Biologically Inspired Artificial Intelligence

Final report

Gender recognition

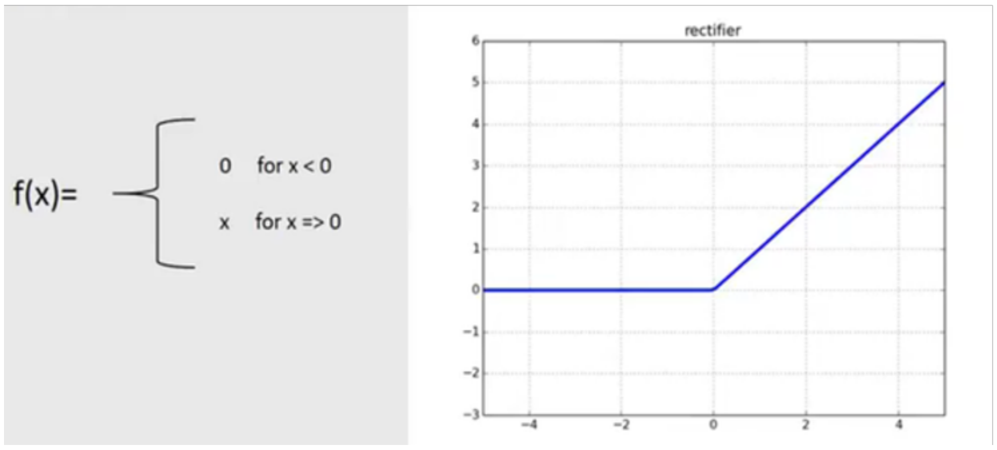
Skład sekcji:
Adam Warzecha
Mikołaj Malich

1. Introduction

The main purpose of this project was to build artificial intelligence able to recognize gender from previously prepared human face's images. To implement and test algorithms we used Kaggle notebook in cooperation with Tensorflow, SKLearn, MatplotLib, NumPy and OpenCV.

2. Task analysis

We started working on the project by analyzing the topic and reading about details of implementing artificial intelligence. Then we found dataset that met our expectations. It was divided into 2 parts: training set (about 47000 photos) and validation (test) set (about 11500 photos).
Each part of dataset contained 2 folders - one for women faces images and another one for men's in a ratio of 50:50. All the photos were imported to algorithm straight from Kaggle cloud database due to its size - ca. 300MB. One of the biggest advantages was that provided dataset contained only prepared, preprocessed images with human faces, so we could focus on gender recognition algorithm. The most important part of our project was to find the right approach to solve the problem - we decided to use 5 convolutional layers in which first one contained 32 filters, next two ones 64 filters, the last two included 128 filters. The main advantage of CNN is that it automatically detects the important features without any human supervision.
For example, given many pictures of cats and dogs it learns distinctive features for each class by itself. CNN is also computationally efficient and the final output is a vector.
To reduce spatial dimensions of the output volume we used Max pooling 2D.
It is a layer that is used to reduce the size while preserving the most important information.
Another helpful thing in our model was the ReLU activation function - it outputs the input directly if it is positive, otherwise it will output zero. In the beginning we used other model based on transfer learning with similar behavior but we had to change it due to continuous problems ex. with generating charts.



The ReLU activation function diagram

3. Internal and external specification

Photos from dataset are recursively loaded to program thanks to Glob. Then they are converted to standard python arrays. During this operation we use OpenCV functions like resize or imread. In case of data structures, we used NumPy arrays for training, test and prediction data (previously mentioned photo arrays are also moved to new ones). Another thing worth mentioning is Adam optimizer which modifies the attributes of the neural network ex. learning rate or decay. ImageDataGenerator, helped us in training deep-learning model, allowed us to extend and modify the dataset. Matplotlib.pyplot is used to generate charts in which we present train results of our model. Confusion matrix was made with Seaborn heatmap function. Project does not contain its own graphic user interface, all operations are performed in Kaggle notebook and the results are shown in output console.

