an object localization and detection challenge in which we would aim to draw a box in the pictures in the place of the masks and classify it accordingly.[6]

Finally, this was a great project to have our first experience in the field of deep learning and to realize its incredible potential and applications. We became fans!

9.References

1. Radiuk, Pavlo M. , 2017 “Impact of Training Set Batch Size on the Performance of Convolutional Neural Networks for Diverse Datasets” Information Technology and Management Science. vol. 20, pp. 20–24
2. Keras, Image data preprocessing. Accessed on 3rd of April 2021 <https://keras.io/api/preprocessing/image/>
3. Keras, *RMSProp.* Accessed on 2nd of April 2021 < https://keras.io/api/optimizers/rmsprop/>
4. MathWorks, *vgg19*. Accessed on 3rd of April 2021 <https://www.mathworks.com/help/deeplearning/ref/vgg19.html>
5. MathWorks, *Pretrained Deep Neural Networks.* Accessed on 3rd of April 2021 <https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html>
6. Ganz, Roy, “Object Localization using Keras”.Analytics Vindhya,13th July 2020 < https://medium.com/analytics-vidhya/object-localization-using-keras-d78d6810d0be>