Age Prediction: Report

Andrey Maximenko, Alexandr Alexin, Vadzim Valodzin, Ihor Kulynych

1 Introduction

In the realm of machine learning and artificial intelligence, the ability to predict age based on facial features stands as a fascinating yet complex challenge. This report delves into the development of an age prediction model, leveraging advanced techniques to analyze and interpret the subtle nuances of facial characteristics. The significance of such a model extends across various domains, from enhancing user experience in digital platforms to aiding in forensic investigations.

This report unfolds in several key sections. Initially, we explore the dataset creation process, detailing the sources and methods used to gather a comprehensive and representative collection of facial images. Following this, an exploratory data analysis provides insights into the dataset's characteristics and underlying patterns. The core of the report focuses on the model development phase, where we discuss the selection of algorithms, training processes, and fine-tuning techniques employed to achieve optimal performance. Lastly, we present the evaluation results, offering a critical analysis of the model's accuracy and efficiency in real-world scenarios.

2 Creation of the Initial Dataset

Our initial dataset was constructed using two primary sources: the UTKFace and IMDB-WIKI datasets. Combined, these sources provided over 550,000 photographs, which formed the basis for our age prediction model.

2.1 Data Cleaning and Preprocessing

The IMDB-WIKI dataset, known for its noisy data, required extensive cleaning. We utilized the provided Matlab file, which describes the dataset, to conduct the following procedures:

1. Removal of photos where the `face_score` was equal to zero. This step was necessary as a zero score indicates the absence of a face in the image.
2. Deletion of images containing multiple faces, to ensure that each photo had a single, clear subject for age prediction.
3. Elimination of images that resulted in a negative age. This was determined by subtracting the date when the image was taken from the date of birth of the subject.

Further manual cleaning was applied, particularly to the youngest (up to 16 years old) and oldest (from 80 years old) age groups. This process involved removing images where the subjects clearly did not match the age category they were assigned to.

2.2 Face Cropping and Dataset Integration

Using our face detection model, we cropped faces from both the UTKFace and IMDB-WIKI datasets. This step was crucial to focus on the facial features, which are essential for accurate age prediction.

2.3 Merging of Datasets

After preprocessing and cropping, the two datasets were merged. This integration was pivotal in creating a comprehensive and diverse dataset, which is critical for the development of an effective age prediction model. The merging process resulted in an initial dataset with a varied age distribution.

2.4 Age Distribution of the Initial Set

The age distribution of our initial set is depicted in the figure below. This distribution is crucial for understanding the demographic coverage of our dataset.

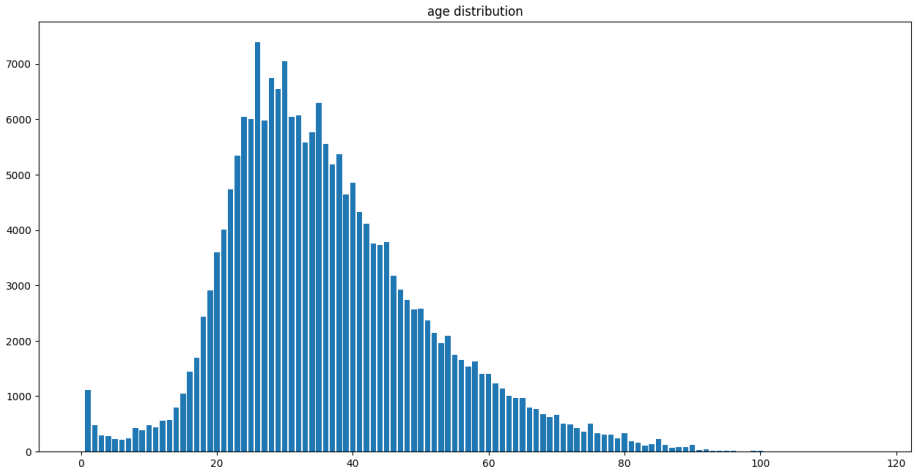

Figure 1: Age distribution of the initial dataset.

3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an essential process to uncover the intrinsic patterns and characteristics within a dataset, especially in image-based applications. It allows us to understand the data better and formulate hypotheses for more complex analyses.

3.1 Analysis of Color Transition

A fundamental aspect of our EDA was the analysis of color transition in images. Color transition is a measure of how much the color changes between adjacent pixels. It provides insights into the texture and detail within the images, which are critical attributes for age prediction models that rely heavily on facial features.

3.2 Technique Description

To analyze the color transitions within images, we employed a gradient calculation technique. Gradients in the context of images represent the directional change in the intensity or color in the image. The Sobel operator, a popular edge-detection algorithm, was utilized to calculate the gradient for each pixel along the x and y axes, effectively measuring the horizontal and vertical changes in color.

The magnitude of these gradients was then computed, which gives us a single value representing the strength of the color change at each pixel. By averaging these values across the entire image, we obtain a measure of the average color transition, which is a quantitative expression of the overall edge content and texture sharpness within the image.

3.3 Results and Interpretation

The results of our color transition analysis are visually represented in the figures. The first figure is a graph showing the average color transition across the dataset, indicating the consistency and variability of textural information in the images. The second figure is a histogram that plots the frequency distribution of the average color transition values. This visualization helps us understand the commonality of different texture strengths within the dataset.

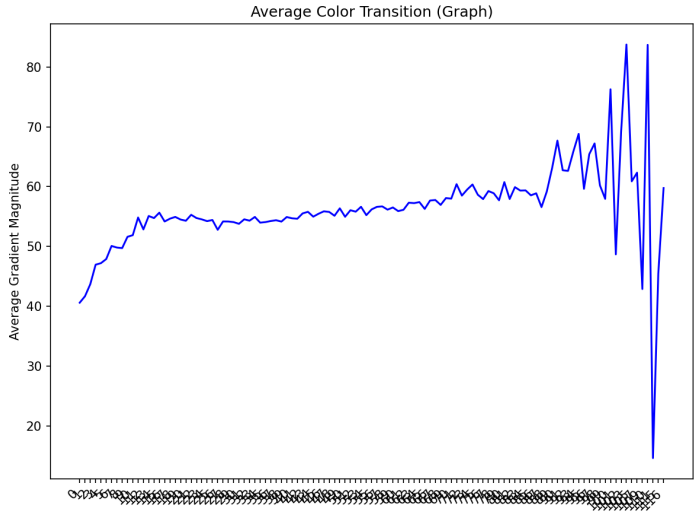

Figure 2: Graph representing the average color transition in images.

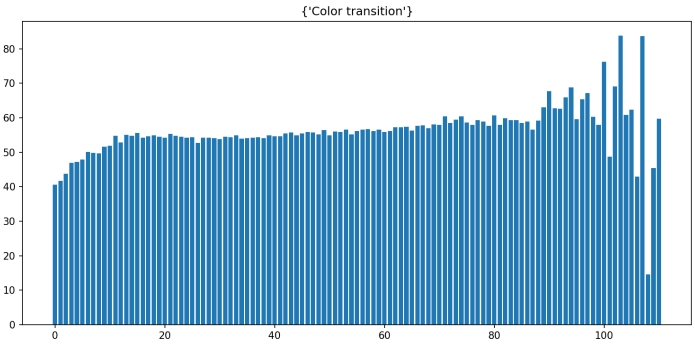

Figure 3: Histogram of the average color transition values.

From these visualizations, we can infer that the majority of the images present a moderate level of color transition, suggesting a balanced presence of edges and smooth areas, which is typical for images of faces. The outliers in the high and low ends could represent images with particularly high detail (like wrinkles) or very smooth areas (like young skin), respectively. These findings lay a foundation for further analysis and feature engineering in the age prediction model development.