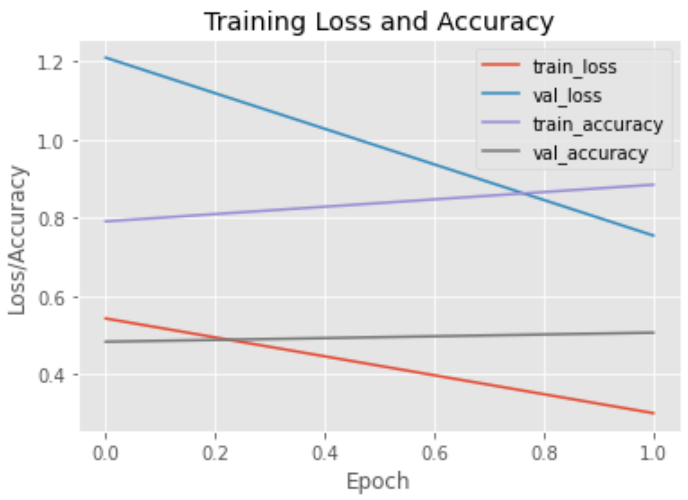
4. Experiments

At the beginning of our experiments, we set epochs to low values due to long training times. We found articles about optimal batch size values for deep learning model working on images. Basing on our research, we started with bath size value equals to 128. Because of too much memory allocation during preprocessing, we did not use the entire dataset to train the model.
Below, we shown our results and details of our experiments:

1.

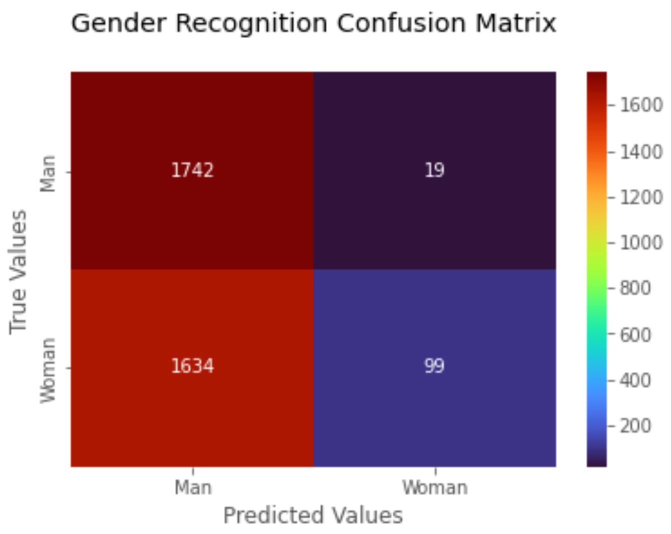
epochs	batch size	train set	validate set	loss	accuracy	val loss	val accuracy
2	128	14102	3494	0.3016	0.8849	0.7552	0.5072

4.1.1 Table with training parameters and results



4.1.2 Training results chart

We noticed decreasing tendency of parameters val_loss and train_loss.
The 75% val_loss result was quite high and the 50% of val_accuracy was not satisfying.
Too few epochs made the chart practically useless.



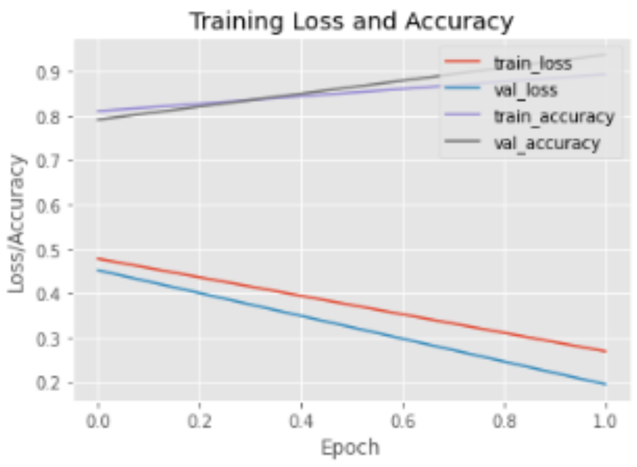
4.1.3 Prediction results

After testing the this model with previously unused photos, the results were terrible.
It correctly predicted only 118 out of 3,494 photos, giving the result of 3.5%.
Only 6% of female faces have been well recognized comparing to 1% of male faces.

