3D reconstruction of the head from photography for 3D-audio modeling: Data augmentation pipeline for ear landmarks detection

Author: Gabriel Giurgică

Supervisors: Roxana-Daniela Florescu

José Javier López Monfort

Master's Thesis presented at the Higher Technical School of Telecommunications Engineering of the Polytechnic University of Valencia, to obtain the Master's Degree in Telecommunications Engineering

Valencia, July 2021

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

Abstract

The ears have a unique character for each person. Due to this fact, the interest in determining its shape has increased considerably in recent decades. Some applications where the ear landmarks detection is useful are: Head-Related Transfer Function determination, 3D human head reconstruction and biometric applications.

This work starts from collection A of the "In-the-wild" Ear Database, which contains 605 ear images containing annotations for 55-points. In order to compensate the limited size of the used database, several data augmentation techniques can be used. The standard mechanisms used for augmentation are: operations to rotate, flip, change brightness, contrast, hue, saturation or channels. Thus, this work presents a comparison between 5 data augmentation pipelines based on the previously mentioned techniques They are used to train multiple models with same architecture for ear landmarks detection. The results obtained after the training are then compared to see which data augmentation pipeline provides the best results

In addition to comparing the 5 data augmentation pipelines, this work also proposes a new neural network architecture, called ResNet-42, for detecting the ear landmarks. Moreover, a different loss function, namely the Wing Loss, is used in contrast to the classical ones used so far for this task.

Resumen[1]

Las orejas tienen un carácter único para cada persona. Debido a este hecho, el interés por determinar su forma ha aumentado considerablemente en las últimas décadas. Algunas aplicaciones en las que la detección de puntos de referencia del oído es útil son: Determinación de la función de transferencia relacionada con la cabeza, reconstrucción de la cabeza humana en 3D y aplicaciones biométricas.

Este trabajo parte de la colección A de la base de datos de oído "In-the-wild", que contiene 605 imágenes de oído que contienen anotaciones para 55 puntos. Para compensar el tamaño limitado de la base de datos utilizada, se pueden utilizar varias técnicas de aumento de datos. Los mecanismos estándar utilizados para el aumento son: operaciones para rotar, voltear, cambiar brillo, contraste, tono, saturación o canales. Así, este trabajo presenta una comparación entre 5 pipelines de aumento de datos basados en las técnicas mencionadas anteriormente.Se utilizan para entrenar múltiples modelos con la misma arquitectura para la detección de puntos de referencia del oído. Los resultados obtenidos después de la capacitación se comparan para ver qué canalización de aumento de datos proporciona los mejores resultados.

Además de comparar las 5 canalizaciones de aumento de datos, este trabajo también propone una nueva arquitectura de red neuronal, llamada ResNet-42, para detectar los puntos de referencia del oído. Además, se utiliza una función de pérdida diferente, a saber, Wing Loss, en contraste con las clásicas utilizadas hasta ahora para esta tarea.

1 The translation of the abstract was done using Google Translate. This is the solution I was able to find to satisfy that rule, which requires that the abstract must be written in Spanish and Valencian.

Table of Contents

1. Introduction

In the last decades, the ear has begun to rise increasingly the interest for several types of applications. It holds a lot of information that can be used for different purposes. For example, the shape of the ear has an extraordinary contribution in finding of Head-Related Transfer Function (HRTF) of a person. Small variations of the pinna (outer part of the ear) can produce substantial changes in the HRTF [1]. Taking into consideration that ear structure has a significant variation among the individuals, it has been recognized its utility in biometric applications [2][3]. Also, based on a 2D wild (unconstrained) image, it has been proven that the 3D human head can be reconstructed including the ears [4].

From an anatomical point of view, the ear can be delimited in three parts: the outer, middle, and inner ear [5]. As can be seen in Figure 1.1 where the different parts are highlighted, the most accessible part of the ear is the outer part. It defines the uniqueness of an individual's ear and it is particularly important in all types of applications mentioned above.

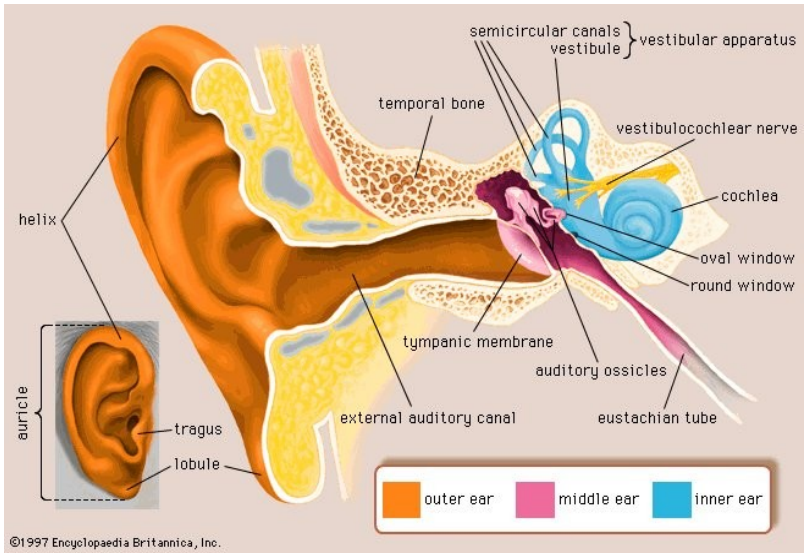


Figure 1.1. Ear structures

In the application of 3D human head reconstruction, detection of the crucial points is a key step to achieve the 3D model. These points, named landmarks, are bordering the distinctive parts of the head. Based on the facial landmarks, there are already available solutions for facial alignment [6][7] and coarse face reconstruction [4][8][9]. Considering that the facial landmarks detection is a common topic and ear landmarks detection is less approached, the current work is focusing on ear landmarks detection.

2. Related Work

To my knowledge, first approaches for the problem of ear landmarks detection in 2D images were proposed by Zhou et al. [10]. They applied many state-of-the-art statistical deformable models to perform ear landmark localization. The authors focused mainly on various Active Appearance Model (AAM) architectures, as they proved to be top performers. They also created a public database (ITWE database) which contains 2663 annotated ear images.

Table 2.1.Network architecture for landmark detection in ear images. It receives as input a gray scale image with 96×96 pixels and outputs a 110-dimensional vector representing 2D coordinates for 55 predefined landmarks.[11]

#	Type	Input	Filter	Stride	Drop	Output
1	Conv/Relu	96×96×1	3×3×1×32	1		96×96×32
2	MaxPool	96×96×32	2×2	2	10%	48×48×32
3	Conv/Relu	48×48×32	2×2×32×64	1		48×48×64
4	MaxPool	48×48×64	2×2	2	20%	24×24×64
5	Conv/Relu	24×24×64	2×2×64×128	1		24×24×128
6	MaxPool	24×24×128	2×2	2	30%	12×12×128
	Flattening	12×12×128				18432
7	Fc/Relu	18432			50%	1000
8	Fc/Relu	1000				1000
9	Fc	1000				110

Other approaches involve the use of Deep Learning (DL) algorithms. Hansley et al. [11] designed a two-stage solution using two instances of a neural network which first combines the convolution and max pooling layers, and then a sequence of fully-connected layers is added. The architecture is shown in Table 2.1. According to the authors *"the first network is used to create an easier landmark detection scenario by reducing scale and translation variations, and the second network is used to generate the 2D coordinates for landmarks".* The DL models were trained using ITWE database. To handle the lack of training data, the authors proposed a data augmentation pipeline which involve rotation, scaling and translation operations to extend the training dataset up to 15500 images.

Besides the solutions mentioned above that directly target 2D ear landmarks detection, Sun et al. [12] has proposed a system by which landmarks can be determined. Their solution predicts the pose and shape parameters of the ear and uses them to reconstruct the 3D ear model. As can be seen in Figure 3.1, the end-to-end architecture of the system was trained using the real images and the generated synthetic images based on reconstructed 3D ear models. The loss function used for training includes landmark loss which is responsible for comparing the landmarks of the original image with the landmarks resulted from synthesized image. The predicted landmarks are manually extracted from the projected vertices of the 3D model.

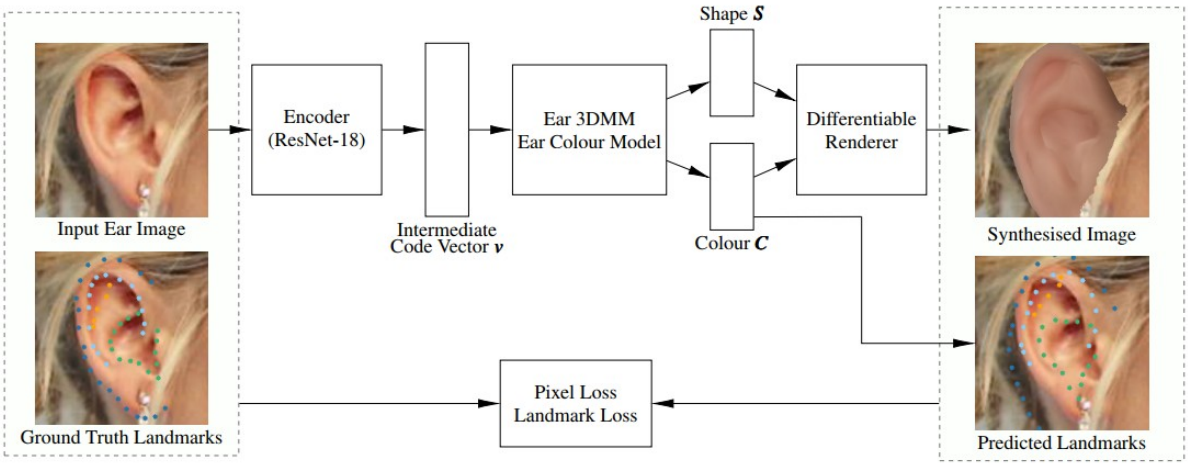


Figure 3.1.Overview of end-to-end architecture of the system proposed by Sun et al. [12]

3. Objectives

The current study uses a new neural network architecture and a new loss function compared to those used so far to detect ear landmarks. This work aims to make a comparison between different methods for enlarging the training dataset using data augmentation techniques.