3.4 Analysis of Wrinkle Frequency

Wrinkle frequency within the facial images is a potentially powerful indicator of age, as the presence and prominence of wrinkles tend to increase with age. To quantify wrinkle characteristics, we utilized an edge detection technique followed by contour analysis.

3.4.1 Technique Description

The method begins with the application of a Gaussian blur to the image to reduce noise, ensuring that subsequent edge detection focuses on significant transitions, such as wrinkles, rather than noise. The Canny edge detector is then applied to identify edges within the blurred image. Contours are extracted from these edges, which can be thought of as connected curves that outline the wrinkles.

The total length of these contours is calculated, providing a measure of the total wrinkle frequency within the image. A larger total contour length is generally associated with a higher frequency of wrinkles.

3.4.2 Results and Interpretation

The histogram and graph show the distribution of total wrinkle lengths across our dataset.

As anticipated, the wrinkle frequency increases with the age classes, indicating that our method is sensitive to the age-related features it is intended to measure. This trend is particularly pronounced in the older age categories, aligning with the expected increase in wrinkles with age.

3.5 Analysis of Texture Features

In addition to wrinkles, the overall texture of the skin varies with age, affecting the smoothness and granularity of facial images. To capture these textural features, we analyzed the images using Gabor filters, which are particularly effective at capturing the spatial organization of structures within an image.

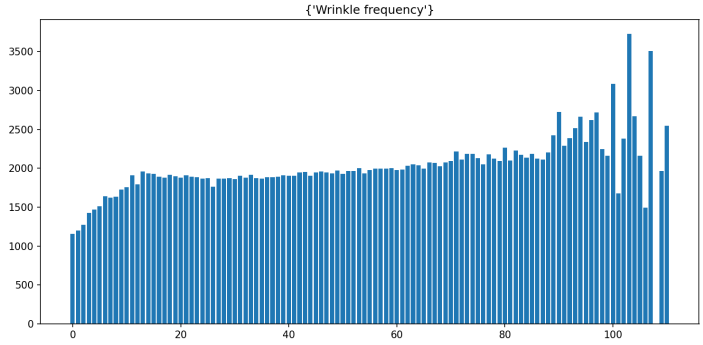

Figure 4: Histogram of wrinkle frequency across images.

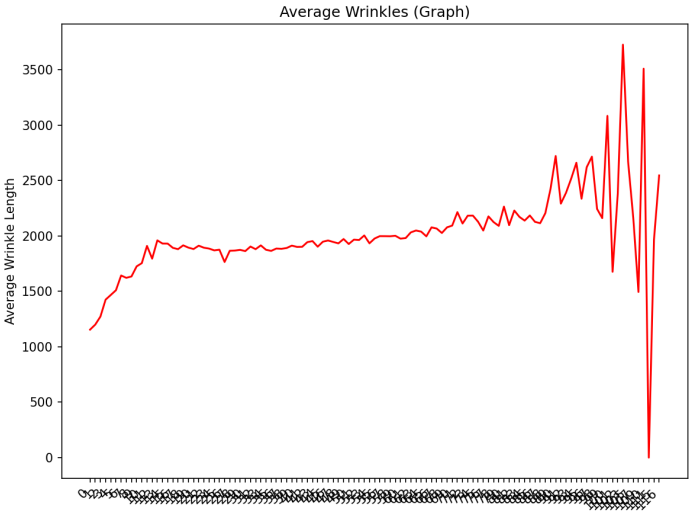

Figure 5: Graph representing the average wrinkle length in images.

3.5.1 Technique Description

Gabor filters are used to analyze the spatial frequency of textures within an image by responding to edges and texture changes. Two separate Gabor filter responses were calculated: one for high frequencies, which captures finer textural details, and another for low frequencies, capturing broader textural patterns.

The average response of these filters provides a measure of how 'textured' or 'smooth' the image is, with higher values indicating more texture and lower values indicating smoothness.

3.5.2 Results and Interpretation

The graphs represent the average texture measured at high and low frequencies across different age groups in our dataset. The histograms below further illustrate the distribution of high and low-frequency texture features across the dataset.

The results from the texture feature analysis suggest that the textural complexity of facial images increases with age. This observation is consistent with the expectation that older skin exhibits more varied textural patterns due to factors such as wrinkles and skin folds.

3.6 Analysis of Gray Index

The gray index is a measure of the intensity of the pixels in grayscale images, which can reflect the overall brightness and potential presence of features like shadows and highlights in facial images. This measure is important as variations in illumination and contrast can influence the perceived age in images. We computed both the average and maximum gray index for images in our dataset.

3.6.1 Technique Description

For this analysis, images were first converted to grayscale, then a Gaussian blur was applied to smooth out the noise, which helps in achieving more accurate intensity measurements. Next, we calculated the average gray index across all pixels in the image to determine the general brightness level. This average serves as a representation of the overall luminance of the image.

Furthermore, the distribution of pixel intensities was examined to find the maximum gray index, which indicates the brightest point in the image. This value can be particularly telling of the presence of highlights that are often associated with more pronounced facial features.

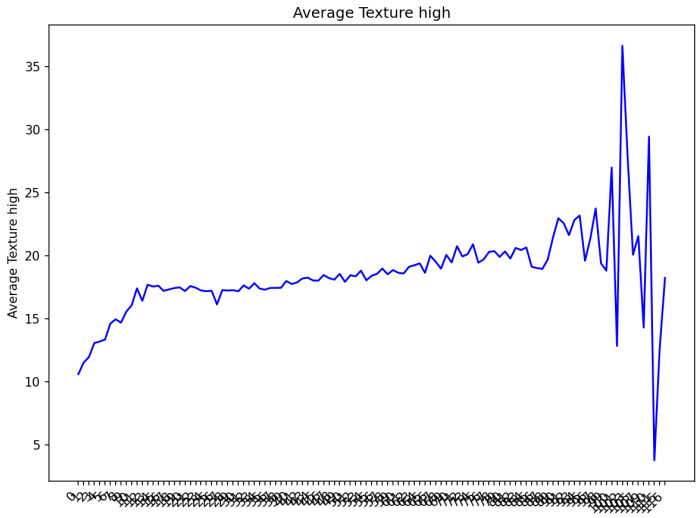

Figure 6: Graph representing the high-frequency texture features in images.

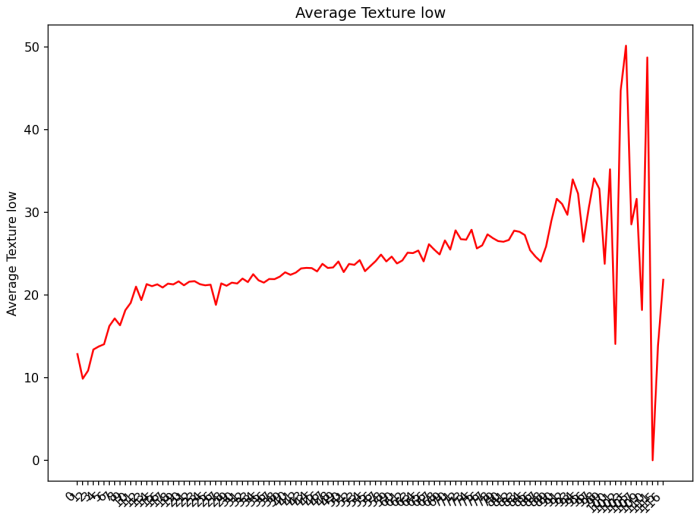

Figure 7: Graph representing the low-frequency texture features in images.