2. To improve the training results of our model during next trainings, we experimentally changed the batch size to 32:

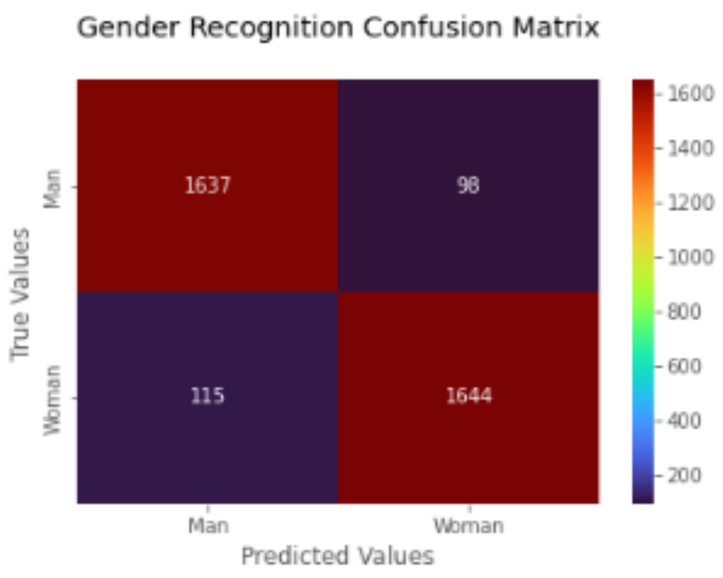
epochs	batch size	train set	validate set	loss	accuracy	val loss	val accuracy
2	32	14102	3494	0.2690	0.8937	0.1947	0.9376

4.2.1 Table with training parameters and results



4.2.2 Training results chart

In comparison to previous results, our results were satisfying. Effectiveness during validation increased by almost 95% and losses also dropped enormously.



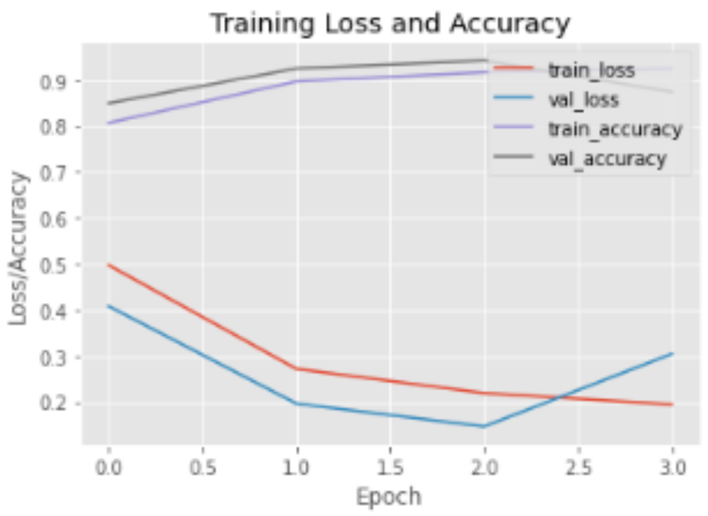
4.2.3 Prediction results

The results were almost identical to the training results. Correct predictions for photos of female and male faces hovered around 93%.

3. We started to slightly increase the number of epochs:

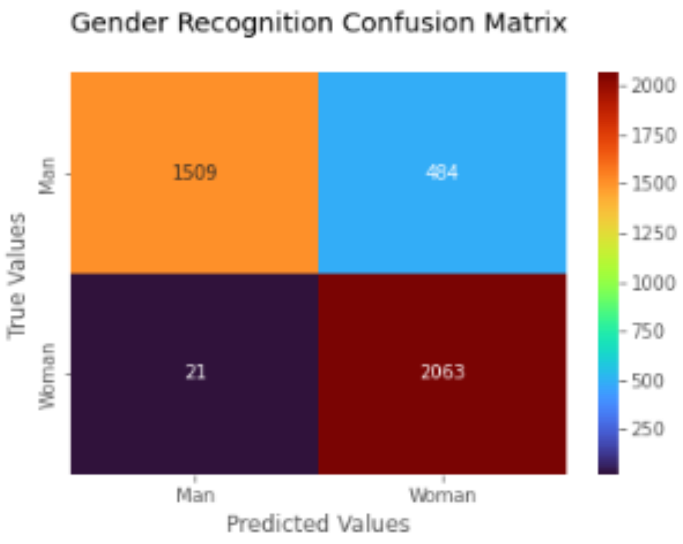
epochs	batch size	train set	validate set	loss	accuracy	val loss	val accuracy
4	32	16453	4077	0.1941	0.9250	0.3041	0.8739

4.3.1 Table with training parameters and results



4.3.2 Training results chart

As we expected, the chart was still far away from perfect, however it can be seen that the linear functions are no longer constant. After some research, we found out that the model tries to use different approach in training and validation.