The proposed new architecture is based on the structure of the ResNet-50 model proposed in [22], the used loss function [24] is designed for robust landmarks detection using CNN, while the data augmentation techniques are inspired by those proposed in [11], [12], [15] and [16].

4. Dataset

To my knowledge, the only dataset that is published for ear landmarks detection problem is collection A from ITWE database [10]. This database consists of 2 sets of data that are addressed for different problems. These datasets are named generic Collection A and Collection B. Collection A is addressed for statistical deformable model construction problems, while the other collection is used for ear verification and recognition in unconstrained environments.

Collection A was collected from Google Images using various ear tags. In this way, 605 ear images "in-the-wild" (unconstrained environment) were collected and then were manually annotated with 55 landmark points. As can be seen in Figure 4.1, the convention for annotating the 55 ear landmarks is: ascending helix (0-3), descending helix (4-7), helix (8-13), ear lobe (14-19), ascending inner helix (20-24), descending inner helix (25-28), inner helix (29-34), tragus (35-38), canal (39), antitragus (40-42), concha (43-46), inferior crus (47-49) and superior crus (50-54). After the annotation process, the images were randomly divided into two sets. The first is

the training set, which contains 500 images, and the second is the testing set, which contains 105 images. For a better understanding of what types of images are in these two datasets, some samples are presented in Figure 4.2 and Figure 4.3.

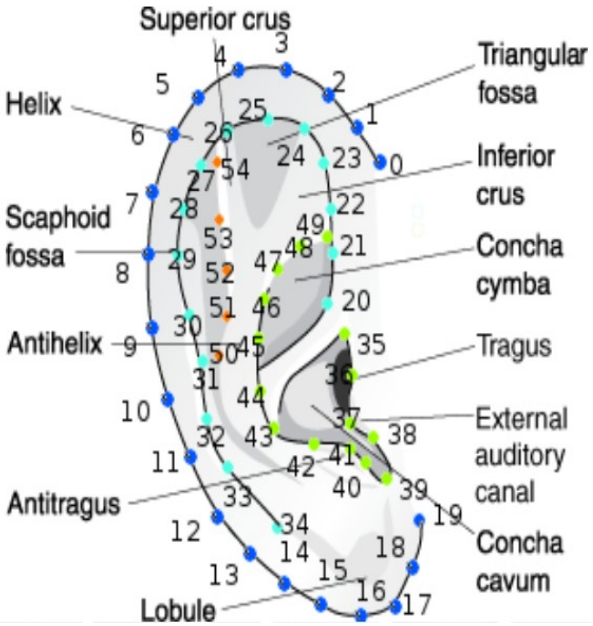


Figure 4.1. Convention for the annotation of ear landmarks: ascending helix (0- 3), descending helix (4-7), helix (8-13), ear lobe (14-19), ascending inner helix (20-24), descending inner helix (25-28), inner helix (29-34), tragus (35-38), canal (39), antitragus (40-42), concha (43-46), inferior crus (47-49) and superior crus (50-54). [10]

Figure 4.2. Two images from the ITWE-A training dataset

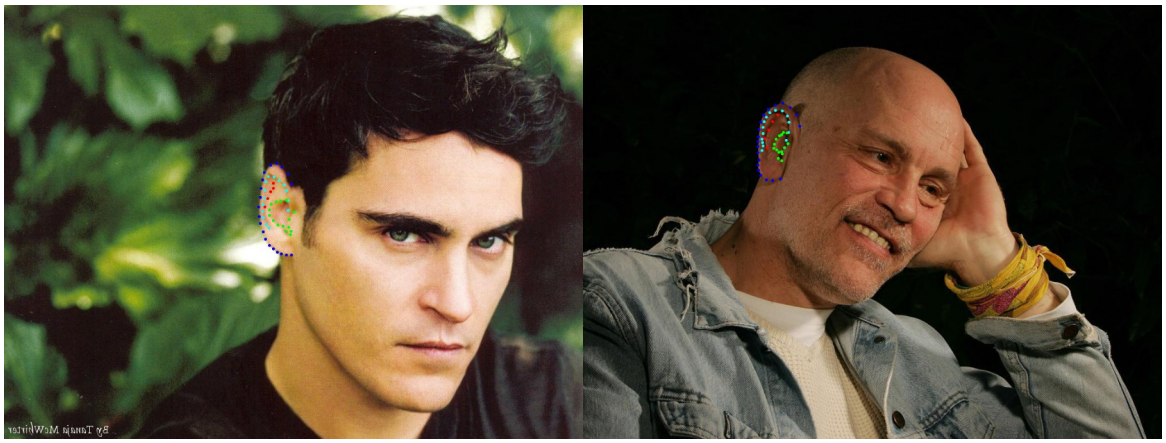


Figure 4.3. Two images from the ITWE-A testing dataset

Collection B was generated using a HoG Support Vector Machine [13] ear detector trained on Collection A. Zhou et al. [10] manually collected 2058 images of 231 people from VGG database [14]. The images were chosen to include visible or partially occluded ears, with variations in angle, lighting, resolution, and aging. Using the ear detector mentioned above, the ear bounding boxes from this collection were generated. A sample from collection B, that also includes the corresponding bounding box is presented in Figure 4.4.

Figure 4.4. Two images from the ITWE-B dataset

Considering the specific particularities of these two collections and the nature of the problem which this project addresses, it is easy to see that collection A from the ITWE database is suitable for further use as a dataset. Since the dataset is divided into training set and test set, it is also necessary to redistribute the images to have a validation set. So, as a result of this redistribution of image, the training set contains 480 images, the validation set only 35 images, while the testing set has 90 images. All three datasets will be used for future experiments.

5. Data Augmentation

The extremly small size of the database is a challenge for training a neural network. To compensate for this shortcoming, image processing techniques are done to expand and diversify the training dataset. As mentioned in section Related Work, Hansley et al. [11] augmented the images using various mechanisms such as: rotating them between -45° and 45°, random scaling them up to 20% of the original ear size in both axes and random translation images up to 20% of the original ear size in each axis. Finally, as a result of these image processing, the size of the training set reached 15500 images. Sun et al. [12] also augmented the images by rotating them between -60° and 60°, thus reaching a dataset having 6000 images in total.

In addition to the rotation technique used in [11] and [12], other image augmentation techniques such as flipping, resizing and photometric distortions are worth considering. These were successfully used both by the authors of SSD: Single Shot MultiBox Detector [15] and by Pierluigi Ferrari in his SSD Keras implementation [16]. The photometric distortions used include randomly changing the brightness, contrast, saturation, or hue of the images and swapping the channels.

In the following, all these image augmentation techniques will be presented. In addition, the cropping process that is used to extract the ear region from images will be detailed. At the end, the order in which these techniques are applied will be shown.

5.1. Rotation

Rotation is a technique that helps easily to increase the number of training images. Rotating an image and then using it to train a neural network helps the network to better generalize the task it performs.

Rotating an image will also change the final size of the image. This can be easily seen in Table 2.1, where each image was rotated randomly with an angle between -70° and 70°.

Table 5.1. Comparison between the original images and their randomly rotated versions with angles between -70° and 70°.

5.2. Flipping

Flipping technique rearranges the pixels while protects the features of the original image. An image can be flipped horizontally or/and vertically. In horizontal flip, the flipping will be on vertical axis, while in vertical flip the flipping will be on horizontal axis.

Table 5.2 presents a comparison between original images and their flipped version. The result of a vertical flip can be seen on the first row of the table, while the horizontal flipping operation is exemplified on third and fourth rows. Finally, on the second row is shown the result after applying both flipping methods.

Table 5.2. Comparison between the original images and their flipped versions.

5.3. Resizing

Since not all images are the same size, it is necessary to apply a resize operation to bring them all to the same size. Resizing can be done by different interpolation methods. The ones chosen for this study are:

- nearest-neighbor interpolation
- bilinear interpolation
- area interpolation
- bicubic interpolation
- Lanczos interpolation

Table 5.3 shows the results of resizing an image from 50x50 to 100x100 using all the interpolations mentioned above.

Table 5.3. Comparison between original images and their resized versions using different interpolation methods: (a) Original image (50x50) ; (b) Resize using the nearest neighbor interpolation (100x100).; (c) Resize using the bilinear interpolation (100x100).; (d) Resize using the area interpolation (100x100).; (e) Resize using the bicubic interpolation (100x100).; (f) Resize using the Lanczos interpolation (100x100). [17]

					
(a)	(b)	(c)	(d)	(e)	(f)

5.4. Photometric Distortions

As it was already mentioned in the beginning of this chapter the photometric distortions that are used for augmentation of the training images are:

- randomly changing the brightness
- randomly changing the contrast
- randomly changing the saturation
- randomly changing the hue
- swapping the channels

5.4.1. Brightness

Changing the brightness leads to a darker or lighter image compared to the original one. This technique allows the neural network to be more robust to variations in illumination levels. To use this technique, the image needs to be in RGB or BGR format.

Table 5.4 shows a comparison between the original images and their variants resulted from randomly changing the brightness.

Table 5.4. Comparison between the original images and their versions resulting from randomly changing the brightness.

5.4.2. Contrast

Contrast is the difference in luminance or color aspects that can make an object more distinguishable or not. Changing the contrast of the images is a good approach to make the model less sensitive to differences in intensity. To use this technique, the image needs to be in RGB or BGR format.

Table 5.5 shows a comparison between the original images and their versions resulted from randomly changing the contrast.

Table 5.5. Comparison between the original images and their versions resulting from randomly changing the contrast.

5.4.3. Saturation

"Saturation can be thought of as the 'amount' of color in an image" [18]. To change the saturation of an image, is necessary to change the color space in which the image is stored in HSV format.