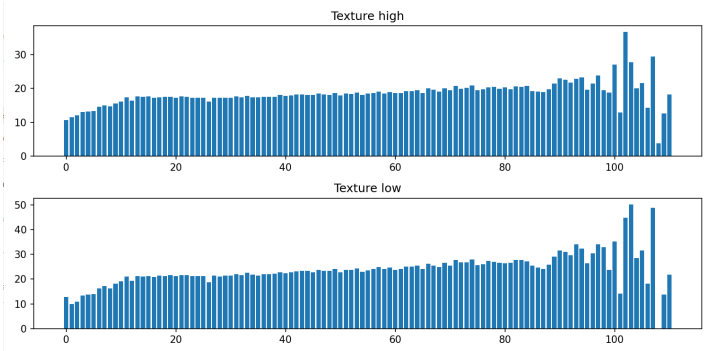

Figure 8: Histograms of high and low-frequency texture features.

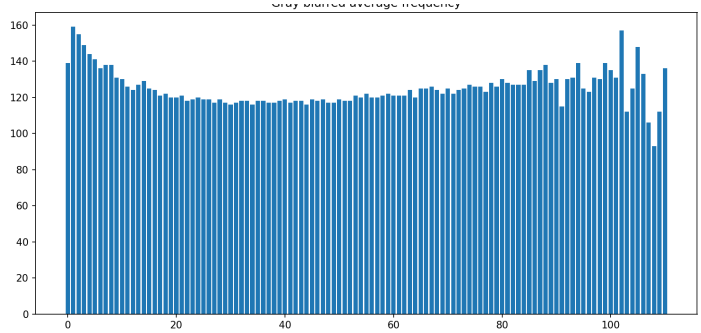

Figure 9: Graph representing the average gray index in images.

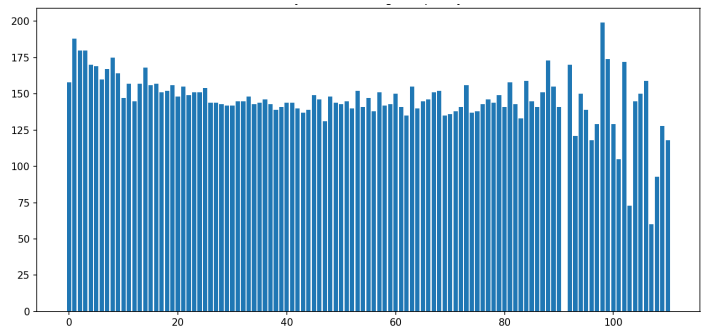

Figure 10: Graph representing the maximum gray index in images.

3.6.2 Results and Interpretation

The graphs represent the average and maximum gray index across different age groups in our dataset.

The graphs show variability in the average gray index, which could correlate with varying image acquisition conditions and natural differences in skin luminance due to age. The maximum gray index also displays variability and a slight increase in the older age groups, possibly due to the increase in contrast from features like wrinkles and spots.

3.7 Conclusion of Exploratory Data Analysis

Our comprehensive Exploratory Data Analysis has provided valuable insights into the characteristics of our dataset. Through the analysis of color transition, wrinkle frequency, texture features, and gray index, we have identified patterns that correlate with the age of the subjects in the images. These findings affirm the complexity of age-related facial features and underscore the importance of considering a multitude of factors when developing an age prediction model.

Each metric we analyzed contributes to a multi-dimensional understanding of the aging process as depicted in facial images. The gradual increase in texture and wrinkle frequency with age, coupled with the nuanced changes in the gray index, emphasizes the intricate relationship between age and facial characteristics. These insights will guide the subsequent feature selection and modeling phases of our project, with the aim of creating a robust and accurate age prediction system.

4 Dataset Augmentation

In order to address the imbalance in our dataset, where certain age groups were overrepresented, we have implemented a series of dataset augmentation procedures. These measures are designed to normalize the number of photos across different age classes, thereby creating a more uniform dataset that can lead to a more generalized and robust age prediction model.

4.1 Normalization of Class Sizes

Initially, the classes with an excessive number of photos were downsized by randomly removing photos until each of these classes contained no more than 4000 images. This step ensures that no single age group dominates the training process.

4.2 Augmentation Techniques

For the underrepresented classes, we employed augmentation techniques to increase their size without compromising the quality and diversity of the images. The specific techniques used were:

1. Merging photos from the same class by combining halves of two different images, as shown in the provided example. This technique generates a new image that retains facial features pertinent to the age class while introducing variations.

Figure 11: Example of a photo created by merging halves from the same class.

Figure 12: Example of a photo created by combining quarters from the same class.

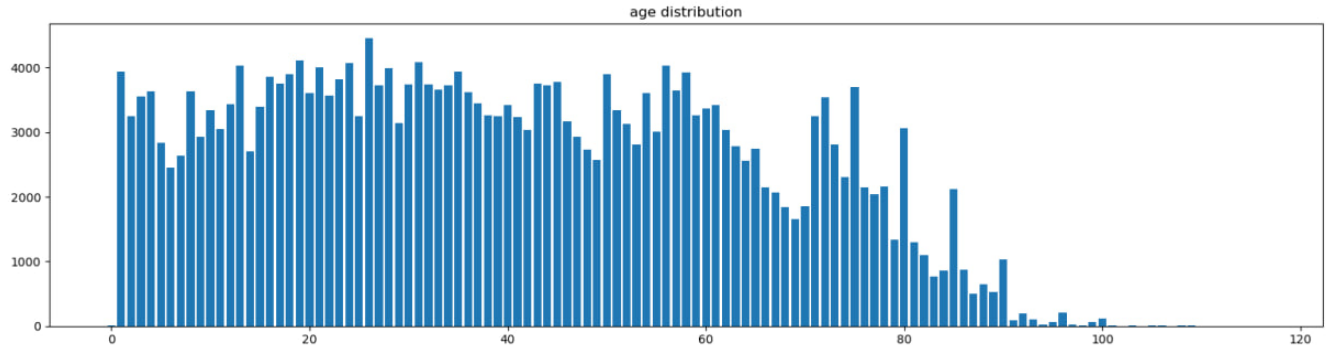

Figure 13: Age distribution after dataset augmentation.

2. In classes with a severe shortage of images, quarters of different photos were combined to further augment the dataset. Similar to the half-and-half method, this approach also went through our face detection model to ensure that only properly cropped faces were included.

These augmentation steps were successful in expanding the classes to a consistent range of 2000-4000 images, with the exception of the oldest age groups, which inherently have fewer representatives.

4.3 Resulting Distribution

The augmentation process resulted in a more balanced distribution of images across age groups, as shown in the graph below. The age distribution after augmentation presents a more uniform dataset, which is crucial for training an unbiased age prediction model.

4.4 Creation of an additional Augmented Set

An additional augmented dataset was created from the processed images by applying rotations of 45 and 90 degrees in both directions and creating mirrored versions of each image. This set aims to simulate various orientations and perspectives, further increasing the robustness of our model against different facial orientations and symmetries.

With these augmentations, we have significantly enhanced the diversity and balance of our dataset, laying a strong foundation for the subsequent model training and testing phases.

5 Evaluation of the face recognition algorithm

This part of the report delves into the performance of an our human face detection model, assessed using the WIDER FACE benchmark and subset of a ImageNet dataset.