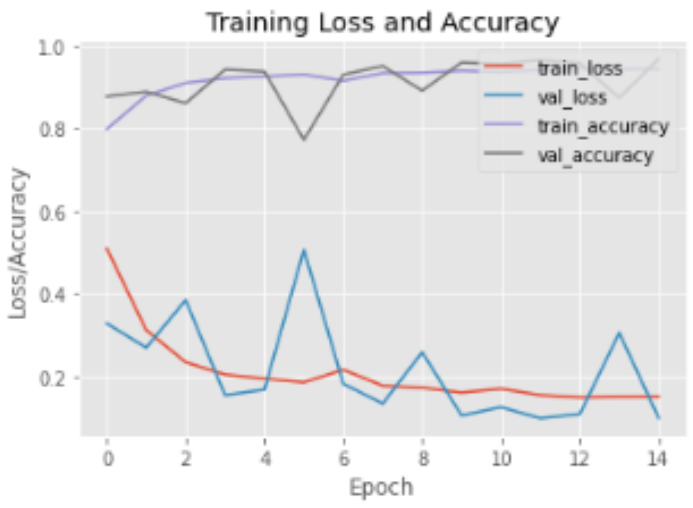
4.3.3 Prediction results

After the predictions, the overall score was consistent with validation score after training. However, compared to the previous confusion matrix chart ,the number of correctly recognized male photos fell noticeably. The overall number of photos of both sexes was similar, so the problem caused by disproportion can be excluded. Moreover, we can conclude here, that the change of approach during training resulted in worse results in recognizing male faces.

4. To allow the model choose a better way to recognize genders, we significantly increased number of epochs:

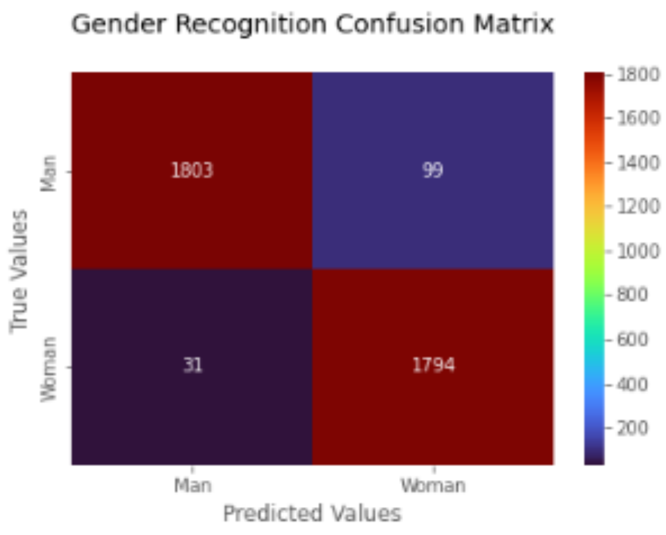
epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
15	32	15042	3727	0.1502	0.9435	0.0986	0.9678

4.4.1 Table with training parameters and results



4.4.2 Training results chart

In this case, model tried several different ways. It ended up with better results than before.