Table 5.6 shows a comparison between the original images and their versions resulted from randomly changing the saturation.

Table 5.6. Comparison between the original images and their versions resulting from randomly changing the saturation.

5.4.4. Hue

"Hue can be thought of as the 'shade' of the colors in an image" [18]. To change the hue of an image, it is necessary to change the color space in which the image is stored in HSV format.

Table 5.7 shows a comparison between the original images and their versions resulted from randomly changing the hue.

Table 5.7. Comparison between the original images and their versions resulting from randomly changing the hue.

5.4.5. Channels swap

The channel swap is a technique that involves exchanging the channels of an image. For a RGB image, the swap (2 1 0) would involve swapping the red and blue channels, keeping the green channel unchanged.

Table 5.8 shows a comparison between the original images and their versions resulted from randomly swapping the channels.

Table 5.8. Comparison between the original images and their versions resulting from randomly swapping the channels.

5.5. Cropping

The current work focuses on detecting ear landmarks. Ear detection is not the subject of this study, so it is necessary to extract the region of the ear from the original image. This step is performed using the ear landmarks provided by the ITWE-A dataset. Initially, the minimum and maximum values on the x and y axis are identified. Then, based on the height and width of the region where the ear is located, the identified values are adjusted to also include an adjacent region of the ear. Once the region is successfully identified, the cropping operation follows, which has the role of preserving only the interest portion of the image.

5.6. Data Augmentation Pipeline

Starting from the data augmentation pipelines proposed in [11], [12] and [16] in this work, 5 other pipelines are proposed to see if other augmentation techniques can help to improve the performance of the neural networks in the problem of ear landmarks detection. Before going and analyzing them we will define the photometric distortion pipeline, which is a sub-pipeline common to all 5 main pipelines.

The main pipelines are generic named:

- Data Augmentation 1
- Data Augmentation 2
- Data Augmentation 3
- Data Augmentation 4
- Data Augmentation 5

5.6.1. Photometric Distortion Pipeline

As can be seen in Figure 5.1, this sub-pipeline has 2 branches. When an image is provided, one of them is randomly selected to augment the image. Each branch has 50% chances of being chosen. Both options share the same types of operations, but the order of performing the operations is different.

The first operation that must be considered regardless of the chosen branch is to convert the image into a 3 channels image. To be able to apply techniques such as random brightness, contrast, hue or saturation, it is necessary to change the type of data, in which the image is stored, in float32. Conversion operations from RGB to HSV and vice versa require that the data type be uint8. Each of the following operations: random brightness, random contrast, random hue, random saturation, and random swapping channels techniques has a 50% chance of being applied to the image provided to them at the input. Thus, the image at the exit of this sub-pipeline has a chance of 3.125% not to suffer any change, excluding the 3-channel image conversion operation.

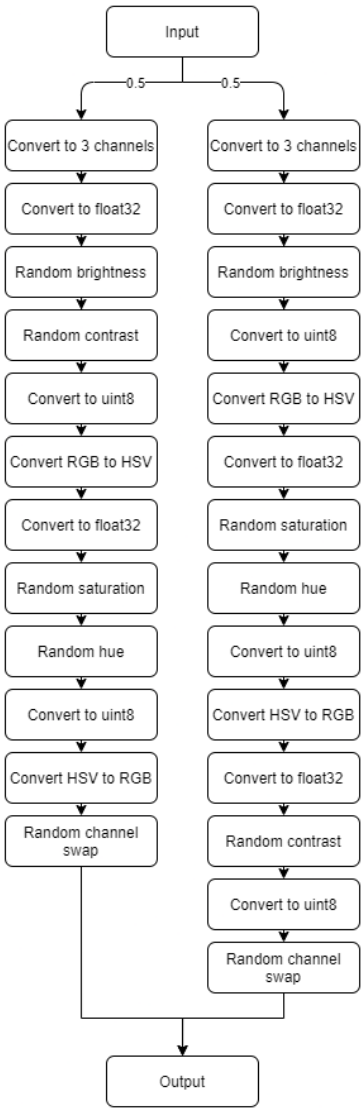


Figure 5.1. Photometric Distortion Pipeline

5.6.2. Data Augmentation 1

First pipeline starts with applying the photometric distortion sub-pipeline described in the previous section. Then, with equal probability, either the random flip horizontal operation or the random flip vertical operation can be chosen. The image provided to the block that performs vertical or horizontal flipping has a 50% chance of being flipped. The next technique applied is random rotation. With 70% chance, the image can be rotated with an angle between -70° and 70°. Finally, the ear region is cropped from the image and it is resized to 224x224 using one of the 5 interpolation methods described in section Resizing. All these steps are illustrated in Figure 5.3.

5.6.3. Data Augmentation 2

The second pipeline applies photometric distortion to the input image. Then with a 70% chance it rotates the image with a random angle between -45° and 45°. Finally the ear region is cropped and resized to 224x224. The pipeline is shown in Figure 5.2.

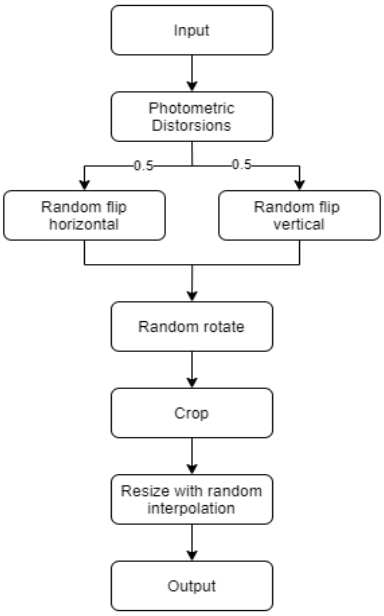


Figure 5.3. Data Augmentation 1

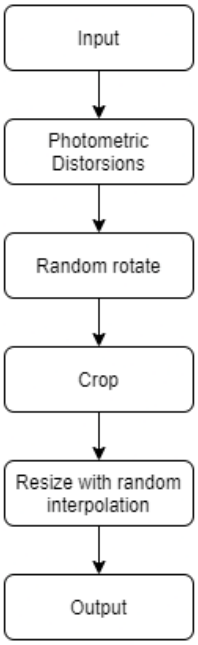


Figure 5.2: Data Augmentation 2

5.6.4. Data Augmentation 3

The third pipeline is like the second one with the same structure shown in Figure 5.2. The only difference is the range of rotation angles that can be applied to an image. So, in this case, the rotation is done at a random angle between -70° and 70°.

5.6.5. Data Augmentation 4

Data Augmentation 4 is the particular case of Data Augmentation 1 in which only random flip horizontal is applied. This can be easily seen by comparing Figure 5.4 with Figure 5.3. The chances of applying random horizontal flipping or random rotation remain the same. Even the range of choice of rotation angle is still unchanged.

5.6.6. Data Augmentation 5

Data Augmentation 5 shares almost the same pipeline as Data Augmentation 4. The only difference is that this pipeline performs random flip vertical while Data Augmentation 4 performs random flip horizontal. This can be easily seen by comparing Figure 5.5 with Figure 5.4.

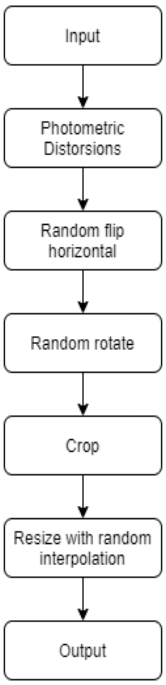


Figure 5.4: Data Augmentation 4

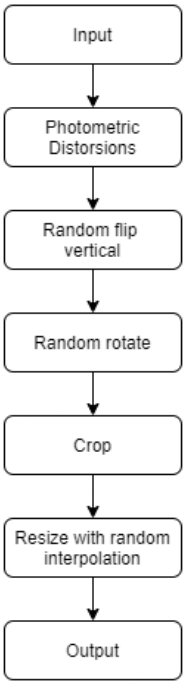
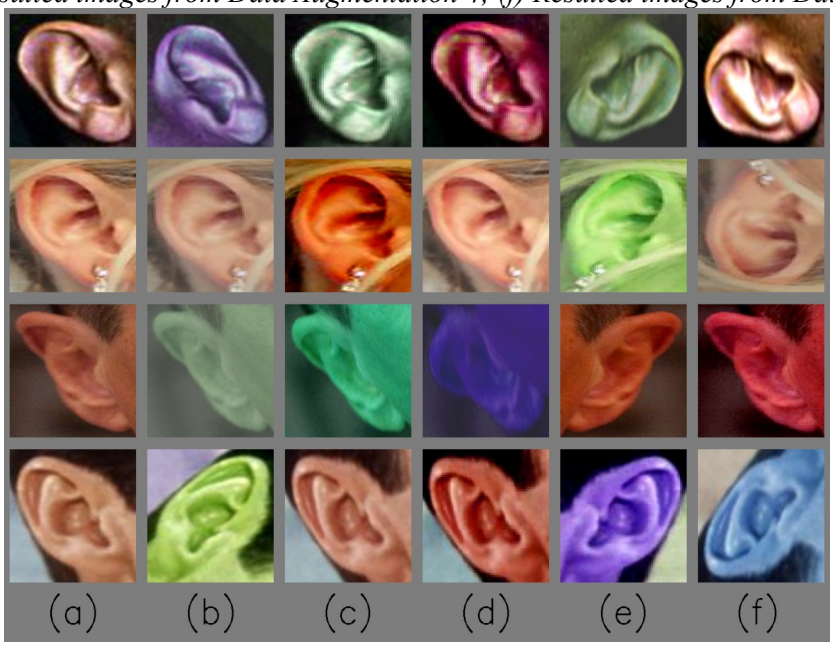


Figure 5.5: Data Augmentation 5

5.6.7. Comparison between pipelines

Now that all 5 main data augmentation pipelines have been presented, the next step is to illustrate what kind of images can be generated with their help. Table 6.1 provides a comparison between the proposed pipelines and the region of the ear cropped from the original image. All images have a size of 224x224. As can be seen from the table at the end of the augmentation process, regardless of the chosen pipeline, it is unlikely that the image will remain unchanged. Depending on the augmentation chain chosen, the output images can be more or less varied.