5.1 WIDER FACE Benchmark, Analysis of the graphs

In the analysis of the performance graphs, it's evident that our model exhibits a distinct characteristic of high precision coupled with low recall. This pattern is particularly noticeable when juxtaposed against the performance curves of other models in the field. Our model's graph shows a noteworthy trend: it maintains a high level of precision up to a point where most other models, barring the top-tier ones, begin to experience a decline in their precision rates as they strive for higher recall. This inflection point, where our model's performance graph concludes, marks a crucial divergence in the precision-recall trade-off that is common in face detection models. It indicates that while our model is exceptionally adept at correctly identifying faces when it does make a detection (hence the high precision), it tends to miss a larger number of faces compared to others (resulting in lower recall). This trend suggests a conservative detection strategy employed by our model, prioritizing the accuracy of each detection over the quantity of faces detected. Consequently, while other models extend their recall at the expense of precision, our model maintains a steadfast focus on ensuring high confidence in its detections, albeit at the cost of not detecting some faces that other models might capture.

5.1.1 Easy Scenario

The results for the easy scenario.

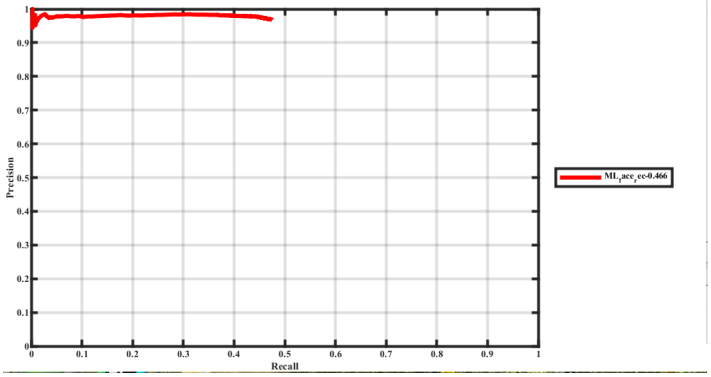

Figure 14: Alone graph result for easy scenario

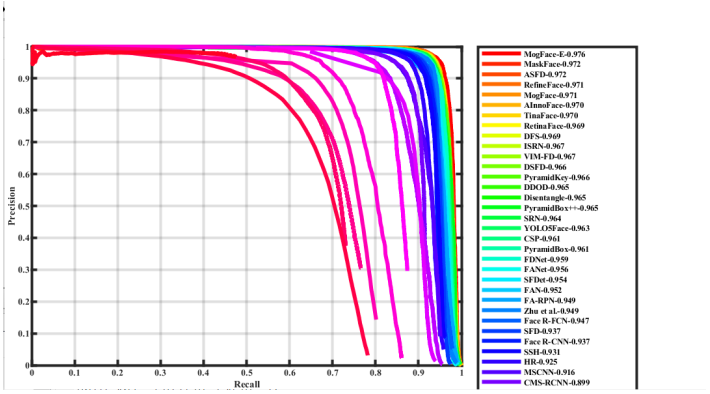

Figure 15: Comparison with some built-in face recognition models

subsubsubsectionMedium Scenario The results for the medium scenario.

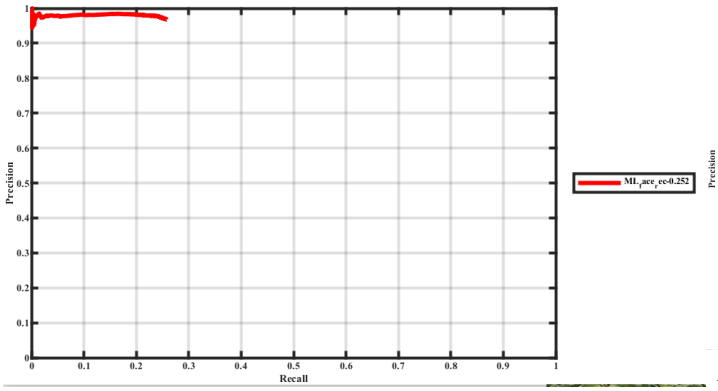

Figure 16: Alone graph result for medium scenario

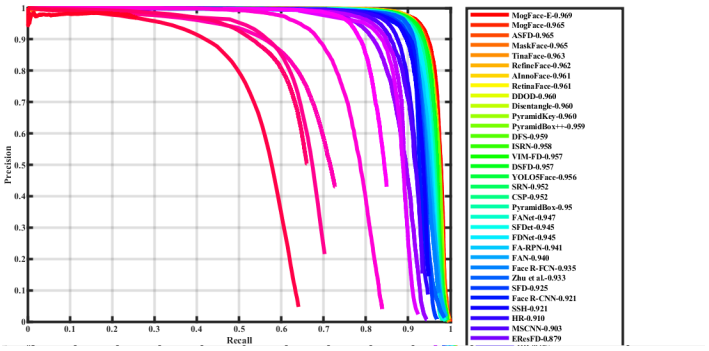

Figure 17: Comparison with some built-in face recognition models

5.1.2 Hard Scenario

The results for the hard scenario.

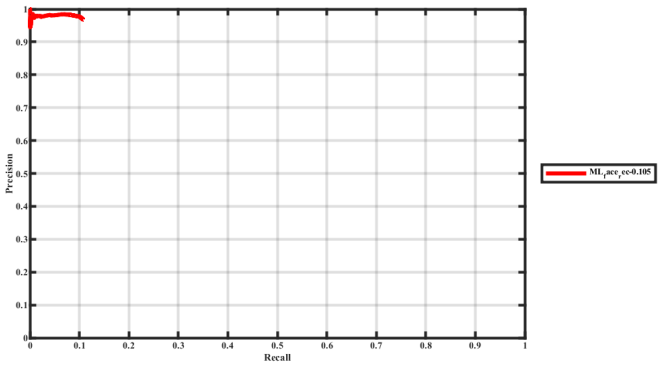

Figure 18: Alone graph result for hard scenario

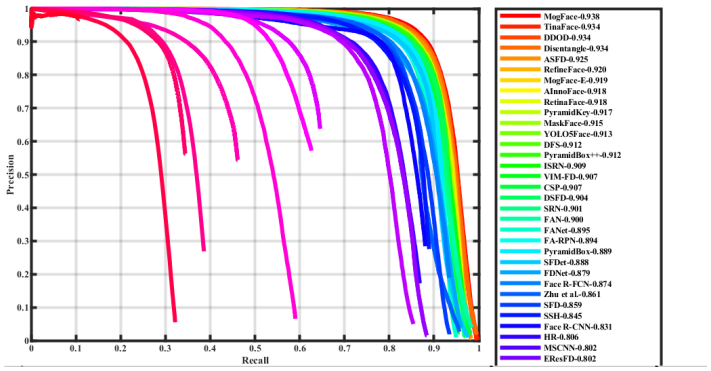

Figure 19: Comparison with some built-in face recognition models

5.2 Analysis of the Benchmark Result

5.2.1 High Precision Explained

The model exhibits high precision, which means that when it identifies a face within an image, there is a high probability that the identified object is indeed a face. This trait indicates a low rate of false positives – instances where non-face objects are mistakenly classified as faces.

5.2.2 Low Recall Explained

Conversely, the model demonstrates low recall, indicating that it misses a significant number of actual faces. In practical terms, this means the model is not detecting every face present in the image, leading to a higher rate of false negatives – actual faces that go undetected.

5.2.3 Implications of Model Performance

The model's tendency towards high precision and low recall suggests a conservative approach in face detection. It opts to minimize incorrect face detections at the expense of potentially missing some faces.

While the high precision is advantageous in scenarios where incorrect face detection is critical, the low recall could be a limitation in situations requiring comprehensive face detection in images.

5.3 Application in Age Prediction

5.3.1 Face Detection as a Foundation for Age Prediction

The role of the face detection model is a preliminary and crucial step in the age prediction process. The quality and reliability of face detection directly influence the subsequent age prediction task.