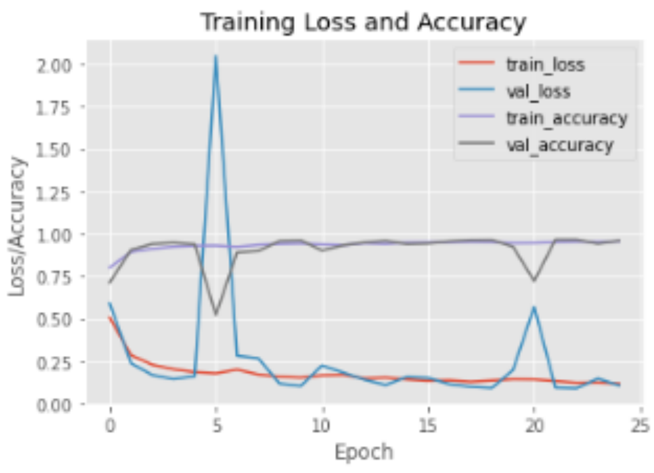
4.4.3 Prediction results

Increasing the number of epochs in this case almost blurred the differences between the percentage of correct prediction. The woman' faces were better recognized by about 4%.

5. In the hope of further improvement, we increased the number of epochs again and modified the number of batch size for testing:

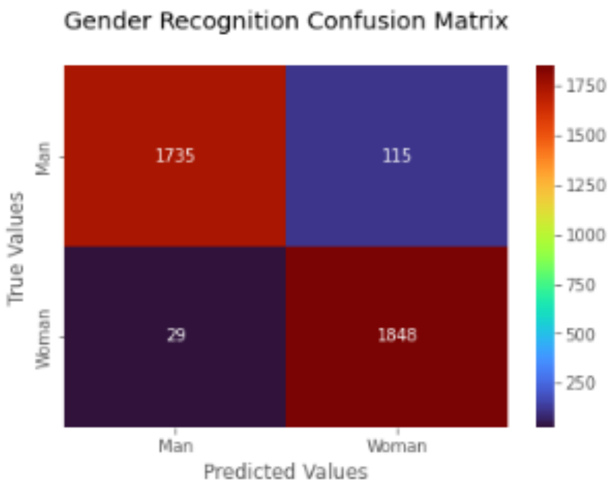
epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
25	64	15042	3727	0.1188	0.9558	0.1080	0.9603

4.5.1 Table with training parameters and results



4.5.2 Training results chart

Changing the batch size resulted in faster selection of the appropriate approach for training. In fifth epoch, model chose the wrong way to recognize gender and then realized its mistake in no time. In the further epochs, it also tried to change the approach, this time less drastically, however it had little effect on the final result.



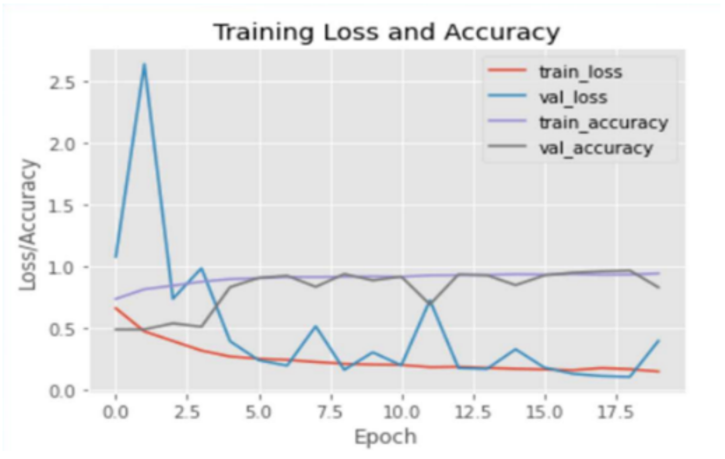
4.5.3 Prediction results

The efficiency of the model remained almost the same as before, unfortunately the minimal differences between the correct gender recognition were still present.

6. We decided to test the behavior of the model during training on a reduced set of photos. To speed up even more, the number of epochs has been reduced:

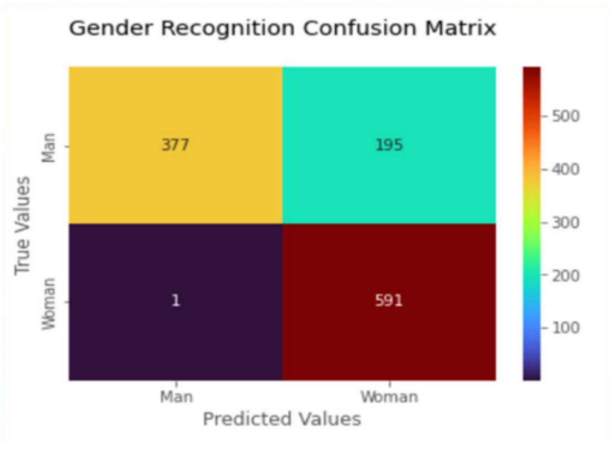
epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
20	64	4700	1116	0.1511	0.9443	0.3984	0.8316

4.6.1 Table with training parameters and results



4.6.2 Training results chart

As we expected from our previous observations, the accuracy of the model was lowered. Here we wanted to explore the differences between gender specific recognition.