Table 6.1: Comparison between data augmentation pipelines: (a) Ear region from original images; (b) Resulted images from Data Augmentation 1; (c) Resulted images from Data Augmentation 2; (d) Resulted images from Data Augmentation 3; (e) Resulted images from Data Augmentation 4; (f) Resulted images from Data Augmentation 5;



6. Neural Network

Neural networks have proven useful in detecting ear landmarks. CNN used in [11] and ResNet-18 used in [12] achieved satisfactory results in the tasks they had to perform. Thus, starting from the previous architectures, this work proposes an architecture, hereinafter referred to as ResNet-42, which uses the residual learning framework [19] and is intended for ear landmarks detection problem. Before going into the structural details of the network, the types of blocks used in it will be presented. It should also be noted that the TensorFlow framework is used.

6.1. Identity block

The identity block has a connection which skips over 3 layers. Its structure can be seen in Figure 6.1. The upper path, called *shortcut path*, skips the lower path, called the *main path*. The shortcut path creates the conditions to be easier to learn an identity function. This means that more identity blocks can be used in a network with minor risk of harming the training set performance. Another detail that must be mentioned is that the entrance and the exit have the same dimension.

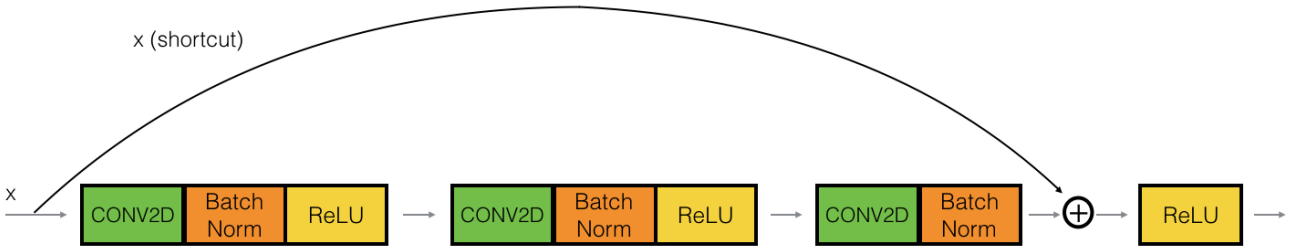


Figure 6.1. Identity block [20]

6.2. Convolutional block

The convolutional block is similar to Identity block, the difference between them is the presence of Error: Reference source not found and Error: Reference source not found on the short path. These layers are used to resize the input to match up the desired output size. The structure of the block is shown in Figure 6.2.

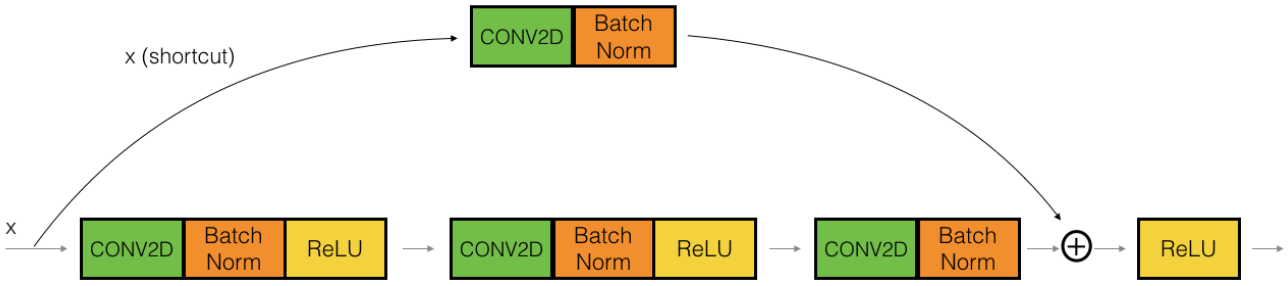


Figure 6.2. Convolutional block [20]

6.3. ResNet-42 architecture

Figure 7.1 shows in detail the architecture of ResNet-42. The abbreviation *ID BLOCK* means Identity block and *ID BLOCK x5* means that there are 5 Identity blocks stacked together.

The present neural network architecture is based on the structure of ResNet-50 model proposed in [20]. Besides the fact that the model proposed in this work is shallower compared to ResNet-50, it uses more often the Error: Reference source not found which helps to increase the performance. The total number of parameters of the model is 12,897,262 of which 12,866,670 are trainable

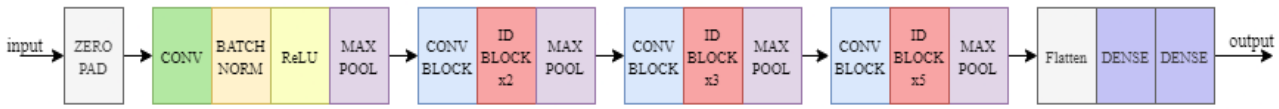


Figure 7.1. ResNet-42 architecture

7. Implementation details

7.1. Training

The first step in being able to train a neural network is to create it. The model, ResNet-42, is created to accept 224x224x3 (height, width, channels) images at its input. After its creation, it is necessary to choose an optimizer and a loss function for the training. The optimizer used is Adam [21], and the loss function is Wing Loss [22]. They will be presented in more details in the sections Adam optimizer and Wing Loss.

The next step is to set the mini-batch size to 32. This is a hyper-parameter that defines the number of samples to be processed before updating the internal trainable parameters of the model. This hyper-parameter is also used in generating augmented input images. The input images are augmented before each iteration, thus allowing the expansion of the training dataset at the time of training. This way of image augmentation allows to constantly create various input images for the model.

Other 2 hyper-parameters worth mentioning are the learning rate and the number of steps per epoch. The learning rate is a hyper-parameter that controls how much the model weights can be updated based on the result of the estimated loss function. This parameter is variable throughout the training epochs. In the first 3 epochs the learning rate has a value of 0.001, in the next 3 it has a value of 0.0001, between 6 and 15 epoch it has 0.00001, and after that its value becomes 0.000001. The number of steps chosen per epoch is 100. This parameter specifies how many updates of the model weights are made before considering that a training epoch has ended.

7.2. Wing Loss

Wing loss [22] is a loss function designed for robust facial landmarks detection using CNN. It has designed to pay more attention to the samples with small or medium range errors, not only for the samples with large range errors. The mathematical relation that describes this function is:

$$wing(x) = \begin{cases} w \ln(1 + |x|/\epsilon) & \text{if } |x| < w \\ |x| - C & \text{otherwise} \end{cases}, \tag{7.1}$$

where x is the difference between the predicted value and the true value, the positive parameter w defines the boundary between the linear part and the nonlinear part, the parameter ϵ limits the curvature of the nonlinear region and C is a constant that eases a smoother transition between linear and nonlinear parts. C can be calculated using the following formula:

$$C = w - w\ln(1 + w/\epsilon) \tag{7.2}$$

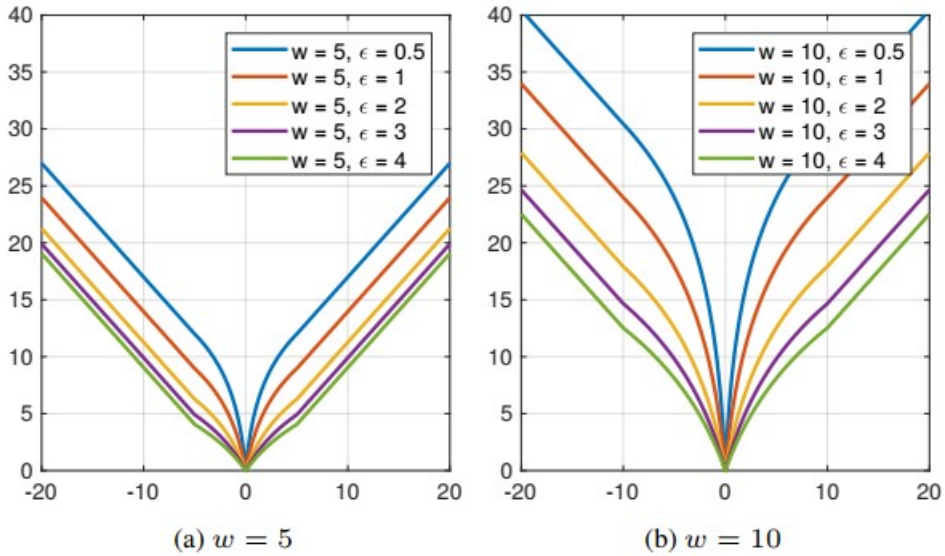


Figure 7.2: Visual interpretation of Wing loss with variation of parameters w and ϵ [22]

The influence of small errors is enhanced using a natural logarithmic function, which introduces a non-linearity when such errors occur. This type of function helps to restore the balance between the influence of errors of varied sizes. However, for situations where there are large errors, the L1 function is used, which helps to quickly reduce them. According to authors, the Wing loss function should behave "for small errors as a log function with an offset, and for larger errors as L1".