5.3.2 Precision in Age Prediction Context

For effective age prediction, a clear and accurate detection of faces is paramount. The high precision of the face detection model aligns well with this requirement. It ensures that the inputs for the age prediction model are actual faces, reducing the likelihood of erroneous age predictions based on incorrect face detections.

In the context of age prediction, the quality of data (accurate face detections) often outweighs the quantity. A smaller number of high-confidence face detections is preferable for making precise age predictions.

5.3.3 Managing Low Recall

For applications where capturing every face is crucial, strategies to mitigate the impact of low recall should be considered. This might include model adjustments or integrating additional methods to capture faces that the primary model misses.

5.3.4 Conclusion and Recommendations

The high precision of the face detection model is well-suited for age prediction tasks, where accurate identification of faces is more critical than detecting every single face.

Continuous evaluation and iterative improvements are recommended to enhance the recall of the face detection model without significantly compromising its precision, thus aligning it more closely with the diverse needs of age prediction applications.

Tailoring the model to the specific requirements of the intended application will be crucial. This involves balancing the trade-off between precision and recall based on the tolerance for false positives and the necessity for comprehensive face detection. In summary, the face detection model's high precision is a significant asset for age prediction tasks, ensuring that age analysis is performed on accurately detected faces. The low recall, while a limitation in some contexts, may not be a significant impediment given the nature of age prediction, which requires clear and distinct face images – conditions where the model already excels.

5.4 Testing on ImageNet100 Dataset

To further evaluate the model, it was tested on a subset of the ImageNet dataset known as ImageNet100, which comprises 100 random classes with a total of 130,000 images. The objective was to detect faces in these images. The test results were as follows:

5.4.1 Overview of Results

Out of 130,000 images, only 55 instances of incorrect face detection were noted. This low error rate aligns with our earlier observations from the WIDER FACE benchmark, reinforcing the model's high precision. The incorrect detections were categorized into three groups:

5.4.2 Random Pictures

In 11 cases, random parts of the images were incorrectly marked as faces. These instances are inconclusive.

5.4.3 Statues and Dolls

Three incorrect detections involved statues and dolls. These misidentifications are somewhat understandable, given the human-like features of these objects.

5.4.4 Animals

A total of 41 detections were related to animals, with 15 cases mistakenly identifying clear animal faces as human. Further analysis revealed no significant issues.

5.5 Results of the evaluation

The comprehensive analysis, encompassing both the WIDER FACE benchmark and the ImageNet100 test, underscores the high precision of our face detection model. While the model demonstrates a remarkable accuracy in face detection, as evidenced by the exceedingly low rate of false positives in a diverse set of images, it also exhibits a tendency towards lower recall. This characteristic, although presenting limitations in scenarios requiring exhaustive face detection, aligns well with applications prioritizing accuracy over detection quantity, such as age prediction tasks. The model's performance in


Figure 20: Random incorrect detections (1-3)

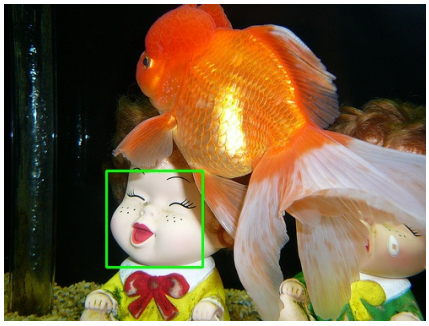

Figure 21: Statue and doll incorrect detections (1-3)

detecting human-like features in statues and dolls, and its distinction between human and animal faces, further validates its precision and underscores the potential for targeted improvements in future iterations.

6 Developing Model

6.1 Introduction

To start with, datasets contain images of different people with different ages, so the type of data is image. According to type of the data, the best model to train is Convolution Neural Network, as CNN which belongs to deep learning model handles images better than other models, especially if image data is preprocessed. In order to construct the best model, the five versions of it are created. With each version, the model is improved and works better with prediction of age of people.

6.2 Construction of CNN model

In order to build the CNN model, we used TensorFlow framework.

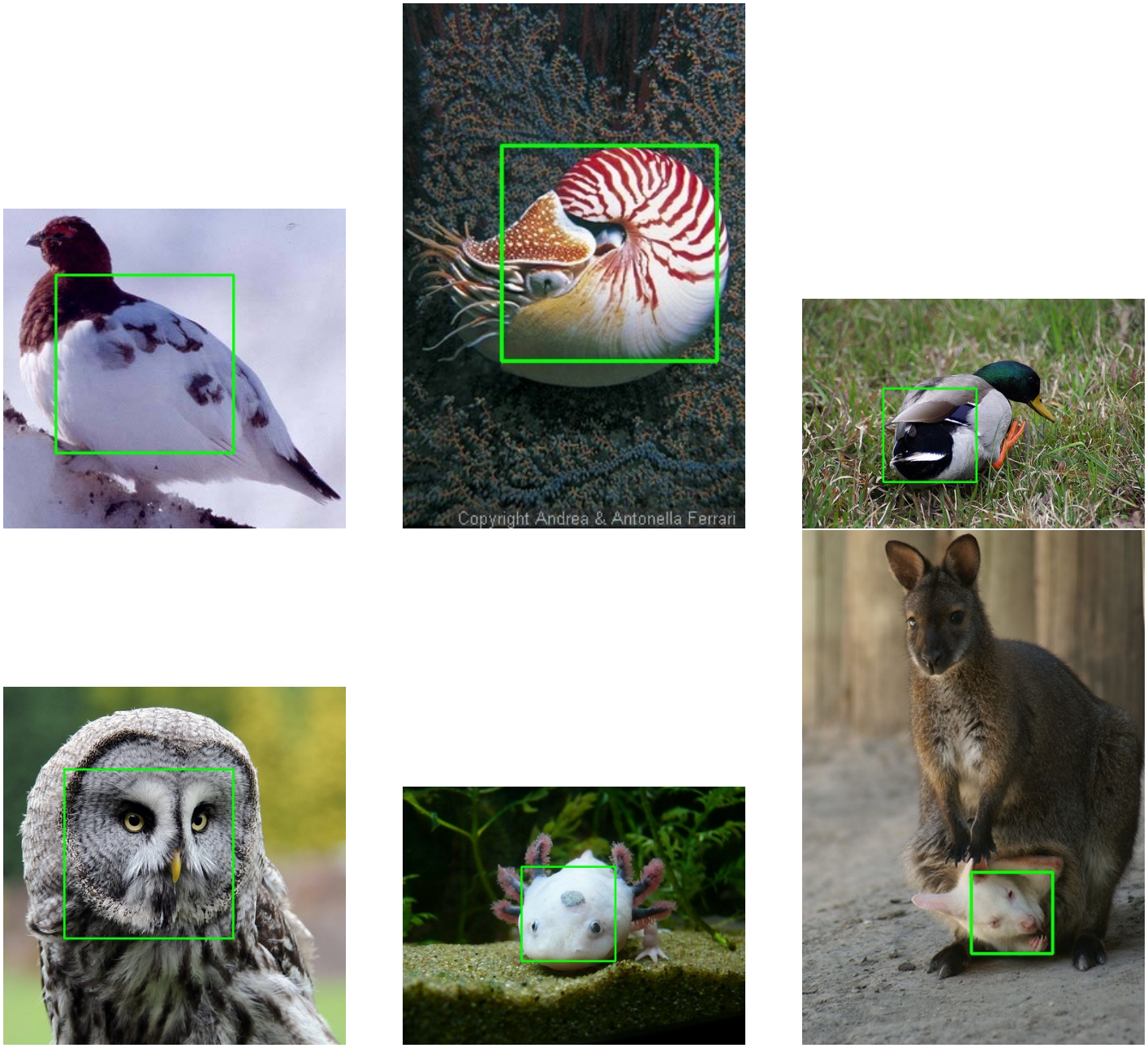

Figure 22: Animals incorrect detections (1-3)

6.2.1 First Version

The starting point of building of the model is simplest its version, the below scheme describe the first version of CNN. The model consists of following layers (Figure 23):

1. One input layer, which specifies the input shape of the image data, which is (128, 128, 1). This indicates that the input images are 128x128 pixels in size, with a single color channel with grayscale.
2. Preprocessing layer which just scales pixel values by 1/255, and four 2D Convolution layers with activation function - Rectified Linear Unit (relu) function.
3. After each layer, there is max pooling operation which reduces the computational cost by decreasing the number of subsequent parameters and computations, which also helps in preventing overfitting and those operations extract the most prominent features like edges while reducing the input size.
4. One flatten layer flattens the 3D output of the previous layer into a 1D array. It is necessary for Dense layers operates on their input.