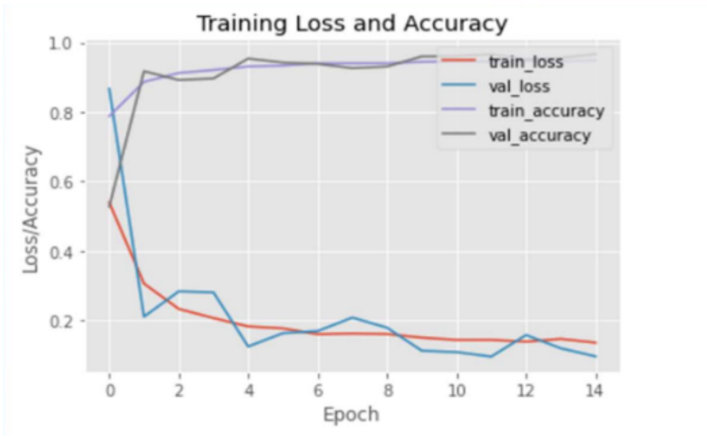
4.6.3. Prediction results

Even though the data sample was not huge, we did not expect such big differences here. The effectiveness of recognizing women is noteworthy, almost 100%.

7. **We were right to conclude, that in case of gender diagnosis, it is more difficult to recognize men. So we decided to experiment a little more. We just reverted back to the previous dataset size but decreased the number of epochs to 15 to finish training before the change of approach seen in the chart 4.5.2:**

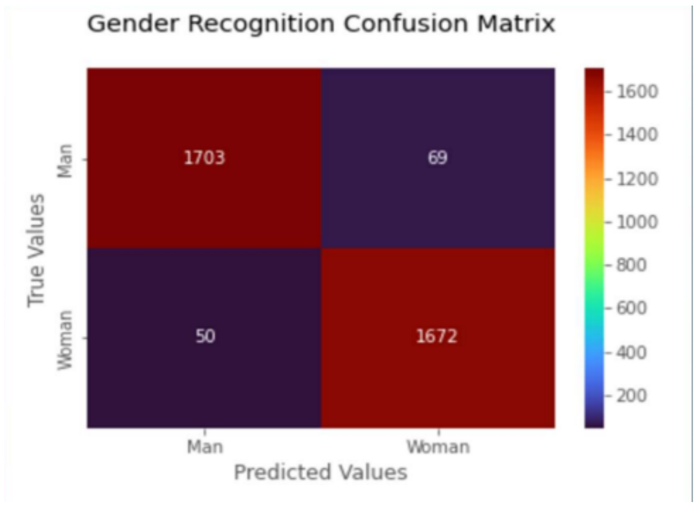
epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
15	64	14102	3494	0.1359	0.9479	0.0964	0.9659

4.7.1 Table with training parameters and results



4.7.2 Training results chart

This time, methods of selecting the approach to training the model had less drastic impact on the final results.



4.7.3 Prediction results

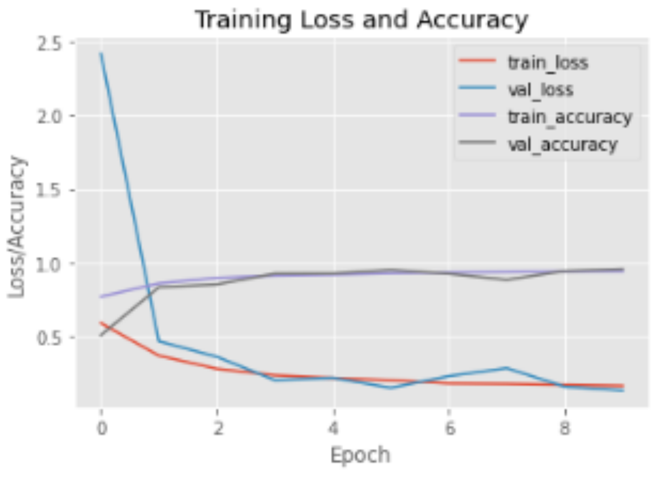
Obtained results as a percentage:

correct overall: 96,59%
correct man: 96,11%
correct woman: 97,10%

8. This time it was possible to blur the differences between correctly recognized photos. In order to check whether it was not a coincidence, we decided to train the model with as many photos as the Kaggle environment allowed. Unfortunately, with more epochs and increased number of images, the environment reported an error and we had to train again:

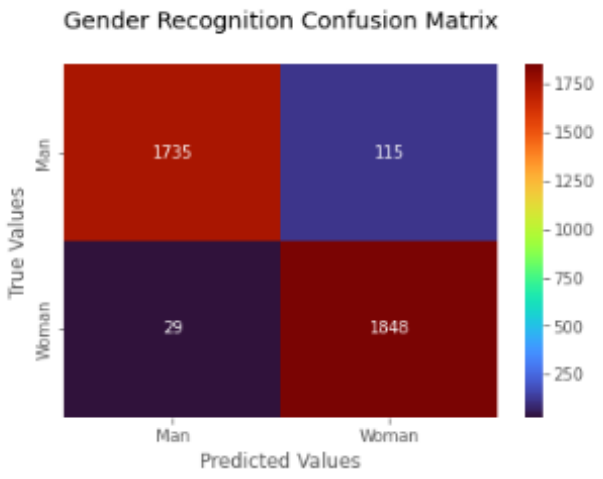
epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
10	64	18 803	4 658	0.1515	0.9406	0.1296	0.9562

4.8.1 Table with training parameters and results



4.8.2 Training results chart

During training, the results remained relatively stable after the model reached a certain value.



4.8.3 Prediction results

Unfortunately, in this case the difference between correctly recognized photos of women was again greater than the men's one (about 4%).

At this stage, we completed our research - our final results are satisfying.

Final table with training parameters and results

	epochs	batch size	train set	validate set	loss	accuracy	val loss	vall accuracy
1	10	64	47009	11649	0.1515	0.9406	0.1296	0.9562
2	20	64	4700	1116	0.1511	0.9443	0.3984	0.8316
3	15	64	14102	3494	0.1359	0.9479	0.0964	0.9659
4	2	128	14102	3494	0.3016	0.8849	0.7552	0.5072
5	2	32	14102	3494	0.2690	0.8937	0.1947	0.9376
6	4	32	16453	4077	0.1941	0.9250	0.3041	0.8739
7	15	32	15042	3727	0.1502	0.9435	0.0986	0.9678
8	25	64	15042	3727	0.1188	0.9558	0.1080	0.9603

5. Summary

While working on this project we improved our Python knowledge, coding skills and gained hands-on-experience in machine learning. By reading a couple of articles and tutorials about AI/ML we got into the topic very well which paid off during implementation. It turned out that our model is a little bit better at recognizing female faces than male ones. We learned that Google Colab gives more flexibility while working on such project than Kaggle (ex. personal camera usage, considerable limitations in memory allocation). Another fact we found out, important to notice - it is better to have more training images and less epochs than less training images and more epochs when it comes to training the model. Artificial Intelligence is quite complicated issue but gives invaluable opportunities to make life easier. For beginners, the appropriate model training parameters setup is very time-consuming. Next time, implementing such project we should use the EarlyStopping parameter - training with very high number of epochs could end earlier when it is based on progress in a certain place. Thanks to technology and AI-powered devices people can save their time in everyday activities. In the future we are going to improve our model, and probably we will use it in some other application for validating gender.

6. References:

- https://www.kaggle.com/datasets/cashutosh/gender-classification-dataset,
- https://www.youtube.com/watch?v=24XH6q1xxHU,
- https://stackabuse.com/image-recognition-in-python-with-tensorflow-and-keras,
- https://pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers.

7. Project files link

https://github.com/tornal-dev/aiGenderRecognition