7.3. Adam optimizer

Adam is an optimization algorithm that has been designed specifically for training deep neural networks. "The name Adam is derived from adaptive moment estimation" [21]. It is a combination of RMSprop and Stochastic Gradient Descent (SGD) with momentum [23]. The algorithm of this optimizer is as follows:

$$V_{dW} = 0, S_{dW} = 0$$

On iteration t:

$$V_{dW}^{t} = \beta_1 \cdot V_{dW}^{t-1} + (1 - \beta_1) \cdot dW$$

$$S_{dW}^{t} = \beta_2 \cdot S_{dW}^{t-1} + (1-\beta_2) \cdot dW^2$$

$$V_{dW}^{corrected} = V_{dW} / (1-\beta_1)$$

$$S_{dW}^{corrected} = V_{dW} / (1-\beta_2)$$

$$W^{t} = W^{t-1} - \alpha \cdot \frac{V_{dW}^{corrected}}{S_{dW}^{corrected} + \varepsilon} \quad ,$$

where dW represent the calculated gradients, V_{dW} is the first momentum exponentially average used in SGD with momentum, S_{dW} is the second momentum exponentially average used in RMSprop, β_1, β_2 are hyper-parameters that determine the number of significant values that are averaged, $V_{dW}^{corrected}$ is V_{dW} after bias correction, $S_{dW}^{corrected}$ is S_{dW} after bias correction, alpha is learning rate, ε is a constant to prevent possible convergence errors and W represents the weights of the model.

According to the authors, the advantages of using this optimization algorithm are:

- straightforward to implement
- requires little memory
- computationally efficient
- invariant to rescaling of gradients
- appropriate for large datasets and/or high dimensional parameter spaces
- appropriate for a wide range of non-convex optimization problems
- hyper-parameters require little or no tuning
- does not require a stationary objective function

The values of the hyper-parameters used for the experiments are:

- $\beta_1 = 0.9$
- $\beta_2 = 0.999$
- $\varepsilon = 1e\text{-}08$

7.4. Evaluation

For the evaluation, the current work uses the same approach in [10] and [11]. The evaluation is done by computing the cumulative error distribution curves. The error is determined using point-to-point Euclidean distance normalized by the ear's bounding box. The mathematical relation is:

$$error = \frac{1}{N} \sum_{n=1}^{N} \frac{\|x_n - y_n\|_2}{d} \quad , \tag{7.3}$$

where N represent the 55 ear landmarks, x is the ground truth landmarks for a given ear, y represents the corresponding predictions, and d is the diagonal of the ground truth of the bounding box. The ear bounding box is generated using the cropping operation descried in section Cropping and then the cropped region is resized to 224x224 using the methods described in section Resizing. This sequence of operations leads to the same bounding box dimensions for all images. This means that d is a constant equal to $224 \cdot \sqrt{2}$.

8. Results

In this section all the proposed data pipelines and the resulted models are analyzed. For the evaluation of the models, the cumulative error distribution curves are plotted. Also, for each data pipeline the following results are presented: a plot of model loss on training and validation datasets and a table containing 36 images with the results of the predictions of the best model on a subset of the testing dataset.

8.1. Data augmentation 1 - Results

As can be seen in Figure 8.1, the neural network seems to have reached a saturation of the loss value after 6 epochs of training. Continuing training will not lead to further improvement in the value of the loss.

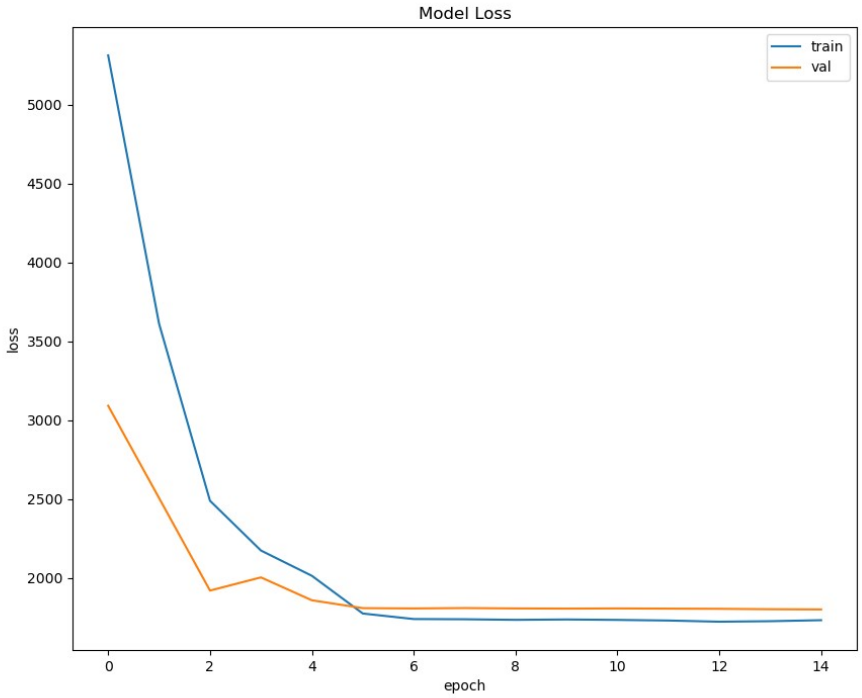


Figure 8.1: Training and validation learning curves resulted using Data Augmentation 1

Once the training is completed, the next step is the evaluation. To see how many epochs of training help to improve performance, the cumulative error distributions of the model for different epochs of training were calculated. The criterion of choosing the epochs for which the model was evaluated is: the value of the loss calculated on the validation dataset must be less than the minimum value up to that moment. The evaluation results are presented in Figure 8.2.

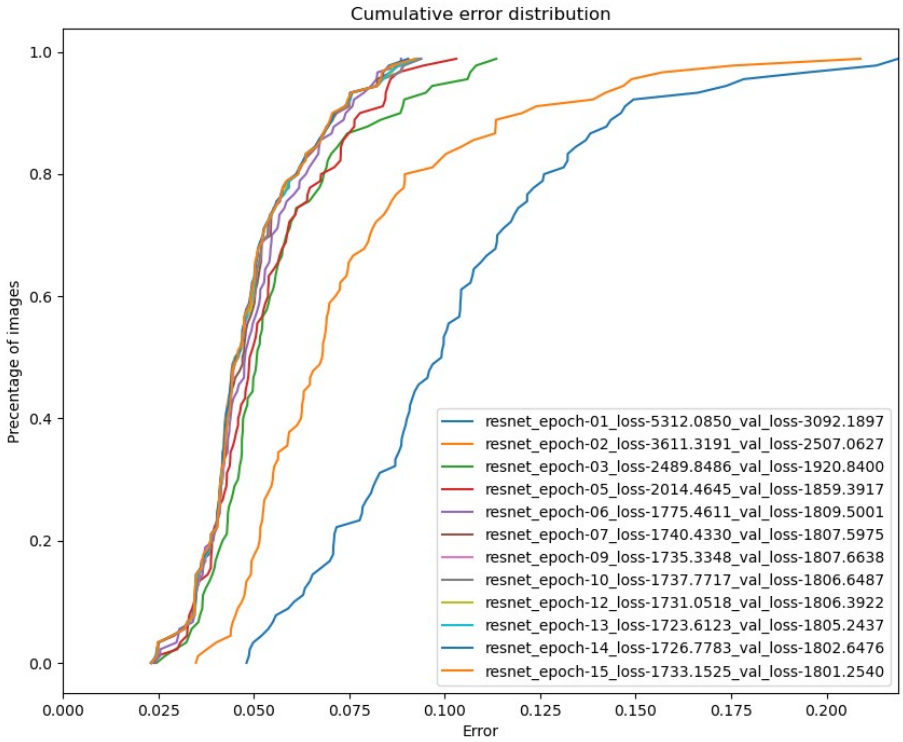


Figure 8.2: Cumulative error distribution calculated on 55 landmarks on the testing dataset using neural networks trained on Data Augmentation 1

Both Figure 8.1 and Figure 8.2 lead to the same conclusion, namely that once a relatively constant value of the loss function is obtained, the performance of the model seems to be capped. In these conditions, it was chosen to be designated the best model for this pipeline, the version saved with the most training epochs.

8.2. Data augmentation 2 - Results

Figure 8.3 reveals that model seems to reach to a saturated loss value after 13 epochs of training. Continuing training does not result in any major improvement in the value of the loss.

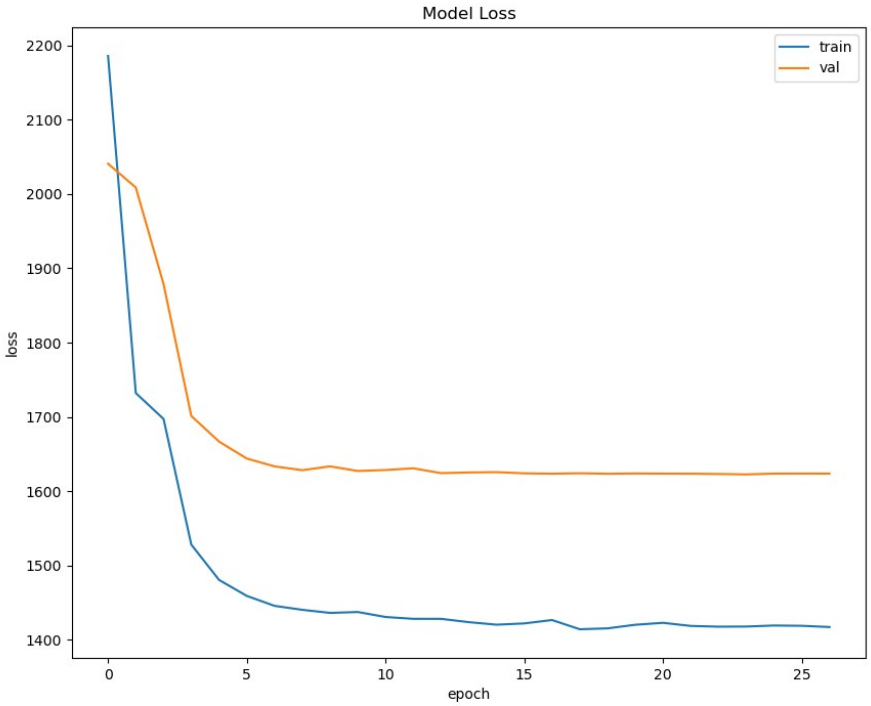


Figure 8.3: Training and validation learning curves resulted using Data Augmentation 2

As explained in Data augmentation 1 - Results, the curves of the cumulative distribution errors of the model for different training stages are displayed. Thus, the result of the evaluation can be seen in Figure 8.4.