5. Two dense layers where the last one is predicted age of the person.

```
model = tf.keras.Sequential([
  tf.keras.layers.Rescaling(1./255, input_shape=(128,128, 1)),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(128, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(128, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dense(1, activation='relu')
])
model.compile(
  optimizer='adam',
  loss='mae',
)
```

Figure 23: First version of the CNN Architecture.

The following metrics are used for CNN:

1. The optimization algorithm for model is Adam
2. The loss function is Mean Absolute Error. In case of the model for prediction age, Mean Absolute Error is good choice for prediction tasks, as it is observed how much predicted age is different from true age.

During fitting the model, the batch size is 128 * 8 in all versions as the number of images in datasets are around 250000. For the first version, the number of epochs is 200. The graph on figure 24 describe the loss and validation loss of a neural network model. The X-axis presents the number of epochs and the Y-axis - the value of loss function and blue curvature represents training loss while orange one - validation loss. Description of the next version's graphs are the same.

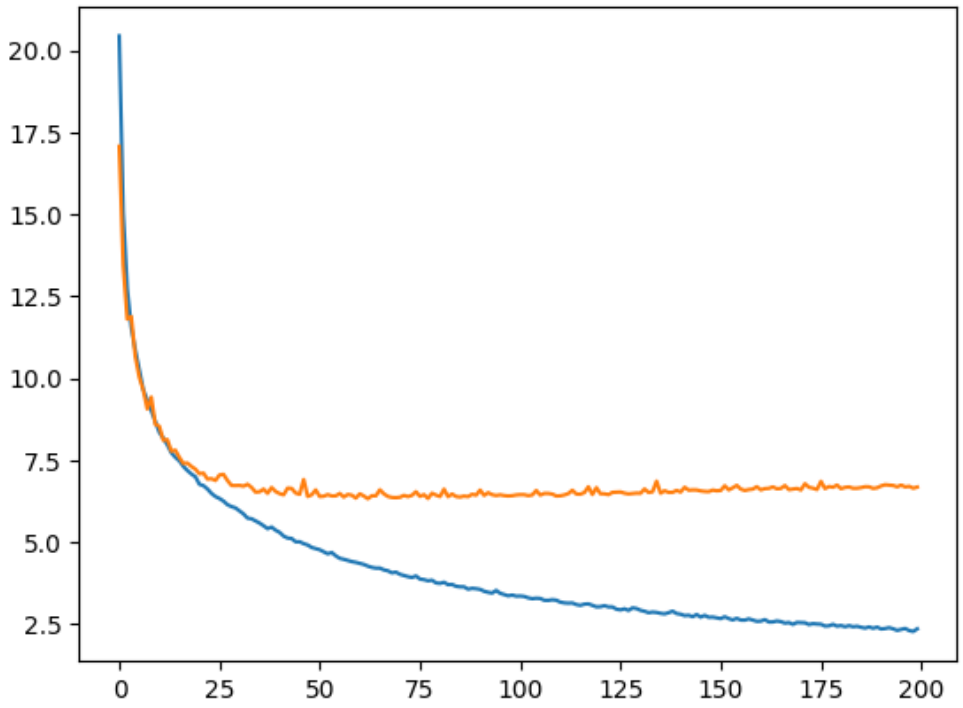

Figure 24: Model v1 losses. Blue line - train set loss, orange line - validation set loss

According to the graph, the validation loss remains constant with tiny fluctuations after 25th epoch, thus it indicates that this version of model is overfitted. Total parameters used is 1439169. MAE for first version is 6.685.

6.2.2 Second version

Previously, the main problem was overfitting, so in order to solve this problem, after each dense layer, one dropout layer is added with dropout rate 0.2. Also, one more dense layer is added. Dropout layers solve the problem with overfitting, generally. The figure 25 shows updated schema.

```
model = tf.keras.Sequential([
  tf.keras.layers.Rescaling(1./255, input_shape=(128,128, 1)),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(128, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(128, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(128, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Flatten(),
  #tf.keras.layers.Dropout(0.25),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(1, activation='relu')
])
model.compile(
  optimizer='adam',
  loss='mae',
  steps_per_execution=64 # in tpu guide was 32
)
```

Figure 25: Second version of the CNN Architecture.

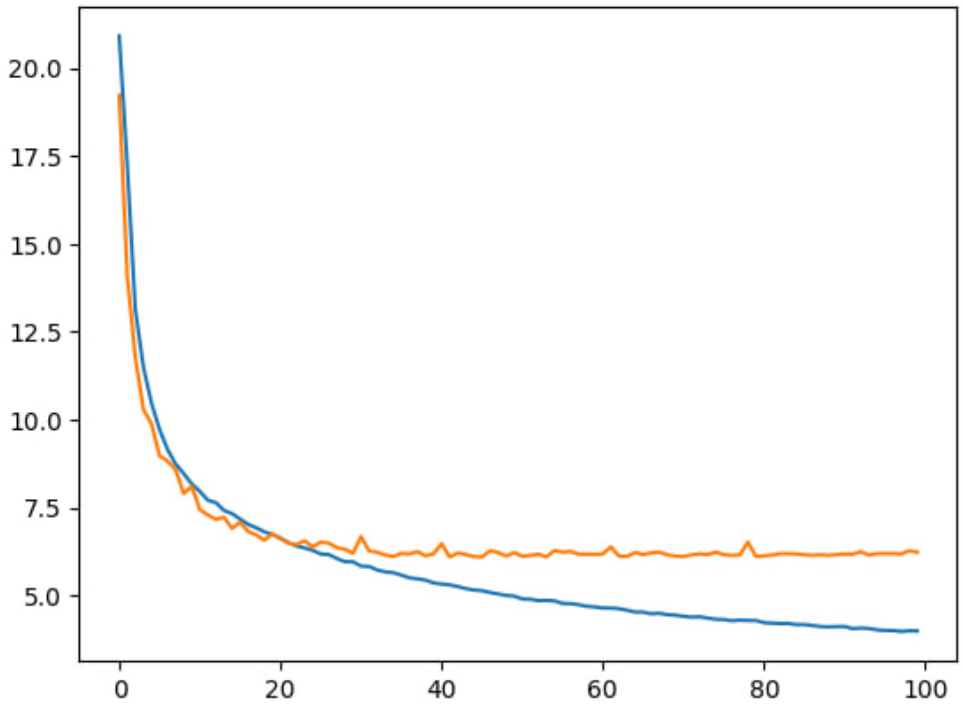

Figure 26: Model v2 losses. Blue line - train set loss, orange line - validation set loss

According to Figure 26, the validation loss converges till 30th epoch in this version, however after 30th epoch the situation are the same as in the first version. In this version, the number of epochs is 100. Batch size does not change. Total parameters used is 570945. MAE for second version is 6.089. This MAE indicates that our model is improved.

6.2.3 Third version

In the third version (Figure 27), after Flatten layer, dropout layer is added with dropout rate 0.2. Looking at graph (Figure 28), it can be observed that validation loss become more convergent than in second version, but overfitting occur after 40th epoch if number of epochs is 100. Total parameters used is 1697665. MAE for third version is 5.507. The value of MAE is decreased quite significantly.

```
with tpu_strategy.scope():
  model = tf.keras.Sequential([
    tf.keras.layers.Rescaling(1./255, input_shape=(128,128, 1)),
    tf.keras.layers.Conv2D(64, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(128, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(256, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(256, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(256, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1, activation='relu')
  ])
  model.compile(
    optimizer='adam',
    loss='mae',
    steps_per_execution=64 # in tpu guide was 32
  )
```

Figure 27: Third version of the CNN Architecture.

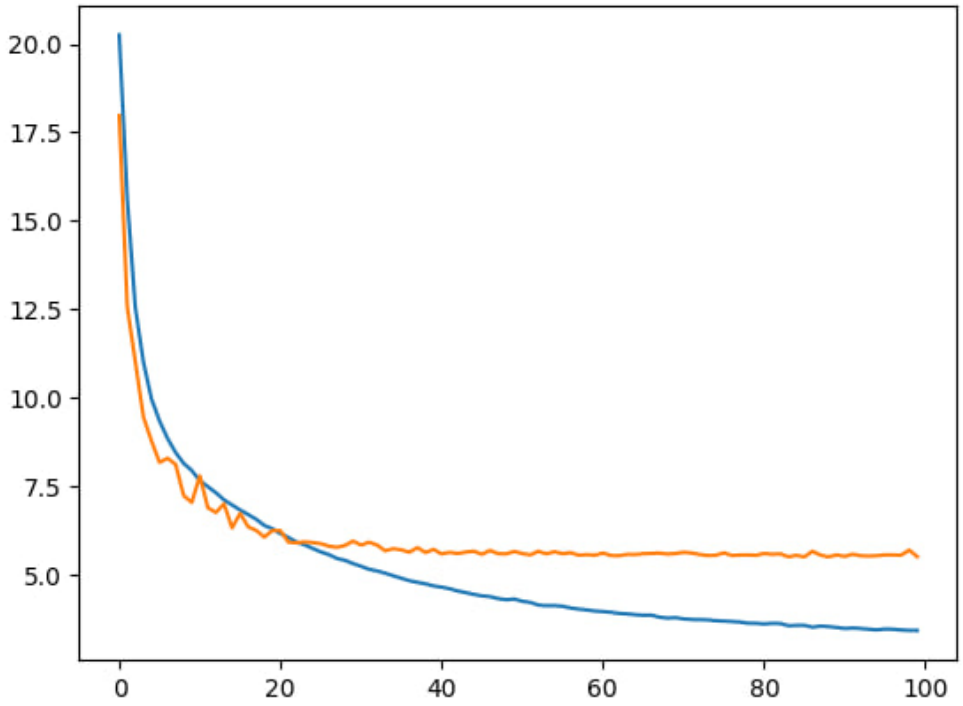

Figure 28: Model v3 losses. Blue line - train set loss, orange line - validation set loss

6.2.4 Forth version

In the CNN's architecture (Figure 29), one dense layer and one dropout layer with the same dropout rate as in previous versions are added. On the figure 30, converge rate of validation loss after 40th epoch is not large. Comparing to previous version of model, a little converge occurs, also gap between validation loss and training loss become smaller, indicating about improvement of model. Number of epochs is 100. Total parameters used is 1333569. MAE for third version is 5.345.

```
with tpu_strategy.scope():
  model = tf.keras.Sequential([
    tf.keras.layers.Rescaling(1./255, input_shape=(128,128, 1)),
    tf.keras.layers.Conv2D(64, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(64, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(256, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(256, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Conv2D(128, 3, activation='relu'),
    tf.keras.layers.MaxPooling2D(),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1, activation='relu')
  ])
  model.compile(
    optimizer='adam',
    loss='mae',
    steps_per_execution=64 # in tpu guide was 32
  )
```

Figure 29: Fourth version of the CNN Architecture.

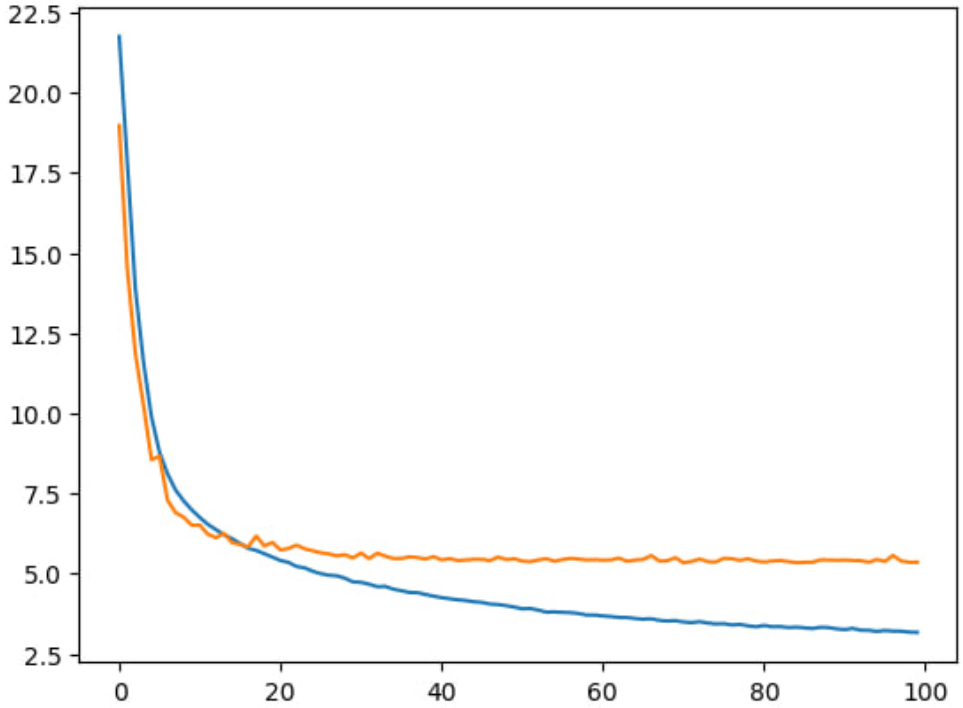

Figure 30: Model v4 losses. Blue line - train set loss, orange line - validation set loss

6.2.5 Fifth version

This version is the last for our CNN model, the final schema of model is depicted by figure 31 , in general the dropout rate was changed on all dropout layers to 0.5. The number of epoch in the final version is 200. However, observing graph (Figure 32), there are convergence of validation loss till 100th epoch, but starting from this epoch the validation loss increasing, so there is still overfitting in the model. In comparison with first version overfitting was partially solved. Total parameters used is 1759681. MAE for fifth version is 5.296. According to value of MAE, it has been decreased significantly in comparison to first version.

```
model = tf.keras.Sequential([
  tf.keras.layers.Rescaling(1./255, input_shape=(128,128, 1)),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(64, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(256, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(256, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Conv2D(256, 3, activation='relu'),
  tf.keras.layers.MaxPooling2D(),
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dropout(0.5),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dropout(0.5),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dropout(0.5),
  tf.keras.layers.Dense(256, activation='relu'),
  tf.keras.layers.Dropout(0.5),
  tf.keras.layers.Dense(1, activation='relu')
])
model.compile(
  optimizer='adam',
  loss='mae',
  steps_per_execution=64 # in tpu guide was 32
)
```

Figure 31: Fifth version of the CNN Architecture.