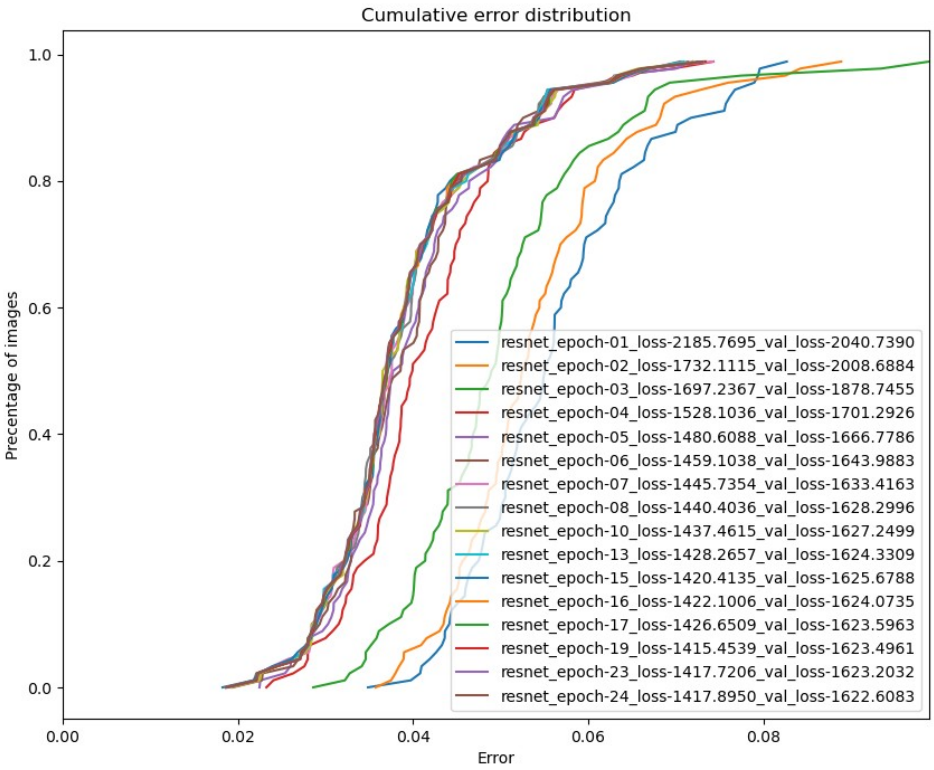


Figure 8.4: Cumulative error distribution calculated on 55 landmarks on the testing dataset using neural networks trained on Data Augmentation 2

Figure 8.3 and Figure 8.4 lead to the same conclusion as described in section Data augmentation 1 - Results. In these conditions, it was chosen to be designated the best model for this pipeline, the version saved with the most training epochs.

8.3. Data augmentation 3 - Results

Figure 8.5 shows that model seems to reach to a saturated value for the loss after 7 epochs of training. Continuing training does not result in any major improvement in the value of the loss.

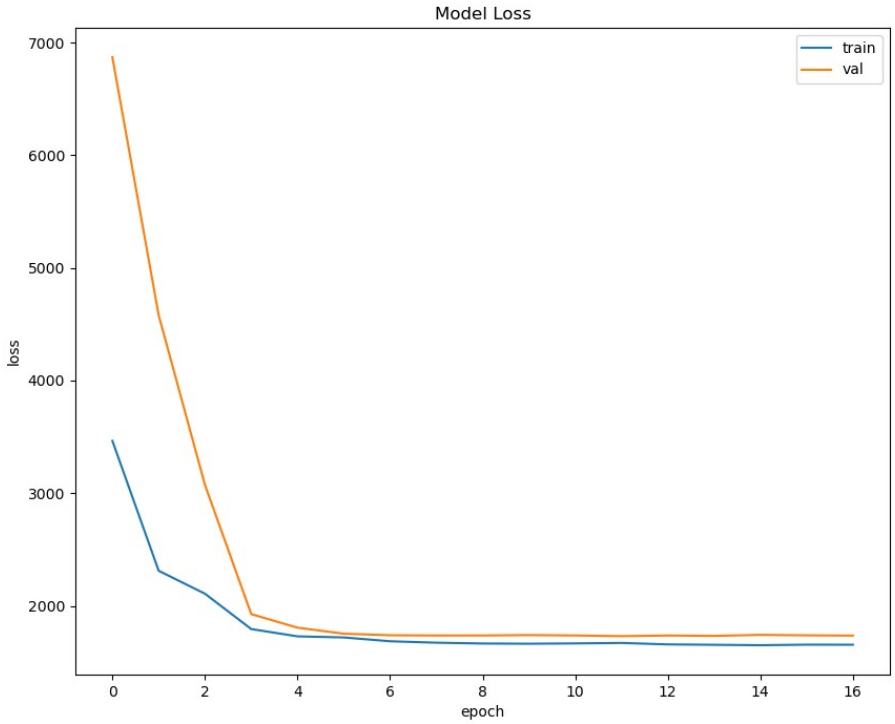


Figure 8.5: Training and validation learning curves resulted using Data Augmentation 3

As explained in Data augmentation 1 - Results, the curves of the cumulative distribution errors of the model for different training stages are displayed. Thus, the result of the evaluation can be seen in Figure 8.6.

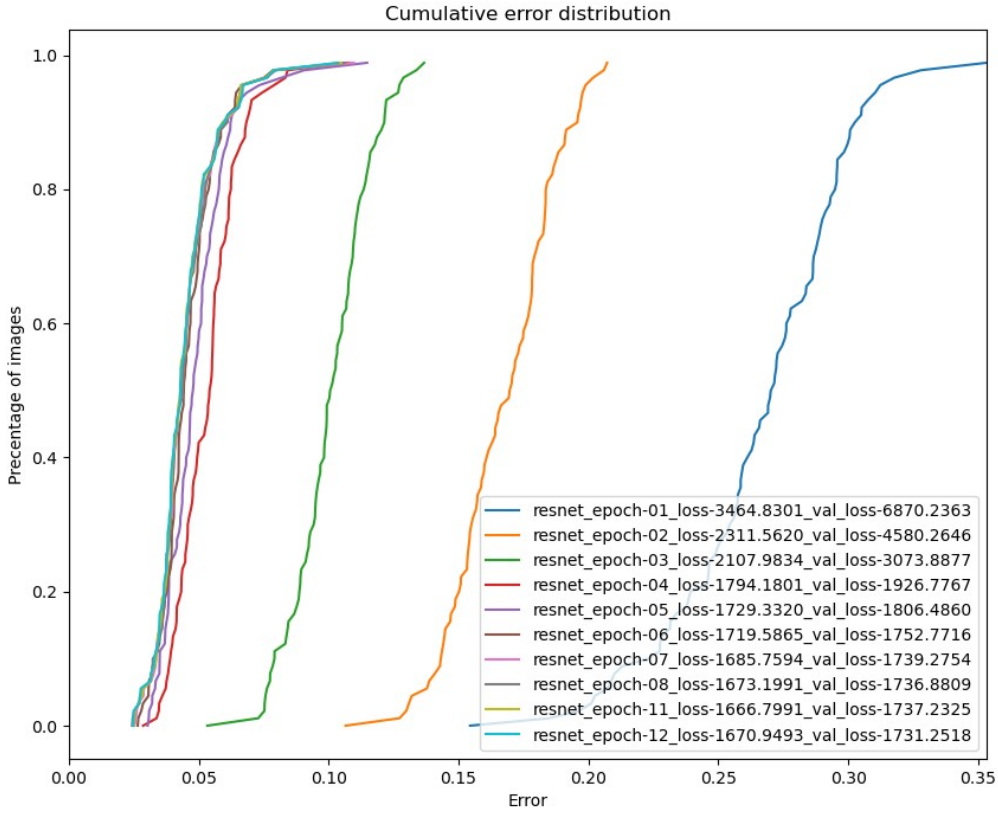


Figure 8.6: Cumulative error distribution calculated on 55 landmarks on the testing dataset using neural networks trained on Data Augmentation 3

Figure 8.5 and Figure 8.6 lead to the same conclusion as described in section Data augmentation 1 - Results. In these conditions, it was chosen to be designated the best model for this pipeline, the version saved with the most training epochs.

8.4. Data augmentation 4 - Results

Figure 8.7 reveals that the saturation in the loss values is reached after 12 epochs of training. Continuing training will not result any major improvement in the value of the loss.

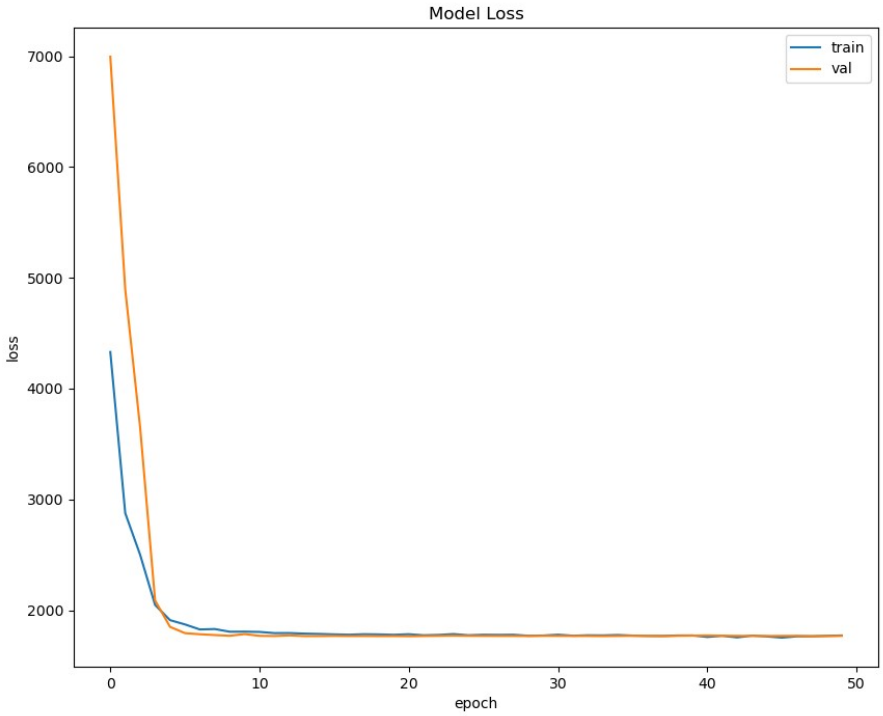


Figure 8.7: Training and validation learning curves resulted using Data Augmentation 4

As explained in Data augmentation 1 - Results, the curves of the cumulative distribution errors of the model for different training stages are displayed. Thus, the result of the evaluation can be seen in Figure 8.8.