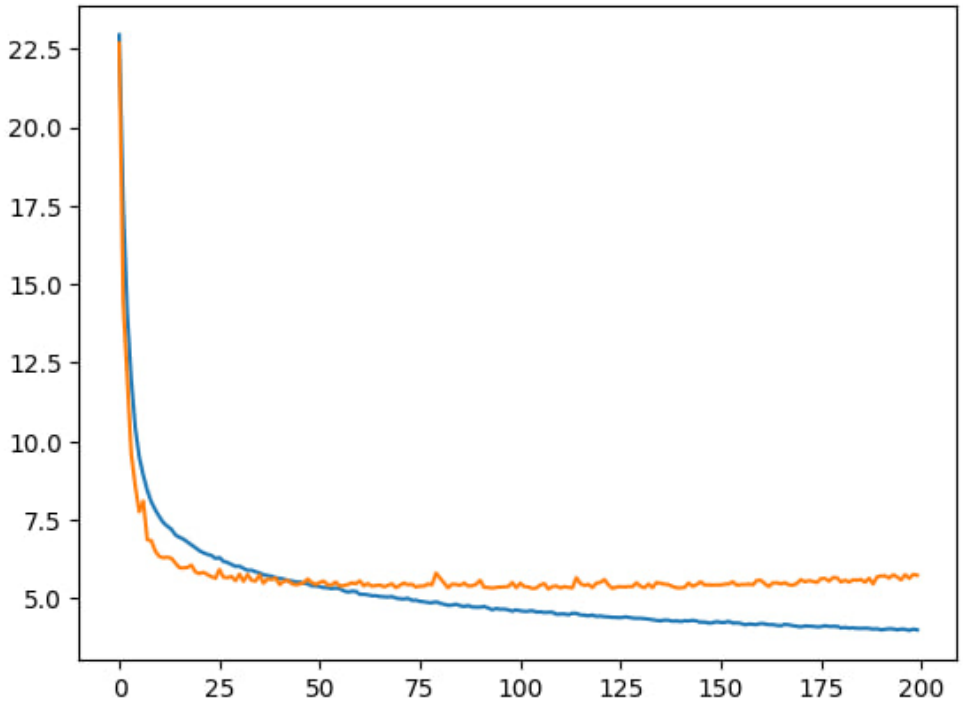

Figure 32: Model v5 losses. Blue line - train set loss, orange line - validation set loss

6.3 Model results

6.3.1 Confusion matrix

The heatmap of confusion matrix is on Figure 33 for our final model. According to confusion matrix, model predicts ages with mean absolute error described in last version of it.

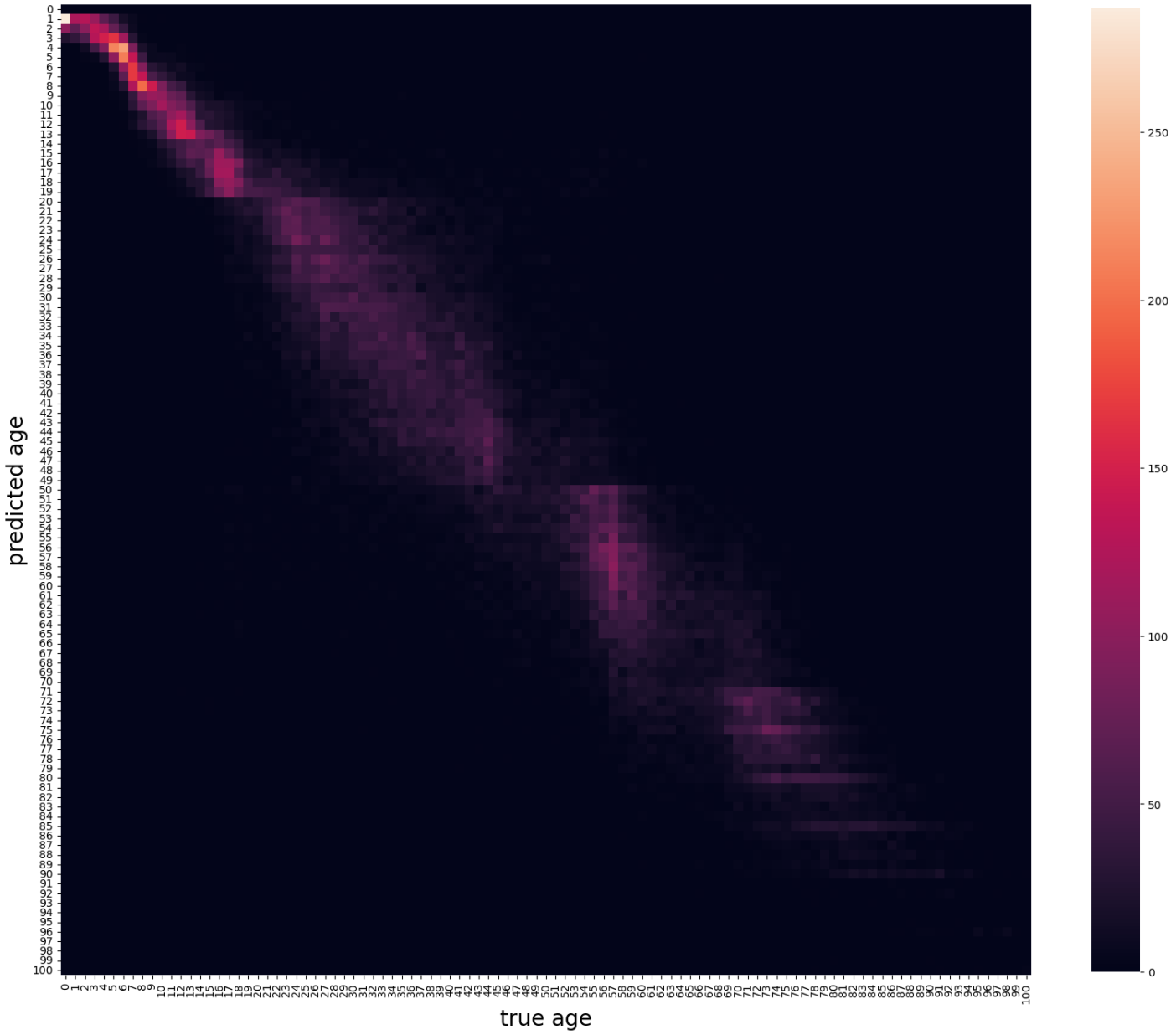

Figure 33: Heatmap of the confusion matrix

6.3.2 Tests on random images from validation set

On the following figures some randomly selected images from validation set and predicted ages for them are depicted

Figure 34

Figure 35

Figure 36

Figure 37

Figure 38

7 Conclusion

The age prediction model demonstrates a promising ability to estimate age from facial features with a degree of accuracy that varies across different age ranges. The model performed exceptionally well in certain cases, matching the predicted age to the actual age, which showcases the potential of the methodology used. However, there were instances where the model's predictions deviated from the true age, indicating areas for improvement. Future work could focus on enhancing the training dataset, refining the model to address these discrepancies, and expanding its applicability across a more diverse set of facial characteristics. This endeavor has provided valuable insights into the complexities of age prediction and sets a foundation for further research in the field of machine learning and facial analysis.