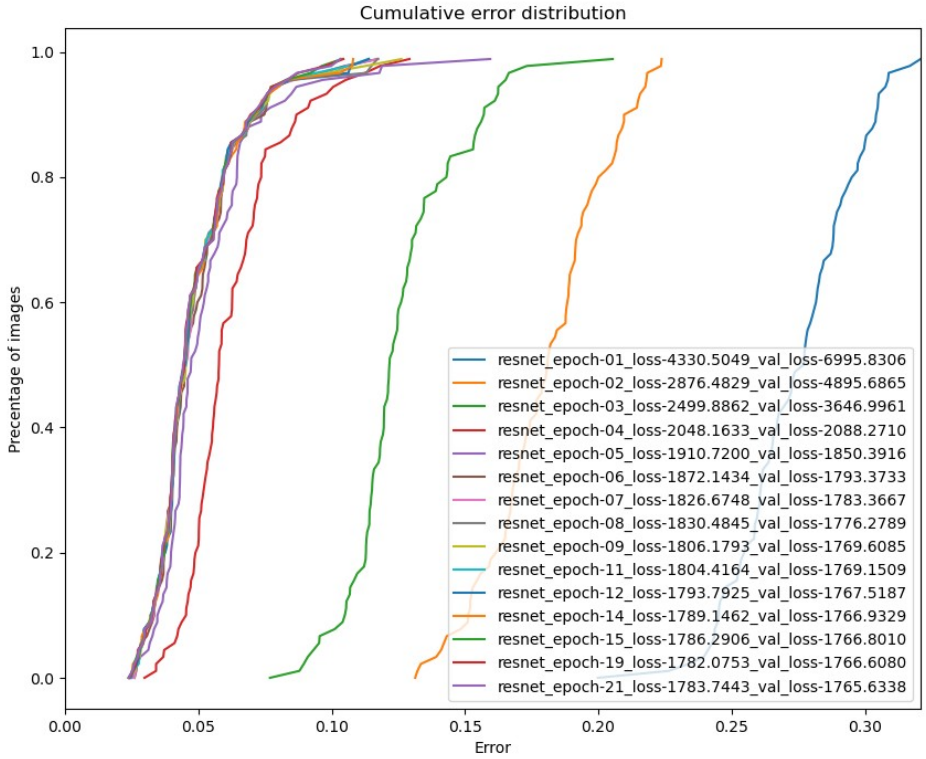


Figure 8.8: Cumulative error distribution calculated on 55 landmarks on the testing dataset using neural networks trained on Data Augmentation 4

Figure 8.7 and Figure 8.8 lead to the same conclusion as described in section Data augmentation 1 - Results. In these conditions, it was chosen to be designated the best model for this pipeline, the version saved with the most training epochs.

8.5. Data augmentation 5 - Results

Figure 8.9 shows that, in this case, the model reaches the saturation of the loss value after 7 epochs of training. Continuing training does not have any major improvement in the value of the loss.

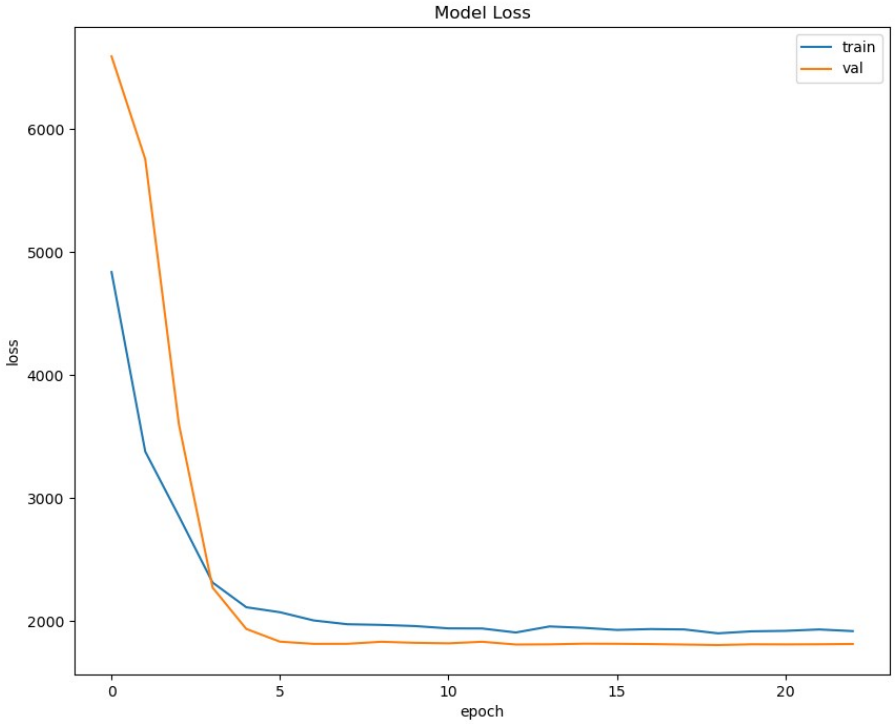


Figure 8.9: Training and validation learning curves resulted using Data Augmentation 5

As explained in Data augmentation 1 - Results, the curves of the cumulative distribution errors of the model for different training stages are displayed. Thus, the result of the evaluation can be seen in Figure 8.10.

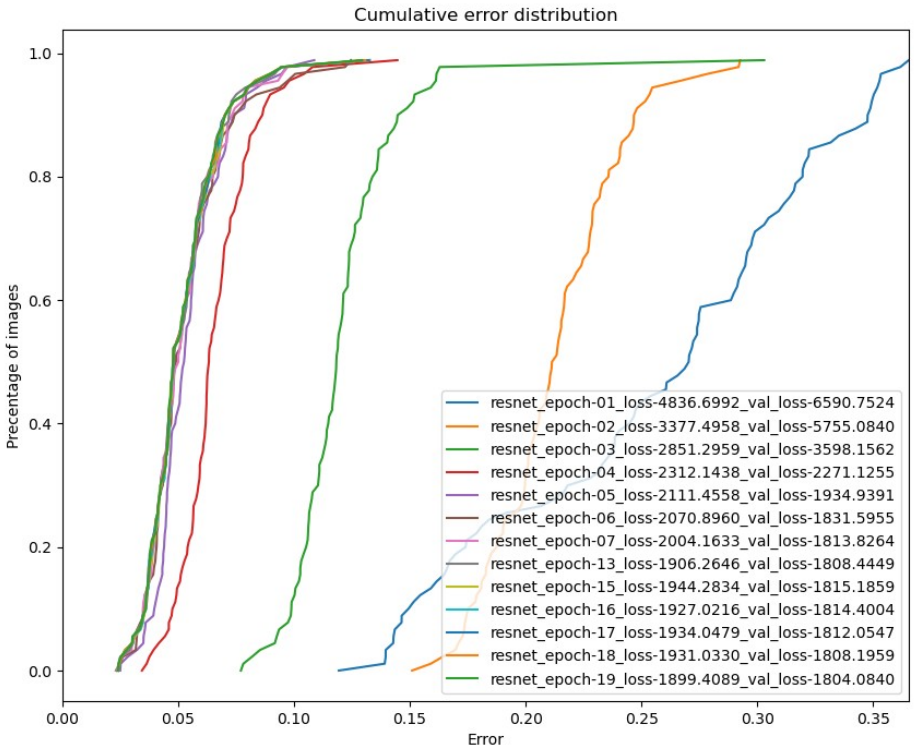


Figure 8.10: Cumulative error distribution calculated on 55 landmarks on the testing dataset using neural networks trained on Data Augmentation 55

Figure 8.9 and Figure 8.10 lead to the same conclusion as described in section Data augmentation 1 - Results. In these conditions, it was chosen to be designated the best model for this pipeline, the version saved with the most training epochs.

9. Conclusion

Now that the results of the trained models using different augmentation pipelines have been analyzed separately and the best model for each pipeline has been chosen, the next step is to make a comparison between them. For a more accurate comparison, the evaluation results will be used.

Figure 9.1 summarizes the results for all the pipelines. As it can be easily seen, the best results were obtained using the Data Augmentation 2 pipeline. The second best model is the one that was trained using the Data Augmentation 3 pipeline. Table 2 provides a visual comparison between the ground truth and performances of the evaluated models.

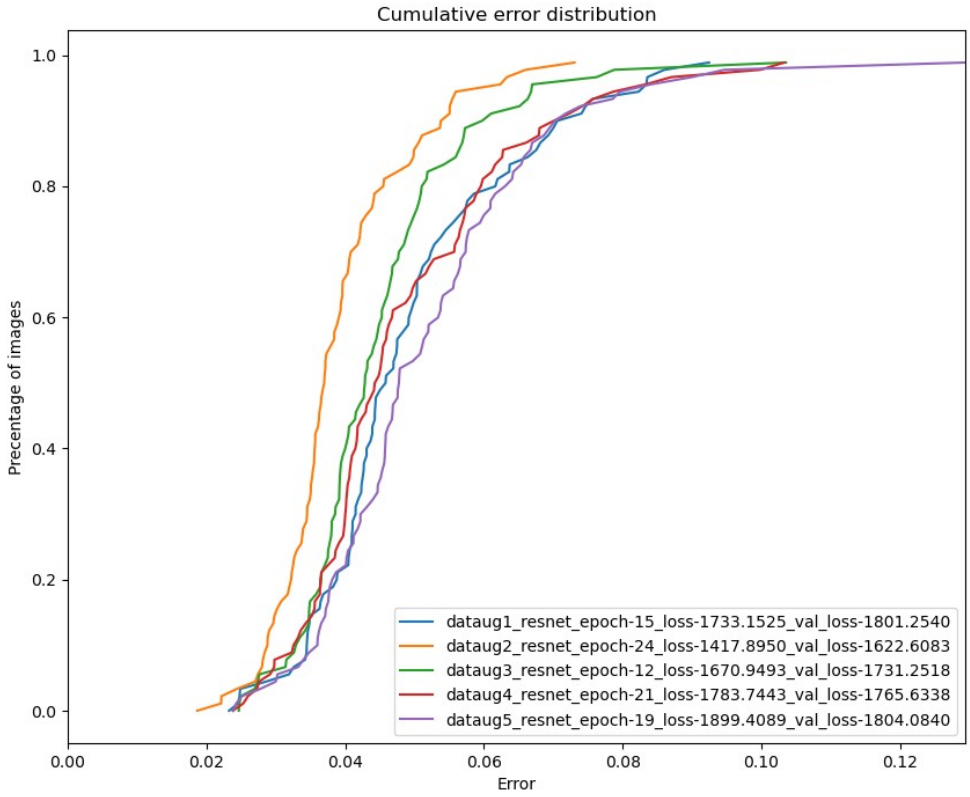
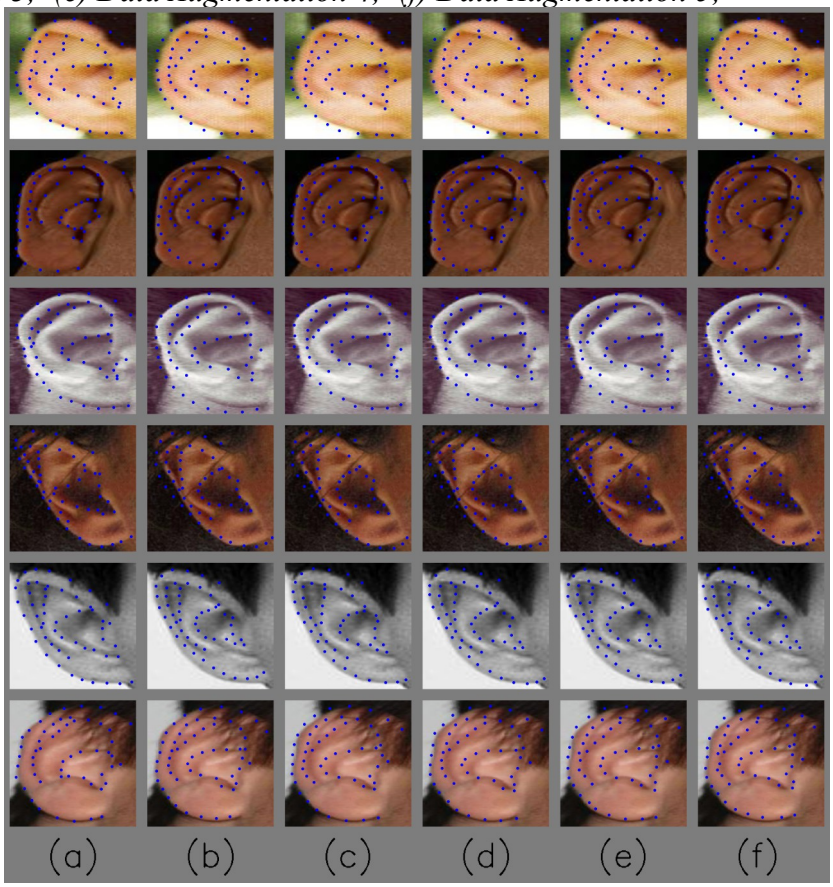


Figure 9.1: Cumulative error distribution calculated on 55 landmarks on the testing dataset using the best neural networks trained on each data augmentation pipeline

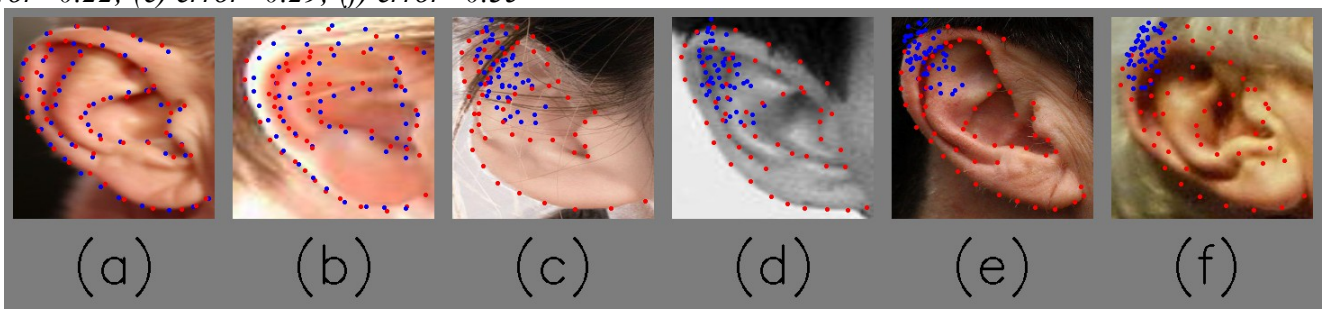
Table 9.1: Comparison between ground truth and predictions of the models trained using different data augmentation pipelines: (a) Ground Truth; (b) Data Augmentation 1; (c) Data Augmentation 2; (d) Data Augmentation 3; (e) Data Augmentation 4; (f) Data Augmentation 5;



Increasing the training dataset by flipping the images horizontally or vertically does not improve the performance of the model. Also, enlarging the dataset with a bigger rotation range does not lead to better results. Given the size of the available training dataset and the current architecture of the neural network, it can be concluded that a less aggressive augmentation pipeline produces better results.

To clearly underline the obtained performances, Table 9.2 presents a visual interpretation for different error values. The ground truth is plotted in the images using red points, while the blue points are the predicted points. An error of 0.018 converted to pixels is approximately 5.7 pixels. This means that for case (a) in the Table 9.2, the distance between the predicted and the real points is on average 5.7 pixels. On the other hand, the image (f) has on average an error of 110.87 pixels between the points.

Table 9.2: Visual interpretation of errors: (a) error=0.018; (b) error=0.073; (c) error=0.15; (d) error=0.22; (e) error=0.29; (f) error=0.35



Bibliography

1: Algazi V R., Duda R. O., Thompson D. M., Avendano C., THE CIPIC HRTF DATABASE, 2001
2: Hang Dai, Nick Pears and William Smith, A Data-augmented 3D Morphable Model of the Ear, 2018
3: AYMAN ABAZA, ARUN ROSS, CHRISTINA HEBERT, MARY ANN F. HARRISON, MARY ANN F. HARRISON, A Survey on Ear Biometrics, 2013
4: Stylianos Ploumpis, Evangelos Ververas, Eimear O' Sullivan, Stylianos Moschoglou,Haoyang Wang, Nick Pears, William A. P. Smith, Baris Gecer, and Stefanos Zafeiriou, Towards a complete 3D morphable modelof the human head, 2020
5: Joseph E. Hawkins, Human ear, 06/09/2021, https://www.britannica.com/science/ear
6: Adrian Bulat, Georgios Tzimiropoulos, How far are we from solving the 2D & 3D Face Alignment problem? (and a dataset of 230,000 3D facial landmarks), 2017
7: Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, Stan Z. Li, Towards Fast, Accurate and Stable 3D Dense Face Alignment, 2021
8: Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, Xin Tong, Accurate 3D Face Reconstruction with Weakly-Supervised Learning: From Single Image to Image Set, 2020
9: Yao Feng, Haiwen Feng, Michael J. Black, Timo Bolkart, Learning an Animatable Detailed 3D Face Model from In-The-Wild Images, 2020
10: Yuxiang Zhou, Stefanos Zaferiou, Deformable Models of Ears in-the-wild for Alignment and Recognition, 2017
11: Earnest E. Hansley, Mauricio Pamplona Segundo, Sudeep Sarkar, Employing Fusion of Learned and HandcraftedFeatures for Unconstrained Ear Recognition, 2017
12: Hao Sun, Nick Pears, Hang Dai, A Human Ear Reconstruction Autoencoder, 2020
13: N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, 2005
14: O. M. Parkhi, A. Vedaldi, and A. Zisserman, Deep face recognition, 2015
15: Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, SSD: Single Shot MultiBox Detector, 2015
16: Pierluigi Ferrari, SSD: Single-Shot MultiBox Detector implementation in Keras, , https://github.com/pierluigiferrari/ssd_keras#overview
17: Chadrick's Blog, cv2 resize interpolation methods, November 14, 2018, https://chadrick-kwag.net/cv2-resize-interpolation-methods/
18: mxnet, Types of Data Augmentation, , https://mxnet.apache.org/versions/1.2.1/tutorials/python/types_of_data_augmentation.html
19: Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep Residual Learning for Image Recognition, 2015
20: DeepLearning.AI, Convolutional Neural Networks,
21: Diederik P. Kingma, Jimmy Ba, Adam: A Method for Stochastic Optimization, 2017
22: Zhen-Hua Feng, Josef Kittler, Muhammad Awais, Patrik Huber, Xiao-Jun Wu, Wing Loss for Robust Facial Landmark Localisation with Convolutional Neural Networks, 2018
23: DeepLearning.AI, Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization,