

Master Thesis



Czech  
Technical  
University  
in Prague

**F3**

Faculty of Electrical Engineering  
Department of Cybernetics

# Automatic Analysis of Facial Wrinkle Characteristics in People with Parkinson's Disease

**Bc. Jan Vaník**

Supervisor: Ing. Michal Novotný, PhD.  
May 2024

## I. Personal and study details

Student's name: **Vaník Jan** Personal ID number: **491878**  
Faculty / Institute: **Faculty of Electrical Engineering**  
Department / Institute: **Department of Cybernetics**  
Study program: **Medical Electronics and Bioinformatics**  
Specialisation: **Image processing**

## II. Master's thesis details

Master's thesis title in English:

**Automatic Analysis of Facial Wrinkle Characteristics in People with Parkinson's Disease**

Master's thesis title in Czech:

**Automatická analýza vráskových charakteristik pacientů s Parkinsonovou nemocí**

Guidelines:

The loss of facial expressivity, called hypomimia, is one of the common manifestations of Parkinson's disease. It has been shown that facial expressivity disruption is one of the early symptoms that can be present ten years before the diagnosis. However, due to the lack of tools enabling objective automatic assessment, hypomimia evaluation is performed only to a limited extent. Modern digital image processing methods and machine learning approaches provide an opportunity to evaluate microexpressions and deep analysis of facial texture. The goals of the thesis are:

1. Study the topic of facial expressivity disruption in people with Parkinson's disease.
2. Study the topic of facial wrinkle assessment.
3. Design an automatic approach for the assessment of facial expressivity disruption based on the evaluation of facial wrinkles.
4. Analyze differences in the facial characteristics of people with Parkinson's disease and healthy control group.

Bibliography / sources:

- [1] Yap, M. H., Batool, N., Ng, C. C., Rogers, M., & Walker, K. (2021). A survey on facial wrinkles detection and inpainting: datasets, methods, and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(4), 505-519.
- [2] Sanchez, M., Triginer, G., Ballester, C., Raad, L., & Ramon, E. (2022, December). Photorealistic Facial Wrinkles Removal. In *Asian Conference on Computer Vision* (pp. 117-133). Cham: Springer Nature Switzerland.
- [3] Sabina, U., & Whangbo, T. K. (2021, October). Nasolabial Wrinkle Segmentation Based on Nested Convolutional Neural Network. In *2021 International Conference on Information and Communication Technology Convergence (ICTC)* (pp. 483-485). IEEE.
- [4] Kim, S., Yoon, H., Lee, J., & Yoo, S. (2022, July). Semi-automatic Labeling and Training Strategy for Deep Learning-based Facial Wrinkle Detection. In *2022 IEEE 35th International Symposium on Computer-Based Medical Systems (CBMS)* (pp. 383-388). IEEE.

Name and workplace of master's thesis supervisor:

**Ing. Michal Novotný, Ph.D. Department of Circuit Theory FEE**

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **15.02.2024** Deadline for master's thesis submission: **24.05.2024**

Assignment valid until: **21.09.2025**

Ing. Michal Novotný, Ph.D.  
Supervisor's signature

prof. Dr. Ing. Jan Kybic  
Head of department's signature

prof. Mgr. Petr Páta, Ph.D.  
Dean's signature

### III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

\_\_\_\_\_  
Date of assignment receipt

\_\_\_\_\_  
Student's signature

## Acknowledgements

Předně bych chtěl poděkovat svému vedoucímu doktoru Michalovi Novotnému za vstřícnost, trpělivost, užitečné rady a zejména ochotu vést moji diplomovou práci.

Dále děkuji svým rodičům, ségrám, prarodičům a dalším členům naší rodiny, kteří mě během studia velmi podporovali, ať už finančně, vlídným slovem, vypráním prádla či uvařením oběda.

Velmi si vážím i všech přátel a kolegů ze školních lavic, kteří z náročného studia udělali během semestrů příjemně strávených nezapomenutelných pět let.

Abych neopomenul, rozesmálo mě i množství lidí, kteří při zjištění, jaké téma jsem si zvolil, na mě vražtili čelo a ptali se: "Tak co, mám Parkinsona?"

## Declaration

I declare that the presented work was developed independently and that I have listed all sources of information within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses. ChatGPT and Grammarly were used as tools for stylistic and grammatical checks in accordance with CTU FEE guidelines.

Prague, May 24, 2024

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. ChatGPT a Grammarly byly použity jako nástroje pro stylistickou a pravopisnou kontrolu v souladu se směrnicemi ČVUT FEL.

V Praze, 24. května 2024

## Abstract

Parkinson's disease is a severe, the second most common progressive neurodegenerative disease. With no causal treatment, current research focuses on finding biomarkers that indicate the onset of the disease before full manifestation. One such biomarker is hypomimia, a lowered magnitude of facial movement, which disrupts wrinkle formation. In recent years wrinkle segmentation with neural networks has become widely used. Due to no sufficient public wrinkle dataset being available, we assembled an original dataset with 674 total high-resolution images being annotated. An architecture trained on this dataset achieved a Jacquard Similarity Index (JSI) of 31.2 %. This model was used to segment wrinkles in video recordings of de-novo-diagnosed 100 Parkinson's disease patients and their matched healthy control counterparts. The parameters calculated from wrinkles represented hypomimia characteristics according to the literature. These features showed significant differences between Parkinson's disease patients and healthy controls. Logistic regression fitted to these features with leave-one-out cross-validation resulted in an accuracy of 74 %. Features extracted from wrinkles segmented by neural networks can indicate Parkinson's disease in an early stage; however, detected wrinkles by neural networks can be unstable during video processing.

**Keywords:** Parkinson's disease, wrinkles, segmentation

**Supervisor:** Ing. Michal Novotný, PhD.  
Praha  
Technická 1902/2  
místopis: B3-613

## Abstrakt

Parkinsonova nemoc je závažné progresivní druhé nejrozšířenější neurodegenerativní onemocnění. Vzhledem k chybějící příčinné léčbě, se snahy vědců ubírají k detekci biomarkerů, které se objevují před plným rozvojem nemoci. Jedním ze zkoumaných biomarkerů je hypomimie, projevující se jako snížené výchylky v obličejových pohybech, která narušuje tvorbu vrásek. Pro segmentaci vrásek se v posledních letech začaly hojně používat neuronové sítě. Vzhledem k nedostatečným dostupným datasetům anotovaných vrásek, jsme zkompletovali vlastní dataset o 674 anotovaných obrázcích s vysokým rozlišením. Architektura natrénovaná na tomto datasetu dosahovala 31.2 % Jacquardova indexu podobnosti (JSI). Tento model byl použit pro segmentaci 100 videí pacientů s de-novo diagnostikovanou Parkinsonovou nemocí a odpovídající skupinou zdravých kontrol. Parametry vypočítané z vrásek reprezentovaly charakteristiky hypomimie podle dostupné literatury a ukázaly signifikantní odlišnost mezi skupinou pacientů a zdravých kontrol. Klasifikátor logistické regrese s křížovou validací typu leave-one-out byl natrénován na těchto parametrech s přesností 74 %. Parametry získané z vrásek segmentovaných za pomoci neuronové sítě mohou indikovat rané projevy Parkinsonovy nemoci, avšak segmentování vrásek pomocí neuronové sítě může být nestabilní při zpracování video záznamů.

**Klíčová slova:** Parkinsonova nemoc, vrásky, segmentace

**Překlad názvu:** Automatická analýza vráskových charakteristik pacientů s Parkinsonovou nemocí

# Contents

<b>1 Introduction</b>	<b>1</b>	4.1.2 Wrinkle Dataset . . . . .	37
<b>Part I</b>		4.2 Neural Network Training . . . . .	40
<b>Theory and Literature Review</b>		4.3 Video Processing . . . . .	41
<b>2 Parkinson's Disease</b>	<b>3</b>	4.3.1 Segmentation of Face Images . . . . .	41
2.1 Prevalence . . . . .	3	4.3.2 2D Face Alignment . . . . .	41
2.2 Patomechanism . . . . .	3	4.3.3 3D Face Aligmnemnt . . . . .	42
2.3 Symptoms . . . . .	3	4.3.4 Hidden Landmarks Elimination . . . . .	44
2.4 Diagnosis . . . . .	4	4.3.5 Wrinkle Parameters Calculation . . . . .	45
2.5 Treatment and Management . . . . .	5	4.4 Data analysis . . . . .	48
2.6 Biomarkers . . . . .	5	<b>5 Results</b>	<b>49</b>
2.7 Hypomimia . . . . .	6	5.1 Dataset Results . . . . .	49
<b>3 Segmentation of Digital Images</b>	<b>9</b>	5.2 Model Training Results . . . . .	49
3.1 Segmentation Metrics . . . . .	9	5.3 Face Alignment Results . . . . .	49
3.2 Convolution . . . . .	11	5.4 Video Evaluation Results . . . . .	50
3.3 Image Histogram . . . . .	13	5.5 Data analysis Results . . . . .	50
3.3.1 Histogram Equalization . . . . .	13	<b>6 Discussion</b>	<b>51</b>
3.4 Classical Image Segmentation Techniques . . . . .	14	6.1 Dataset . . . . .	51
3.4.1 Thresholding . . . . .	14	6.2 Wrinkle Segmentation Model Training . . . . .	51
3.4.2 Clustering . . . . .	14	6.3 Facial Alignment and Video Evaluation . . . . .	52
3.4.3 Edge Detection . . . . .	15	6.4 Data Analysis . . . . .	52
3.4.4 Graph Cut Based Segmentation . . . . .	15	6.5 Meeting the Thesis Goals . . . . .	53
3.4.5 Line Segmentation . . . . .	17	<b>7 Conclusion</b>	<b>55</b>
3.5 Image Segmentation with Convolutional Neural Networks . . . . .	20	<b>Bibliography</b>	<b>57</b>
3.5.1 Layers of CNNs . . . . .	20	<b>Appendices</b>	
3.5.2 Batch Normalization . . . . .	24	<b>A Results</b>	<b>66</b>
3.5.3 Losses . . . . .	25		
3.5.4 Training and Validation . . . . .	25		
3.5.5 CNN Architectures for Image Segmentation . . . . .	29		
3.6 Segmentation of Wrinkles . . . . .	30		
3.6.1 Facial and Wrinkle Datasets . . . . .	31		
3.6.2 Prior Work in Wrinkle Segmentation . . . . .	31		
3.6.3 Wrinkle Segmentation Summary . . . . .	34		
<b>Part II</b>			
<b>Research Project</b>			
<b>4 Methodology</b>	<b>36</b>		
4.1 Datasets . . . . .	36		
4.1.1 Dataset of Parkinson's Disease Patients and Healthy Controls . . . . .	36		

## Figures

2.1 PD patient and their symptoms .	4
2.2 Sketch of a PD patient's face . . . .	6
3.1 Convolution . . . . .	11
3.2 Padding . . . . .	12
3.3 Gaussian visualization . . . . .	12
3.4 Gaussian visualization . . . . .	13
3.5 Graphcut . . . . .	17
3.6 Hough transform . . . . .	19
3.7 Activation functions . . . . .	22
3.8 Pooling . . . . .	23
3.9 Dilatation . . . . .	24
3.10 Fractionally strided convolution	24
3.11 Learning rate schedulers . . . . .	27
3.12 Augmentation . . . . .	28
3.13 VGG . . . . .	29
3.14 Residual block . . . . .	30
3.15 Resnet . . . . .	31
3.16 Deeplab v3 . . . . .	32
3.17 UNet . . . . .	33
3.18 Wrinkles localization . . . . .	34
4.1 Selected images from FFHQ	
Stage1 . . . . .	38
4.2 Selected images from FFHQ Stage	
4 . . . . .	39
4.3 Landmarks . . . . .	42
4.4 Segmented face . . . . .	43
A.1 Hybrid Hessian Filter response .	66
A.2 Histogram equalization results .	67
A.3 CNN pre-annotation . . . . .	67
A.4 Manual annotation . . . . .	68
A.5 Wrinkle segmentation results . .	69
A.6 Landmarks elimination results .	70
A.7 Facial alignment . . . . .	71
A.8 $mean_{JSI}^{ALL2}$ . . . . .	72
A.9 $mean_{JSI}^{ALL5}$ . . . . .	73
A.10 $normstd_{TUBU}$ . . . . .	73
A.11 $mean_{JSI}^{U-B5}$ . . . . .	74
A.12 ROC . . . . .	74
A.13 Confusion Matrix . . . . .	75

## Tables


2.1 Hypomimia detection comparism	8
3.1 Edge Detection Comparison . . . .	16
4.1 Clinical data . . . . .	37
5.1 Models Performance . . . . .	50
A.1 Statistical tests results . . . . .	70
A.2 Means and stds of calculated values . . . . .	72

# Chapter 1

## Introduction

Parkinson's disease (PD) is a progressive neurodegenerative disorder caused by the loss of dopaminergic neurons in the substantia nigra pars compacta (SNc) region of the midbrain. Although the causes of the disease are not yet exactly known, age is identified as one of the main risk factors. PD primarily affects individuals over the age of 50. PD is the second most common neurodegenerative disease after Alzheimer's disease [1]. With the population getting older overall, it is realistic to expect increasing numbers of patients with PD. It is important to note that PD is not yet fully understood [2].

Because of the severe impact on the quality of a patient's life, there is a need to find a causal treatment for PD. There have been attempts to find early biomarkers that could diagnose PD in prodromal stages [2]. One of the first occurring motor symptoms is hypomimia [3]. An important sign of facial expressivity disruption is a change in movement and formation of wrinkles [4]. Therefore, reliable wrinkle segmentation and detection of significant wrinkle parameters could be used to reveal PD in the early stages, a possible approach to finding wrinkle parameters for hypomimia is debated in bachelor thesis [5] written in Czech. Wrinkle segmentation has risen in popularity because of the recent expansion of neural networks, which outperform prior basic image segmentation techniques.

The goal of this diploma thesis is to study facial disruption of wrinkle formatting caused by hypomimia, review approaches for wrinkle segmentation, and propose a method for an automatic evaluation of wrinkle disruptions caused by Parkinson's disease. This diploma thesis is divided into two parts. In Part I we briefly discuss Parkinson's disease and look into hypomimia and previous approaches for its detection in Chapter 2; then image segmentation techniques, neural networks, and prior work in wrinkle segmentation are debated in Chapter 3. The research project in Part II consisting of wrinkle dataset assembling, training of a neural network model, and video analysis is divided into methodology, results, and discussion (Chapters 4, 5 and 6) concluded in Chapter 7. The dataset and video-analyzing code were uploaded to GitHub  for easy access. Results too large to be displayed in the main text can be found in Appendix A.





## **Part I**

### **Theory and Literature Review**

## Chapter 2

# Parkinson's Disease

### 2.1 Prevalence

Parkinson's disease (PD) primarily affects individuals over the age of 50 years, with occurrences in younger age groups being extremely rare. In the population, the incidence is 0.3%, but it grows to 3 % among individuals aged over 80. This makes it the second most common neurodegenerative disease after Alzheimer's disease [1]. With the population getting older overall, it is realistic to expect increasing numbers of patients with PD [2]. PD is rarely inherited, but different ethnicities can vary in the incidence of PD [1]. It has been shown that chemical exposures can alter the probability of PD onset, with smokers and caffeine users being less likely to develop PD [1].

### 2.2 Patomechanism

Two factors cause PD: (i) loss of dopaminergic neurons in substantia nigra pars compacta (SNc) and (ii) deposition of  $\alpha$ -synuclein in neurons, for PD specifically, both factors mentioned must be present [1]. This results in motor and mental complications [1, 3]. The cause of changes in the human body leading to PD is not yet fully understood and is viewed as multifactorial [1].

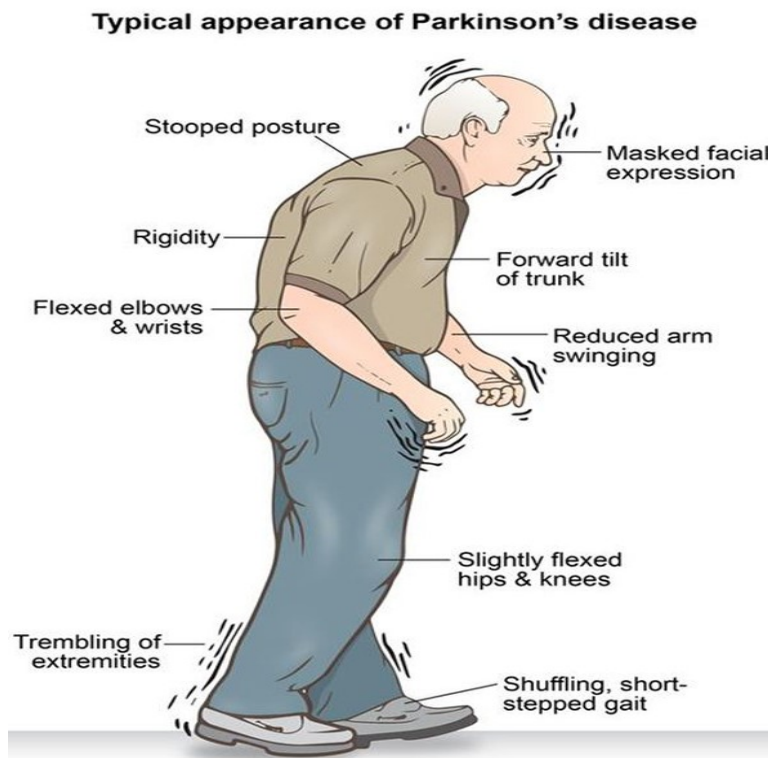
Dopamine is a neurotransmitter with diverse functions. In motor pathways, it modifies responses in basal ganglia, so movements are more sudden. The exact motor wiring is beyond the scope of this work and for deeper understanding, see [6, 7]. Lack of dopamine then induces problems with a sudden voluntary starting and ceasing of a movement accompanied by tremors and progressively more severe cognitive impairment [3].

### 2.3 Symptoms

The first occurring motor symptoms are best remembered by an abbreviation TRAP: tremor, rigidity, akinesia, and postural instability. The tremor mostly dominates one side and its average frequency is approximately 5 Hz. Rigidity is described as a force preventing one from moving, whereas akinesia is a

lack of movement. Postural instability increases the risk of falls, resulting in fractures, particularly in the advanced stages of the disease. Besides motor symptoms, PD patients also exhibit depression, fatigue, and REM sleep disorders [1, 3, 8]. Milder forms of these abnormalities are discussed in Section 2.6. The most used scale for measuring the severity of PD symptoms is the Unified Parkinson's Disease Rating Scale (UPDRS) described by the Movement Disorder Society (MDS). This MDS-UPDRS rating has to be performed by specialized physicians or nurses [9].

As the disease progresses, the severity of the symptoms intensifies. In its final stages, most of the patients are barely able to move, talk, or swallow and suffer from cognitive impairment, often leading to dementia [3]. PD patients' quality of life is lowered, as well as life expectancy. PD patients are expected to live approximately 5 years shorter than people without diagnosed PD. There exist many causes of death among PD patients, with pneumonia being a significantly more frequent cause compared to healthy controls [10]. Figure 2.1 depicts a PD patient with the typical symptoms.



**Figure 2.1:** PD patient and their symptoms. Image source: [11].

## 2.4 Diagnosis

As mentioned in Section 2.2, the loss of dopaminergic neurons in SNc causes Parkinson's disease. Therefore, neuroimaging would be ideal for definitive

PD diagnosis. This approach would be costly, which is why diagnosis is done differently. MDS provides commonly used guidelines for how PD should be diagnosed. It offers supportive criteria, exclusion criteria, and red flags. If any exclusion criterion is present, PD is not diagnosed. In other cases, it depends on the ratio of the supportive criteria and the red flags. Supportive criteria can be a beneficial response to dopaminergic therapy or rest tremor of a limb. Among the exclusion criteria are normal functional neuroimaging of the presynaptic dopaminergic system or an absence of observable response to high-dose levodopa. Red flags contain a complete absence of progression of motor symptoms and also a rapid progression of gait impairment. For more information about diagnosis, please read the original paper [12].

## ■ 2.5 Treatment and Management

To this day, there is no approved medication to cure or even slow down PD's progression [2]. The main focus of current treatment is to diminish symptoms and consequently improve the patient's quality of life. The standard most used medication is called levodopa, this drug can be ingested and can penetrate the blood-brain barrier. In the brain, levodopa transforms into dopamine to substitute its lack caused by PD. There are other chemicals used for PD treatment and their exact mix depends on each individual's reactions and side effects [1,2]. For more information about the precise treatment mechanism, see [1].

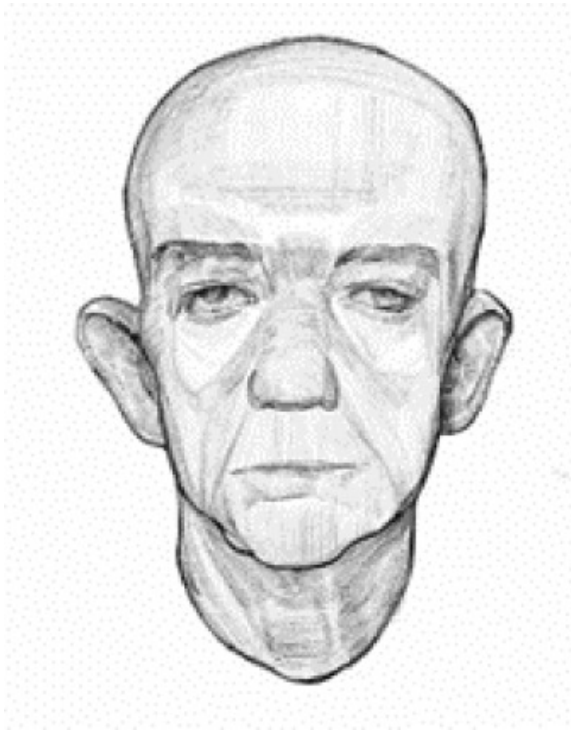
Another treatment option is to use electrodes to stimulate the brain from the inside with a signal of low amplitude and high frequency. This method is called deep brain stimulation (DBS). It can also improve the quality of life of a patient, especially when they build resistance to medications. In the future, hopes for causal treatment are placed in gene therapy or dopaminergic cell implantation, however, these approaches are only in the early development stages. Diagnosing PD in the prodromal stage could also be beneficial to the treatment because, at that time, fewer dopaminergic neurons are lost [1,2].

## ■ 2.6 Biomarkers

The scientific community puts effort into discovering biomarkers - early signs of PD that can be revealed before the diagnosis. Most discussed motor biomarkers are milder forms of akinesia and rigidity: (i) hypomimia (also called facial bradykinesia) [13–21], (ii) dysarthria [22–26], (iii) dysphagia [27], (iv) reduced arm swing [28,29], and (v) sialorrhoea (drooling) [30,31]. However, there is no consensus on sialorrhoea being an early sign [32]. Other studies suggest non-motor symptoms can be used as biomarkers: (vi) composition of bodily fluids [33,34] or (vii) measuring sleep disorders [35]. A study from 2013 suggests that men suffering from idiopathic rapid eye movement sleep behavior disorder (iRBD) will develop either PD or dementia with an 81% chance [36].

## 2.7 Hypomimia

Hypomimia (facial bradykinesia) is one of the earliest symptoms of PD. Mimic muscles differ significantly from skeletal muscles mainly in two ways; most are not attached to a bone and can be controlled voluntarily and involuntarily. Research suggests that hypomimia manifests in (i) reduced spontaneous smiling - both amplitude and frequency, (ii) reduced blinking rate in early stages, (iii) flattened nasolabial folds, (iv) widened palpebral fissures, (v) reduced movement of the upper part of face while smiling, (vi) unintentional mouth opening, (vii) reduced number of wrinkles or creases around the mouth, and (viii) reduced range of lip movement. All factors together result in PD patients having more emotionless facial expressions. Many of the symptoms above improve with levodopa treatment [4,37]. A sketch of a PD's patient's face is illustrated in Figure 2.2.



**Figure 2.2:** Sketch of a PD patient's face. Image source: [19]

### ■ Prior Work in Hypomimia Evaluation

Most prior work in hypomimia detection relies on facial landmarks to varying degrees. Facial landmarks are important points on a human face, e.g. mouth corners, eyes, or a nose tip. There are various available algorithms used for facial landmark detection. For hypomimia evaluation, only one approach uses a singular image. Evaluating only one image saves memory and time, but it sacrifices the accuracy of the classification of healthy controls from

PD patients. Other approaches utilize more images of the same person, vast majority use videos [13–21]. Table 2.1 summarizes previous approaches in hypomimia evaluation techniques.

Author, Year	Input	Method
Grammatikopoulou, 2019	Images	Variability and means of locations of facial landmarks were used to analyze a series of selfies from different places and times [13].
Maycas-Cepeda, 2021	Video	Blink rate of recorded speech videos was evaluated [14].
Su, 2021	Video	Histogram of oriented gradients was used to analyze smiling videos [15].
Novotny, 2022	Video	Variability and means of locations of facial landmarks as well as the entropy of facial segments were used to analyze monologue videos [16].
Marsili, 2014	3D Video	3 IR videorecorders were used to analyze the movement of reflective markers attached to the smiling patient's skin [17].
Jin, 2020	Video	Jitter of facial landmarks was calculated and the most important landmarks for hypomimia detection were established from video recordings [18].
Liqiong, 2022	Video	Third-party software was used to analyze emotions from videos of slideshow readings [19].
Bandini, 2017	Video	Emotion expression was analyzed from videos of patients asked to show different emotions. The emotion estimation was performed from the location of facial landmarks [20].
Rajnoha, 2018	Image	A CNN was used for feature extraction of a single image. The features were then used to train a random forest tree classifier [21].

**Table 2.1:** Comparison of prior work in hypomimia analysis. Input = Input needed to evaluate a single person.

## Chapter 3

# Segmentation of Digital Images

A grayscale image  $I$  in the computer memory is represented by a  $I \in \mathbb{R}^{H \times W}$  matrix, where we call  $H$  the image height and  $W$  image width, cells of this matrix are called pixels. Let  $I_{i,j}$  represent a pixel in the  $i$ th row and  $j$ th column. The value of  $I_{i,j}$  is called the image intensity, which tells how much the corresponding pixel shines. Image intensities are often stored as non-negative integers ranging from  $I_{i,j} \in [0, 255]$ . Value 0 of  $I_{i,j}$  results in the corresponding pixel not shining and its color is black, whereas value 255 results in a white pixel. Note that we will be using zero-based indexing in the future, meaning  $i$  ranges from 0 to  $H - 1$  and  $j$  from 0 to  $W - 1$ . Thus  $I_{0,0}$  denotes the pixel in the upper left corner of image  $I$ .

A color image  $I^{RGB}$  has three channels  $C = 3$  for each color red, green, blue, and  $I^{RGB} \in \mathbb{R}^{H \times W \times C}$  three-dimensional matrix. When we refer to a pixel at coordinates  $i, j$  of a color image  $I^{RGB}$  we mean triplet of values for each channel, thus  $I_{i,j,:}^{RGB} \in \mathbb{R}^3$ , where  $:$  denotes all values alongside this dimension.

In image segmentation, we assign a class to each pixel in the image  $I$ . Let  $M$  denote a  $H \times W$  matrix called a segmentation mask (further abbreviated to just a mask) for the image  $I$  or  $I^{RGB}$ . The values of  $M$  are integers and they correspond to a class associated with their value. In binary segmentation, these classes are called foreground and background. Usually,  $M_{i,j} = 0$  indicates that an image pixel in the coordinates  $(i, j)$  corresponds to the background, and  $M_{i,j} = 1$  denotes the foreground pixel.

### 3.1 Segmentation Metrics

Segmentation metrics, which evaluate the performance of algorithms, require both a mask of ground truth labels and a mask of the resulting image segmentation predictions. In image segmentation, we denote foreground as a positive class and background as a negative class. True positives (TP) is defined as the sum of all correctly annotated foreground pixels, see Equation 3.1.



$$TP = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [pred(i, j) = 1 \wedge truth(i, j) = 1] \quad (3.1)$$

where  $pred$  is a mask of predicted labels,  $truth$  is a mask of ground truth labels and  $[\cdot]$  is the Iverson bracket (it is equal to 1 if the inside is true and 0 otherwise). True negatives (TN), false positives (FP), and false negatives (FN) are defined similarly in Equations 3.2 - 3.4.

$$TN = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [pred(i, j) = 0 \wedge truth(i, j) = 0] \quad (3.2)$$

$$FP = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [pred(i, j) = 1 \wedge truth(i, j) = 0] \quad (3.3)$$

$$FN = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [pred(i, j) = 0 \wedge truth(i, j) = 1] \quad (3.4)$$

From these four basic metrics, the following metrics can be derived [38]:

- Accuracy (ACC)

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.5)$$

- Precision

$$PRECISION = \frac{TP}{TP + FP} \quad (3.6)$$

- Recall

$$RECALL = \frac{TP}{TP + FN} \quad (3.7)$$

- F1 score (F1)

$$F1 = 2 \times \frac{PRECISION \times RECALL}{PRECISION + RECALL} \quad (3.8)$$

- DICE Similarity Coefficient (DSC)

$$DSC = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (3.9)$$

- Jaccard Similarity Index (JSI)

$$JSI = \frac{TP}{TP + FP + FN} = \frac{DSC}{2 - DSC} \quad (3.10)$$

When using a dataset where a positive class is not as common as a negative one - an unbalanced dataset; it is standard not to choose accuracy as the performance metric. This is because high accuracy can be achieved by predicting a negative class for every pixel. F1, JSI, and DCS do not compute with true negatives, consequently they are more suited for unbalanced datasets. JSI can be called Intersection over Union (IoU) of the ground truth mask and the prediction mask.

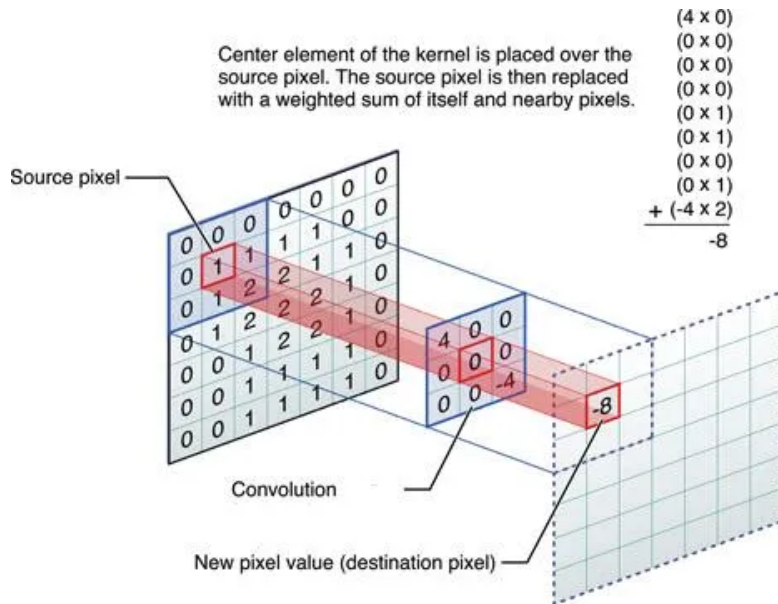
## 3.2 Convolution

2D image convolution, represented by the symbol  $*$ , is the fundamental of many image segmentation techniques. It is an operation between an image and a kernel, which is typically a smaller square matrix. The product of convolution between an image  $I$  and kernel  $k \in \mathbb{R}^{M \times N}$  is computed as follows:

$$f_{x,y} = I_{x,y} * k_{x,y} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{x-m,y-n} \cdot k_{m,n} \quad (3.11)$$

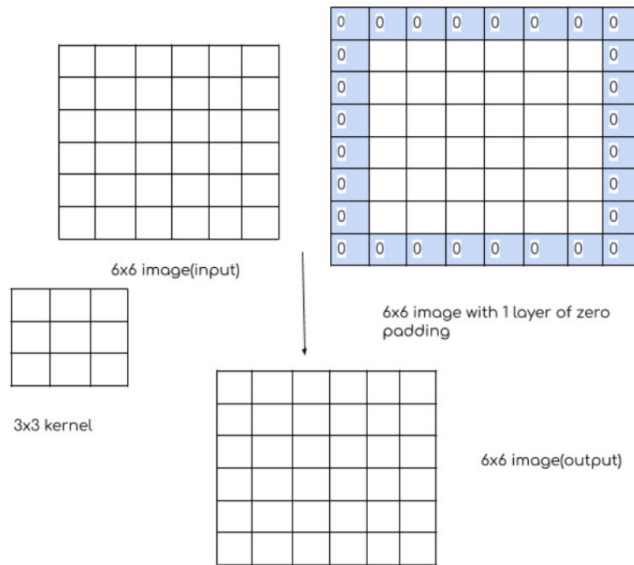
where  $f$  is the result of the convolution.

For simplicity, imagine a sliding kernel that multiplies image values in a neighborhood of a pixel, sums these values together, and moves to the next pixel. For a single pixel neighborhood, the procedure is illustrated in Figure 3.1. One undesired property of convolution is that it makes the output smaller in spatial dimensions than the input if the kernel size is larger than 1. Consider a case where  $x, y = 0$  and  $m, n = 1$  in the Equation 3.11, this will result in reaching for the value of  $I_{-1,-1}$  which is undefined. To handle this issue, padding was introduced. Padding implies adding rows and columns of chosen values around the original which causes the output image to have a preferred size. One of the most used padding is zero padding, meaning the added rows and columns all have values of zero, you can see zero padding visualized in Figure 3.2.



**Figure 3.1:** Visualization of convolution for one pixel. The original image is on the left, the kernel in the middle, resulting pixel on the right. At the top of the image, there is a detailed summation. Image source: [39].

Convolution with a specific kernel can yield desired results, there are many kernels used for image sharpening or edge detection, but probably the

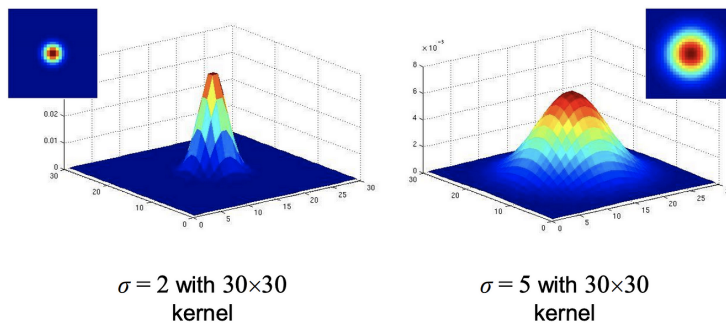


**Figure 3.2:** Visualization of zero padding. The  $6 \times 6$  original image is padded with one row of zeros at each side. The result of convolving the padded image with a  $3 \times 3$  kernel yields a  $6 \times 6$  result. Image source: [40]

most used kernel is the 2D Gaussian. Convolution with this kernel is called Gaussian smoothing or Gaussian blurring. The 2D Gaussian kernel, denoted by  $G(x, y; \sigma)$ , is a sampled two-dimensional probability distribution function defined as:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad (3.12)$$

where  $G(x, y; \sigma)$  is the value of the Gaussian kernel at position  $(x, y)$  with standard deviation  $\sigma$  controlling the spread of the distribution - the bigger the sigma, the smoother the result is. The 2D Gaussian is depicted in Figure 3.3



**Figure 3.3:** Visualization of two 2D Gaussian kernels. On the left 2D Gaussian with  $\sigma = 2$ , and on the right  $\sigma = 5$ . Image source: [41].

## 3.3 Image Histogram

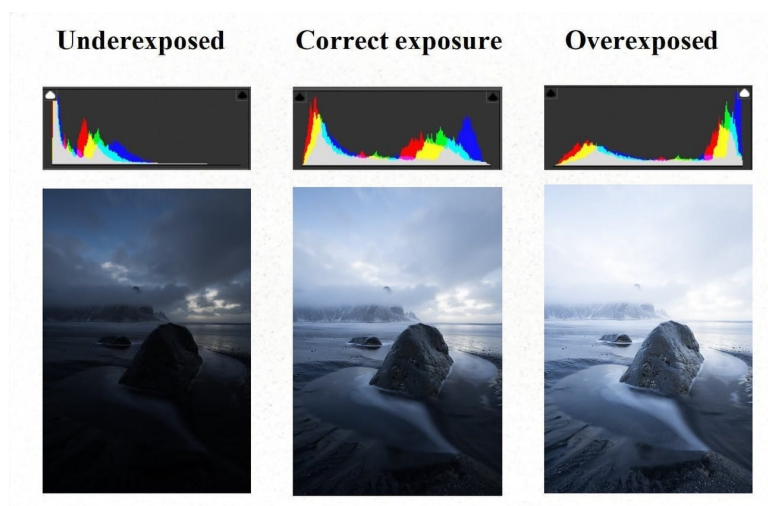
An image histogram is a discrete representation of pixel intensities in an image. Let  $I$  denote the image again, the histogram  $h \in \mathbb{R}^{256}$  for 256 image intensity levels is defined as:

$$h_i = \frac{n_i}{HW} \quad (3.13)$$

where  $n_i$  is the number of pixels in the image with intensity  $i$ . For color images, separate histograms are computed for each color channel.

### 3.3.1 Histogram Equalization

In general, we would like an image's histogram to be as spread as possible. Image with a histogram skewed too much to the left or right results in underexposed or overexposed images, respectively, see Figure 3.4 for visual examples.



**Figure 3.4:** Visualization of image histogram. The underexposed image has a histogram skewed to the left, and the overexposed to the right. The correctly exposed image has its histogram evenly spread. Image source: [42].

If we are provided an overexposed or underexposed image there is a technique for its correction - image equalization. The following lines describe how histogram equalization of an image  $I$  is performed.

---

#### Histogram Equalization Algorithm [43]:

*For each channel  $c$  repeat:*

- 1. Histogram Computation:**

- Compute histogram  $h$  from image  $I$  as described in Equation 3.13.

**2. Cumulative Distribution Function (cdf) computation:**

$$cdf(i) = \sum_{k=0}^i h_k. \quad (3.14)$$

**3. Normalize cdf:**

- compute normalized cdf  $cdf_n$  so that it ranges from 0 to 255:

$$cdf_i^n = 255 \cdot \frac{(cdf_i - \min(cdf))}{\max(cdf) - \min(cdf)} \quad (3.15)$$

**4. Round all values of  $cdf_n$  to integers.**

**5. Apply Transformation to the Original Image:**

- Obtain new pixel value for the equalized image  $I^{eq}$  given by

$$I_{i,j,c}^{eq} = cdf_{I_{i,j,c}}^n \quad (3.16)$$

where  $i, j$  are the space coordinates and  $c$  is the chosen channel.

## 3.4 Classical Image Segmentation Techniques

Before the era of neural networks, image segmentation was done by methods that consisted of simple steps. We will briefly explore some image segmentation techniques not utilizing neural networks. The following algorithms can be found implemented available online, most of them can be found in the OpenCV library for Python [44].

### 3.4.1 Thresholding

Thresholding is done by finding pixel threshold  $T$  that separates well foreground and background. Then mask  $M$  for a grayscale image  $I$  can be defined by equation:

$$M_{i,j} = \begin{cases} 0 & \text{if } I_{i,j} < T \\ 1 & \text{if } I_{i,j} \geq T \end{cases} \quad (3.17)$$

Images are often noisy, thus it is a good practice to apply a 2D Gaussian convolution before thresholding to ensure smoother segmentation boundaries.

### 3.4.2 Clustering

Clustering is used for estimating clusters corresponding to each class. Famous clustering techniques are K-means, expectation maximization (EM) algorithms, and SLIC Superpixels [45]. Pixels closest to cluster centers are labeled by a label assigned to the center, selected distance metrics can make clustering applicable to color images.

### 3.4.3 Edge Detection

When our desired segmentation masks are important edges of an image, we can choose from many edge detection techniques. The image edge is a neighborhood of pixels where the image function changes rapidly in a direction. Edge detection in its simplest form can be done by convolution of an edge-detecting kernel with the input image followed by thresholding. By choosing specific kernels we can detect edges of various sizes and directions. Examples of these kernels can be the Sobel windows.

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (3.18)$$

where  $S_x$  is the Sobel window for edge detection in the horizontal direction and  $S_y$  is the Sobel window for edge detection in the vertical direction.

Other more complex edge detection algorithms are the Marr-Hildreth Edge Detector, Canny Edge Detector, and Gabor filters. They all can be described in three basic steps:

#### 1. Preprocessing

- Image is blurred with a 2D Gaussian Kernel. This prevents the response from Step 2 from being too wiggly due to random noise.

#### 2. Convolution

- Blurred image is convolved with a specific kernel or more kernels.

#### 3. Postprocessing

- Postprocessing consists mainly of thresholding and non-maxima suppression.

Specifics of each of the three approaches mentioned above are summarized in Table 3.1 below. For more details about Canny Edge Detector and Marr-Hildreth Edge Detector see [45] and we refer you to [46] for more information about Gabor filters.

### 3.4.4 Graph Cut Based Segmentation

A graph  $G(E, V)$  is a structure made of vertices  $V$  and edges  $E$ . Let  $e = (u, v) \in E$  be one of the graph's edges and  $u, v \in V$  graphs vertices; the existence of such an edge  $e$  implies a connection from vertex  $u$  to vertex  $v$ . We can assign weights to each edge  $w(e) \in \mathbb{R}$  in a weighted graph. A special type of graph is undirected graphs, in undirected graphs  $(u, v) \in E$  implies  $(v, u) \in E$  and also  $w((u, v)) = w((v, u))$  [45].

We can transform  $H \times W(\times C)$  image  $I$  to an undirected weighted graph by following steps:

Algorithm Name	Preprocessing	Convolution Kernel	Postprocessing
Canny	Gaussian smoothing	Sobel operators	Gradient estimation Non-maxima suppression Hysteresis thresholding
Marr-Hildreth	Gaussian smoothing	Laplacian of Gaussian (LoG)	Zero crossing detection
Gabor	None	Gabor kernel(s)	Tresholding

**Table 3.1:** Comparison of Edge Detection Algorithms. Canny = Canny Edge Detector; Marr-Hildreth = Marr-Hildreth Edge Detector; Gabor = Gabor filters.

Transformation an of image to a graph [45, 47]:

1. Create graph vertices

- Create set of vertices  $V$

$$V = \{v_{i,j} \mid i \in [0, H - 1], j \in [0, W - 1]\} \quad (3.19)$$

where  $H, W$  are the height and width of image  $I$  respectively.

2. Create graph edges

- Start with empty edges set  $E$
- For each pair of different vertices  $v_{i,j}$  and  $v_{k,l} \in V$  do:

$$E = E \cup (v_{i,j}, v_{k,l}) \text{ if } (i, j) \in N_{(k,l)} \quad (3.20)$$

where  $i, j$  and  $k, l$  are the image coordinates;  $N_{(k,l)}$  is the neighbourhood of coordinates  $k, l$

3. Assign weights to edges

- for each edge  $e = (v_{i,j}, v_{k,l}) \in E$  compute:

$$w((v_{i,j}, v_{k,l})) \propto e^{(I_{i,j} - I_{k,l})^2} \quad (3.21)$$

4. Add source and sink vertices

- add two special vertices "source"  $s$  and "sink"  $t$  to  $V$
- for each vertex  $v \in V$  add an edges  $(s, v)$  and  $(t, v)$  to  $E$

- for each vertex  $v_{i,j}$  assign weights to  $(v_{i,j}, s)$  and  $(v_{i,j}, t)$  so that:

$$w((v_{i,j}, s)) \propto -\ln(\text{Pr}(I_{i,j} | \text{fore})) \quad (3.22)$$

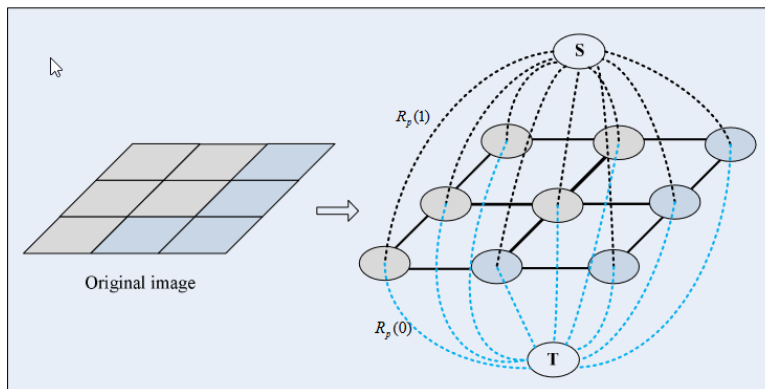
$$w((v_{i,j}, t)) \propto -\ln(\text{Pr}(I_{i,j} | \text{back})) \quad (3.23)$$

where  $\text{Pr}(I_{i,j} | \text{fore})$  denotes apriori probability that pixel of image  $I$  at coordinates  $i, j$  belongs to foreground; back = background; fore = foreground.

After creating a weighted undirected graph  $G(E, V)$  from an image accordingly, the graph's vertices are partitioned into two sets  $V_a$  and  $V_b$  so that  $V_a \cup V_b = V$  and  $V_b \cap V_a = \emptyset$ . The quality of such partition is measured by metric *cut* described in the following equation.

$$\text{cut}(V_a, V_b) = \sum_{u \in V_a, v \in V_b} w((u, v)) \quad (3.24)$$

Algorithms that find these two subsets with minimal cut are called graph cut algorithms, widely used examples are approaches from the Max-Flow Min-Cut algorithm family. Figure 3.5 depicts an image's visual representation as a graph. When the optimal partition is found, vertices connected to the "source" vertex are labeled as foreground and background otherwise. For multiclass segmentation, the same process as above can be applied to subgraphs of the original graph recursively [45, 47].



**Figure 3.5:** Visualization of image to graph transformation.  $S$  and  $T$  "source" and "sink" vertices respectively.  $R_p(1)$  = weight of edge  $(p, s)$ ;  $R_p(0)$  = weight of edge  $(p, t)$ . Image source: [47].

### 3.4.5 Line Segmentation

Lines and curves can also be important image features, for example, a road, a vessel, and wrinkles fall into this category.

### Hough Transform

The Hough method is a method for finding known patterns by transforming image points to parameter space. It is best explained as an example of



segmenting a line. A line can be parametrized by a non-commonly used equation:

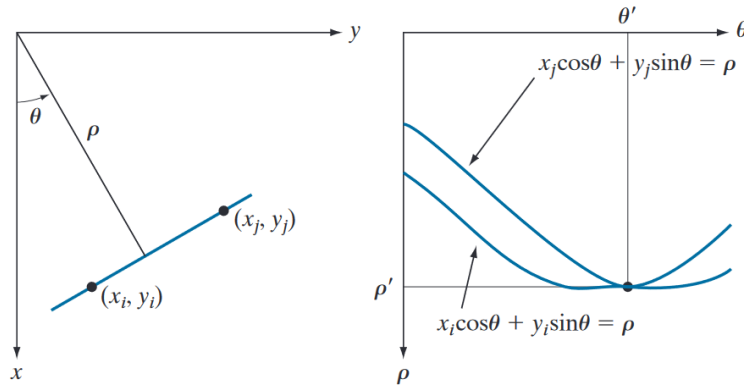
$$\rho(\theta, x, y) = x \cos(\theta) + y \sin(\theta) \quad (3.25)$$

where  $x, y$  are space coordinates;  $\theta$  is angle between the line and the x-axis;  $\rho$  is the distance of the line to the origin. For finding different templates other parameterizations need to be computed accordingly.

### Hough Transform Algorithm [45]:

1. **Detect edge points  $E$**  (see Subsection 3.4.3)
2. **Create parameters grid**
  - create vector  $axis_\theta$  is a uniformly spaced numbers in the interval  $[-\pi/2, \pi/2[$
  - create vector  $axis_\rho$  is a uniformly spaced axis in the interval  $[-D, D]$ , where  $D$  is the distance between image corners
  - create a grid  $G$  which is a  $|axis_\rho| \times |axis_\theta|$  matrix filled with zeros.
3. **Let points vote for parameters**
  - for edge point  $x, y$  in  $E$ 
    - for  $j$  in range( $|axis_\theta|$ )
      - $\theta = axis_\theta(j)$
      - compute  $\rho$  as described in Equation 3.25
      - find  $i$  so that  $axis_\rho(i) \leq \rho < axis_\rho(i + 1)$
      - $G_{i,j} = G_{i,j} + 1$
4. **Selecting most voted line**
  - Detected line is a line with  $\rho$  and  $\theta$  corresponding to coordinates with maximum value in the grid  $G$

Hough transform of two points is depicted in Figure 3.6



**Figure 3.6:** Visualisation of Hough transform for two points. In the left part of the image are visualized two points in cartesian space. In the right part, both points are transformed into sinusoids in parameter space. Image source: [45].

### Frangi Filter

Frangi filter is an algorithm designed for detecting blood vessels in both 2D and 3D images. It estimates the Hessian matrix for each pixel of the image and computes eigenvalues  $\lambda_1$  and  $\lambda_2$ . If  $\lambda_2 \gg \lambda_1$ , it is suggested that there is a direction in which image intensity changes, and in the perpendicular direction there is no change, which indicates a line. If  $\lambda_2 \approx \lambda_1 \gg 0$  it signals a blob-like structure there [48].

---

#### Frangi Filter Algorithm for a grayscale image [48]:

1. Choose a sigma  $\sigma$

- The sigma size determines the width of detected curves.

2. Gaussian second derivatives computation

- Precompute 2D Gaussian  $G(x, y; \sigma)$  second derivatives kernels  $G_{xx}$ ,  $G_{xy}$  and  $G_{yy}$ . Subscript  $xx$  denotes derivating twice w. r. t.  $x$ , similarly for other subscripts. 2D Gaussian is defined in Equation 3.12.

3. Hessian Matrix Computation:

- Coompute  $I_{xx} = I * G_{xx}$ , similarly for  $I_{xy}$  and  $I_{yy}$ .
- The Hessian matrix for pixel  $i, j$  is defined as:

$$H_{i,j} = \begin{bmatrix} I_{xx;i,j} & I_{xy;i,j} \\ I_{xy;i,j} & I_{yy;i,j} \end{bmatrix} \quad (3.26)$$

4. Eigenvalue Decomposition:

- Compute the eigenvalues  $\lambda_{1;i,j}$  and  $\lambda_{2;i,j}$  of the Hessian matrix  $H_{i,j}$  at each pixel  $i, j$ .

**5. Blobness and Vesselness Measure Calculation:**

- define  $R, S$  and vesselness  $V$  as

$$R = \frac{\lambda_1}{\lambda_2}, S = \sqrt{\lambda_1^2 + \lambda_2^2} \quad (3.27)$$

$$V = e^{-\frac{R^2}{2\beta^2}} (1 - e^{-\frac{S^2}{2c^2}}) \quad (3.28)$$

where  $\beta$  and  $c$  are tunable parameters.

**6. Choosing bright or dark vessels:**

- If we are looking for dark vessels on a bright background, we set  $v = 0$  for pixels where  $\lambda_2 > 0$ . On the contrary, when bright vessels are sought we set  $V = 0$  for pixels where  $\lambda_2 < 0$ .

**7. Repeat steps 1. - 6. for different sigmas and find the maximum among the responses  $V$ .**

## 3.5 Image Segmentation with Convolutional Neural Networks

The recent uprise of neural networks also affected image segmentation and these approaches tend to surpass classical image segmentation techniques. A convolutional neural network (CNN) can be described as a sequence of layers applied to the original image. These layers contain tunable parameters  $\theta$ . Applying a segmentation neural network  $NN$  for to an input image  $x_{in} \in \mathbb{R}^{H_{x;in} \times W_{x;in} \times C_{x;in}}$ , results in  $y_{out} = NN(x_{in}; \theta)$ , where  $y_{out} \in \mathbb{R}^{H_{y;out} \times W_{y;out} \times C_{y;out}}$ .  $H_{x;in}, W_{x;in}, C_{x;in}$  are the input dimensions;  $H_{y;out}, W_{y;out}, C_{y;out}$  are the output dimensions. It is a good practice to set  $H_{in} = H_{out}$  and  $W_{in} = W_{out}$ . For binary image segmentation  $C_{out} \in \{1, 2\}$ , for multiclass segmentation  $C_{out} = n_{classes}$ , where  $n_{classes}$  number of classes for desired segmentation task. NN is trained to generate for each input  $I_{in}$  predictions  $y_{out}$  closest to ground truth segmentation labels  $y_t$ .

### 3.5.1 Layers of CNNs

Layers of CNNs can be described as operations to an input  $x \in \mathbb{R}^{H_x, W_x, C_x}$  producing output  $y \in \mathbb{R}^{H_y, W_y, C_y}$ .

## ■ Activation Functions

Activation function  $f_{act}$  does not change the input dimensions meaning  $H_x = H_y, W_x = W_y$ , and  $C_x = C_y$ . It is applied to each spacial element of  $x$  separately, meaning  $y_{i,j,c} = f_{act}(x_{i,j,c})$ , where  $i, j, c$  are row, column and channel respectively.

### ■ Rectified Linear Unit (ReLU)

ReLU is the most often used activation layer [49] for its fast computation. It is defined as [50]:

$$y_{i,j,c} = ReLU(x_{i,j,c}) = \begin{cases} x_{i,j,c} & \text{if } x_{i,j,c} \geq 0 \\ 0 & \text{if } x_{i,j,c} < 0. \end{cases} \quad (3.29)$$

ReLU has several modifications that address zero outputs of ReLU for every negative number, namely Leaky ReLU and Exponential Linear Unit (ELU), their definitions are below [50].

### ■ Leaky ReLU

$$y_{i,j,c} = LeakyReLU(x_{i,j,c}) = \begin{cases} x_{i,j,c} & \text{if } x_{i,j,c} \geq 0 \\ kx_{i,j,c} & \text{if } x_{i,j,c} < 0, x_{i,j,c}, k \in \mathbb{R} \end{cases} \quad (3.30)$$

$k$  is chosen to be a small positive integer, e.g. 0.1.

### ■ ELU

$$y_{i,j,c} = ELU(a) = \begin{cases} x_{i,j,c} & \text{if } x_{i,j,c} \geq 0 \\ k(e^{x_{i,j,c}} - 1) & \text{if } x_{i,j,c} < 0, k \in \mathbb{R} \end{cases} \quad (3.31)$$

### ■ Sigmoid Function

The sigmoid function is denoted by  $\sigma()$  and is computed as follows [50]:

$$y_{i,j,c} = \sigma(x_{i,j,c}) = \frac{1}{1 + e^{-x_{i,j,c}}}, a \in \mathbb{R} \quad (3.32)$$

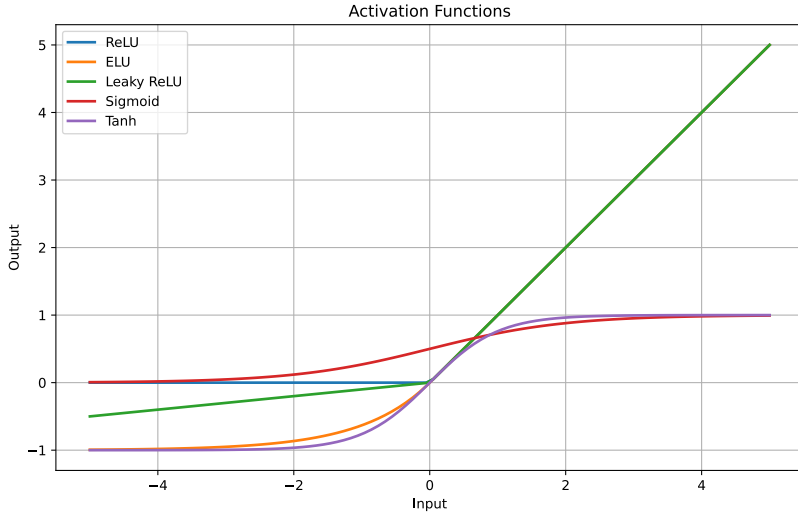
The output of  $\sigma()$  is a real number between 0 and 1. For this reason, the sigmoid function is often used in the last layer of a neural network for binary classification. Its output can represent the probability of a pixel belonging to the foreground.

### ■ Hyperbolic Tangent

Hyperbolic Tangent ( $\tanh$ ) can be also a choice for an activation function, it is defined as [50]:

$$y_{i,j,c} = \tanh(x_{i,j,c}) = 2\sigma(2x_{i,j,c}) - 1 \quad (3.33)$$

where  $\sigma()$  is the sigmoid function. Tanh and all previously mentioned activation functions are visualized in Figure 3.7.



**Figure 3.7:** Visualization of activation functions. ReLU and ELU are covered by Leaky ReLU for positive inputs because their output is the same.  $k$  is set to 0.1 for Leaky ReLU and to 1 for ELU.

■ **Softmax**

Softmax is most commonly used as the last layer for multiclass classification. It is described as:

$$y_{i,j,c} = softmax(x_{i,j,c}) = \frac{e^{x_{i,j,c}}}{\sum_{k=0}^{C_x-1} e^{x_{i,j,k}}} \quad (3.34)$$

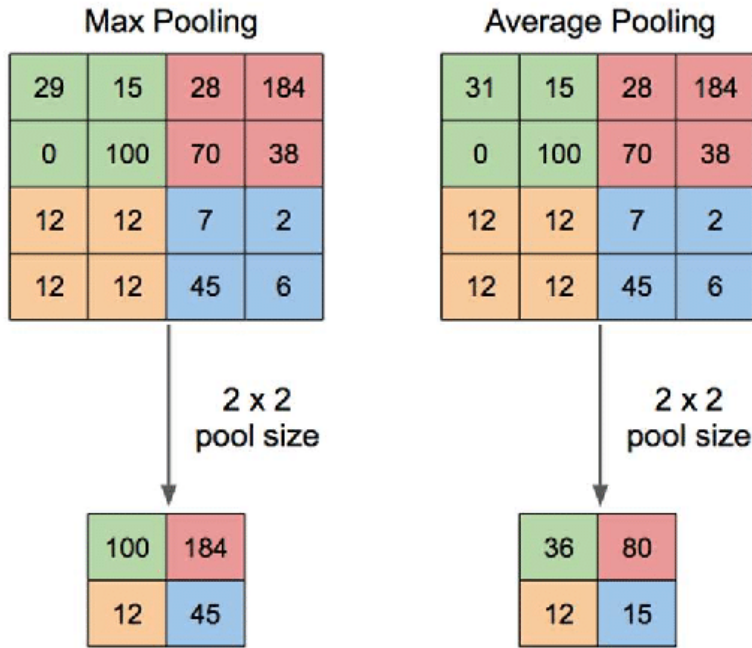
Results of softmax layers have nice properties, they sum up to 1 alongside the channel dimension and they fall within the range of 0 to 1. This is why  $y_{i,j,c}$  can be used as the posterior probability of a spatial  $(i, j)$  coordinate belonging to a class  $c$  [50, 51].

■ **Pooling Layers**

Pooling layers, also known as downsampling layers, reduce the size of the input. The most used downsampling factor is 2, meaning  $H_y = H_x/2$ ,  $W_y = W_x/2$ , and the number of channels remains the same. In average pooling, the input within  $2 \times 2$  non-overlapping windows is averaged for each channel, in max-pooling the maximum in the window is calculated instead [52]. Both max pooling and average pooling are depicted in Figure 3.8.

■ **Convolutional Layers**

Convolutional layers are the backbone of CNNs, at default, a convolutional layer has  $C_x \cdot C_y$  kernels. Each kernel performs convolution independently with its corresponding input channel and produces an output for the corresponding



**Figure 3.8:** Visualisation of max pooling is on the left and average pooling on the right. As you can see, the input gets downsampled by 2, input is  $4 \times 4$  and output is  $2 \times 2$ . Image source: [52]

output channel. They are summed together if multiple results are generated for a single output channel (when  $C_x > 1$ ). To preserve the original height and width, padding has to be applied as described in Section 3.2. For convolutional layers, we also define parameter stride. Stride  $s$  a positive integer, which indicates how many pixels the convolution kernel moves when transitioning to the next pixel. We can also define dilation, denoted by  $d$ , which essentially controls the spacing between the elements of the convolution kernel [51, 53], as illustrated in Figure 3.9.

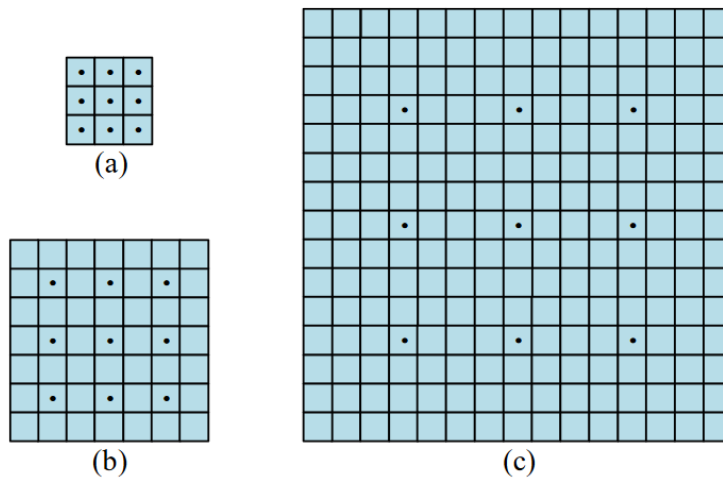
The output height  $H_y$  can then be computed as follows [54]:

$$H_y = \frac{H_x + 2 \cdot p - d \cdot (H_k - 1) - 1}{s} + 1, \quad (3.35)$$

where  $p$  denotes number of rows used for padding and  $H_k$  kernel height.  $W_y$  can be computed similarly. While it is possible to use different dilation, stride, padding, and kernel sizes for the height and width dimensions, the vast majority of networks use the same values for both dimensions.

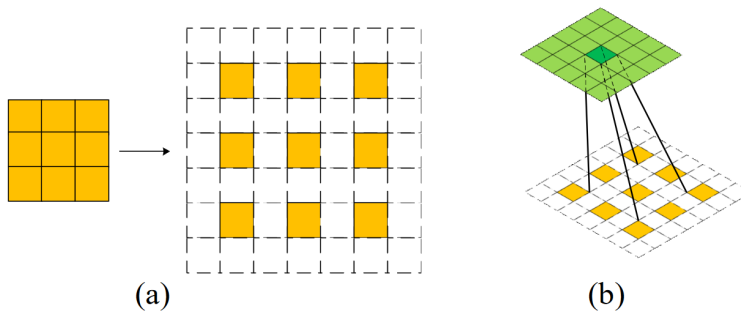
### ■ Fractionally-Strided Convolutional Layer

A special type of convolutional layer is the fractionally-strided convolution layer. The main contrast to classic convolution is that padding is applied by inserting a row and column of zeros between each row and column of the



**Figure 3.9:** Comparison between general convolution kernel and dilated convolution kernel. Black dots mark the neighborhood of a computed pixel. (a) A general  $3 \times 3$  convolution kernel (b) A 2-dilated  $3 \times 3$  convolution kernel (c) A 4-dilated  $3 \times 3$  convolution kernel. Image and caption source: [52]

original image, resulting in output two times larger, that is why this layer is also called the upsampling layer [53]. The fractionally-strided convolution is depicted in Figure 3.10.



**Figure 3.10:** Visualisation of fractionally-strided convolution. (a) the original array is padded with one row and column between each two rows and columns creating a  $7 \times 7$  array (b) convolution of the padded array with  $3 \times 3$  kernel resulting in  $5 \times 5$  output. Image source: [53].

### 3.5.2 Batch Normalization

Neural networks process images in small groups - batches, number of images in these batches is called batch size  $B$ . It has been shown, that normalizing a batch so that its average is 0 and standard deviation is 1 is beneficial to the network's performance. This calculation is done for each batch processed by a neural network and it is referred to as batch normalization [55].

### 3.5.3 Losses

The objective function we try to minimize in the segmentation task with neural networks is called the loss function. Among the most well-known loss functions are cross-entropy loss and dice loss.

#### Crossentropy Loss

Cross entropy loss  $l_{i,j,c}$  for one output element  $y_{out;i,j,c}$  and target  $y_{t;i,j,c}$  is computed as:

$$l_{CE}(y_{out;i,j,c}, y_{t;i,j,c}) = -w_c \log(\text{softmax}(y_{out;i,j,c})) \cdot y_{t;i,j,c}, \quad (3.36)$$

where  $w_c$  are weights assigned to each class. For one image, the loss is then:

$$L(y_{out}, y_t) = \sum_{c=0}^{C-1} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} l(y_{out;i,j,c}, y_{t;i,j,c}) \quad (3.37)$$

For binary segmentation, the output can have only one channel, in this case, we define binary cross-entropy loss as:

$$l_{BCE}(y_{out;i,j}, y_{t;i,j}) = -(y_{t;i,j} \cdot \log(\sigma(y_{out;i,j})) + (1 - y_{t;i,j}) \log(1 - \sigma(y_{out;i,j}))) \quad (3.38)$$

#### Dice Loss

Cross-entropy loss can yield very good segmentation results when the numbers of pixels for each class are approximately the same. If this is not the case, weights for each class can be assigned to help the imbalance, but a more stable solution is to use dice loss [56]. Dice loss for one channel is defined as [56]:

$$l_{dice}(y_{out}, y_t) = 1 - \frac{2 \cdot \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} y_{t;i,j} y_{out;i,j} + \gamma}{\sum_{i=0}^{H-1} \sum_{j=0}^{W-1} y_{t;i,j}^2 + \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} y_{out;i,j}^2 + \gamma}, \quad (3.39)$$

where  $\gamma$  is a parameter chosen for faster convergence. For multiclass segmentation,  $l_{dice}$  is applied to each channel separately and the background channel can be omitted.

### 3.5.4 Training and Validation

The neural network training process utilizes backpropagation, a form of gradient descent known as stochastic gradient descent (SGD). Stochastic gradient descent is specific in not computing derivation of the loss function for all data at once but for small batches of images. CNNs are trained by upgrading parameters in their layers, from the layers we have mentioned, the only tunable parameters (also called weights) are the values of kernels in convolution layers. The learning process is broken into epochs, in one epoch usually all data are fed to the network, and its parameters are adjusted. The number of epochs is one of the hyper-parameters that needs to be chosen by the user.



## ■ Backpropagation

As we already defined input to our network is denoted by  $x_{in}$  and the output by  $y_{out} = NN(x_{in}; \theta)$ , where  $\theta$  are parameters of the network. If we have ground truth labels  $y_t$  we call this supervised learning. Backpropagation is a process where the derivation of the loss function  $l()$  is computed w. r. t.  $\theta$ . Neural network output  $NN(x_{in}; \theta)$  can be expressed as sequence of  $N$  layers:

$$NN(x_{in}; \theta) = L_N(L_{N-1}(\dots; \theta_{N-1}); \theta_N) \quad (3.40)$$

and loss  $l()$  then expressed as

$$l(NN(x_{in}, \theta), y) = l(L_N(L_{N-1}(\dots; \theta_{N-1}); \theta_N), y_t),$$

$\theta_i$  are the parameters of the  $i$ th layer  $L_i$ .

Let us set a temporary goal which is to compute the derivation of  $l()$  w.r.t.  $\theta_{N-1}$ . The derivation can be computed by applying the chain rule:

$$\frac{\partial l(y_{out}, y_t)}{\partial \theta_{N-1}} = \frac{\partial l(y_{out}, y_t)}{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))} \cdot \frac{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))}{\partial \theta_{N-1}} \quad (3.41)$$

and

$$\frac{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))}{\partial \theta_{N-1}} = \frac{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))}{\partial L_{N-1}(\dots; \theta_{N-1})} \cdot \frac{\partial L_{N-1}(\dots; \theta_{N-1})}{\partial \theta_{N-1}}. \quad (3.42)$$

Together it yields

$$\frac{\partial l(y_{out}, y_t)}{\partial \theta_{N-1}} = \frac{\partial l(y_{out}, y_t)}{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))} \cdot \frac{\partial L_N(L_{N-1}(\dots; \theta_{N-1}))}{\partial L_{N-1}(\dots; \theta_{N-1})} \cdot \frac{\partial L_{N-1}(\dots; \theta_{N-1})}{\partial \theta_{N-1}}. \quad (3.43)$$

It can be generalized to:

$$\frac{\partial l(y_{out}, y_t)}{\partial \theta_i} = \frac{\partial l(y_{out}, y_t)}{\partial L_N} \cdot \prod_{j=i+1}^N \frac{\partial L_j}{\partial L_{j-1}} \cdot \frac{\partial L_i}{\partial \theta_i} \quad (3.44)$$

Thus in practice, the derivation of parameters in all layers is computed iteratively by applying the chain rule.

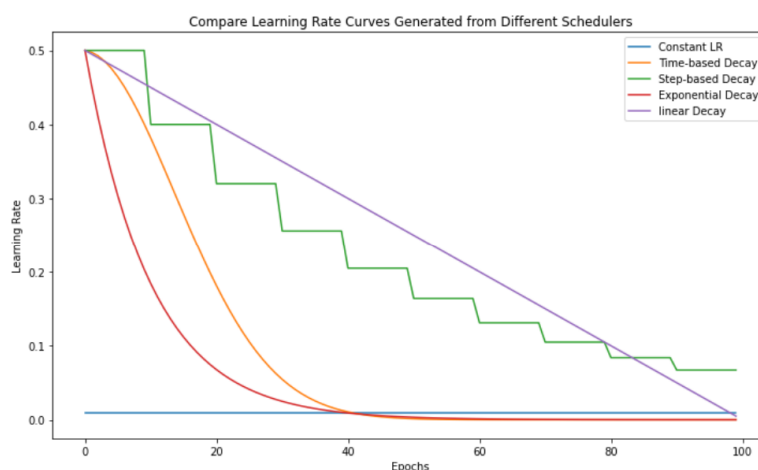
## ■ Learning Rate

Learning rate  $lr$  is a hyper-parameter, that determines the magnitude of weight adjustments during the training process. Selecting an appropriate learning rate is not an easy task. It can be picked heuristically by testing different learning rates and seeing how fast the loss function results decrease.

During the training phase, when the loss function approaches some local minimum the learning rate can be too high and it skips the local minimum, this is why sometimes it can be beneficial to lower the learning rate during the training process. It is achieved by using a learning rate scheduler, which predetermines how fast the learning rate drops in each epoch. For example,

an exponential decay learning rate scheduler multiplies the learning rate each epoch by constant  $\gamma$ . Exponential decay learning rate scheduler and other broadly used schedulers are plotted in Figure 3.11. There are numerous other strategies and approaches to selecting a learning rate suitable for each task [57].

Optimizers are algorithms that update the weights of the neural network during training. A well-known optimizer (and arguably the most used one) is the Adam optimizer (abbreviated from adaptive momentum estimation) [58], but plain stochastic gradient descent can be selected, too.



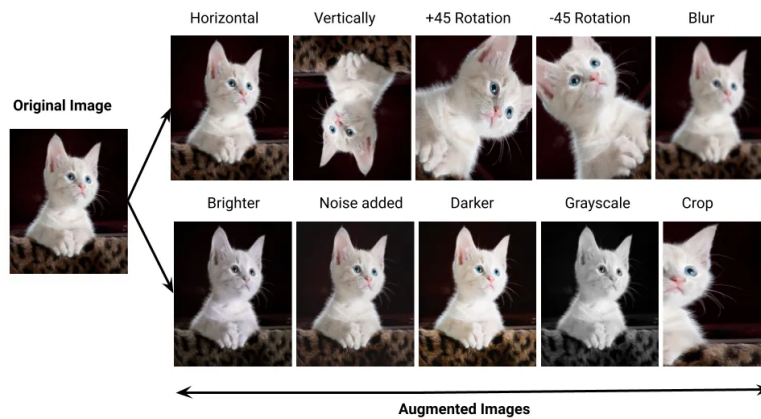
**Figure 3.11:** Visualized different learning rate schedulers. Image source: [59].

## Validation

During the training, the neural network is learned only from the training data, when presented with new data, its behavior may be unpredictable. If the network fails to provide desired results for other data than the training data, we call this state overfitting. During the training, we usually set a portion of our collected data aside - those are called testing data and we select a metric that evaluates how well our network performs. During the training, we can feed the testing data to the network without backpropagating the results, and if the metric consistently worsens, it is best to shut off the training process completely.

## Data Augmentation

Data augmentation is a method to create more training data without costly labeling procedures and data acquirement, which may help with overfitting. Data augmentation techniques for images may include flipping, rotation, color jitter, contrast change, noise addition, sharpening, blurring, cropping, resizing and many more [60]. Some of the basic augmentations applied to a single image are depicted in Figure 3.12. It is of course important to augment the ground truth labels in the same way as the image.



**Figure 3.12:** Basic image augmentations applied to an image of a cat. Image source: [61].

## ■ Training process

The training process can be described in a few basic steps:

---

### Steps of Training Process

#### 1. Prepare data and network

- load training data and testing data. Your data should be stored in RAM for quick access during training.
- load your neural network  $NN$  model to RAM or GPU if possible
- initialize optimizer, loss function  $l()$ , and learning rate scheduler

#### 2. Choose hyperparameters

- Set hyperparameters like learning rate, batch size, number of epochs

#### 3. Training loop

- for epoch in range(number of epochs)
  - for batch =  $(x_{in}, y_t)$  in training data
    - a. augment the batch
    - b. move batch to GPU if possible
    - c. compute  $y_{out} = NN(x_{in})$
    - d. compute  $l(y_{out}, y_t)$
    - e. backpropagate loss function with the optimizer to update parameters
  - update learning rate in the optimizer via its scheduler
  - validate network on the test data (not necessarily every epoch)
    - if validation is significantly decreasing: terminate training

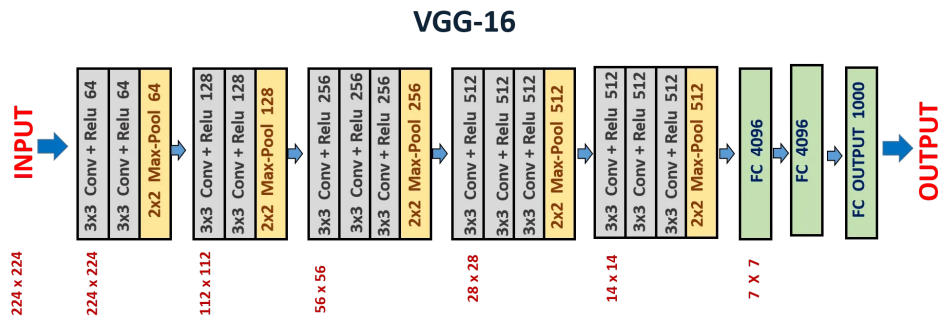
#### 4. Save the best-performing model (on the testing dataset) to a file

### 3.5.5 CNN Architectures for Image Segmentation

Neural network architecture is a given set of layers, their specific sizes, used activation functions, and ordering. A NN architecture is best visualized by a schematic as in Figure 3.13. These schematics contain the type of each layer, their parameters, the size of their corresponding input, and the direction of forward propagation.

#### Fully Convolutional Neural Networks

A fully convolutional neural network (FCNN) consists mainly of convolution layers and activation functions. One of the FCNN networks used for image classification is the VGG net. It contains many convolutional layers followed by a few fully connected layers - these layers apply matrix multiplication so that the output is a single vector of length  $n_{classes}$ . A schematic of the VGG structure is depicted in Figure 3.13.

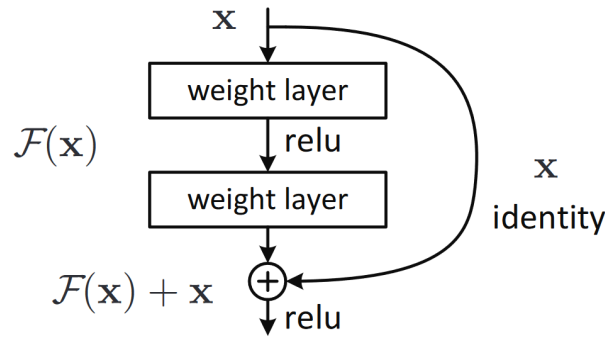


**Figure 3.13:** Schematic of the VGG-16 net. Beneath each layer, there is a size of  $H \times W$  in the layer. Image source: [62].

#### ResNet

When using too many convolution layers, the gradient of the parameters in the first layers can be small, it is called the vanishing gradient problem. This issue can be solved by adding an identity mapping of input to each layer to its output. This layer is called the residual block and it is depicted in Figure 3.14. ResNet is a very famous architecture employed for image classification consisting mainly of residual blocks and it is visualized in Figure 3.15.

We previously mentioned that both ResNet and FCNs were used for image classification, not segmentation. If we remove the last fully connected layers and replace them with upsampling layers, we can achieve pixel-wise segmentation. DeepLab v1, v2, v3, and v3+ are notable examples of architectures for image segmentation with many convolution layers, they use convolution layers with different dilatations [65]. Deeplab v3 architecture is illustrated in Figure 3.16.



**Figure 3.14:** Visualized residual block image from the original paper. Image source: [63].

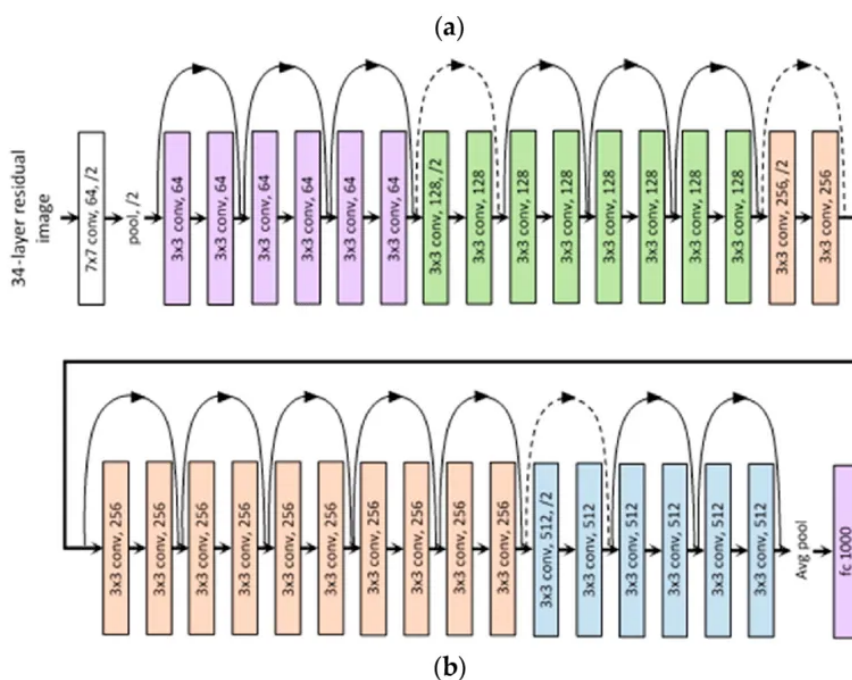
## ■ UNet

A famous neural network for biomedical image segmentation is called UNet for its U-shape structure with multiple levels. In the downsampling part, called the encoder, each level has two convolution layers and a downsampling layer. Before downsampling the result on each level is stored. When the last level is reached, the decoder part begins with iterative upsampling. With each upsampling, the signal gets one level up and it is concatenated with the stored result from the decoder part. The concatenation is iteratively again convolved by two convolution layers and upsampled until the initial level is reached. Then there is a last convolution layer where output has  $n_{classes}$  channels [67]. Visualization may be needed to fully understand the process, the scheme of UNet is provided in Figure 3.17.

Many variations and improvements of UNet were developed over time e.g. using more convolutions in each layer or adding residual units. UNet++ network modifies the concatenation inside the UNet [68]. UNet-like networks are particularly well suited for biomedical data segmentation because they incorporate both local information from the upper levels and global information from lower levels [65].

## ■ 3.6 Segmentation of Wrinkles

Wrinkles are folds of facial skin and are one of the aging features. Different types of wrinkles can arise from (i) loss of skin elasticity, (ii) gravity, (iii) repeated facial expressions, or (iv) inherited factors [69]. When showing facial expressions or when talking, different types of wrinkles can emerge or change their position. Prior knowledge can help with wrinkle detection; for example, wrinkles are typically horizontal lines on the forehead. We can name wrinkles according to their localization, as depicted in Figure 3.18.



**Figure 3.15:** Visualized schematic of ResNet. It consists primarily of residual blocks. Image source: [64].

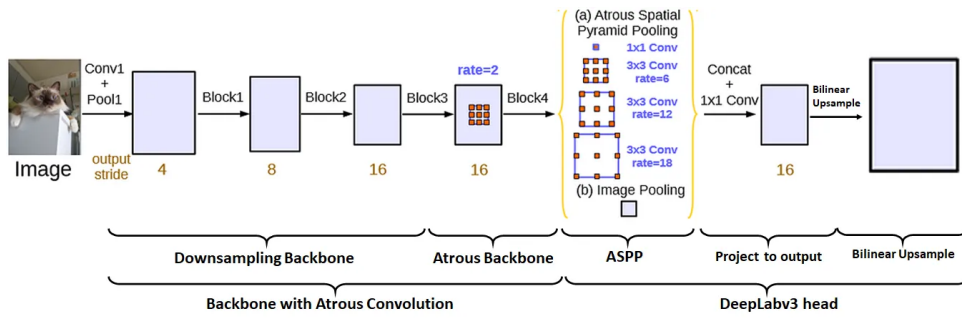
### 3.6.1 Facial and Wrinkle Datasets

Datasets play a critical role in neural network training because a network usually needs a large amount of annotated data. To our knowledge, no publicly available dataset for wrinkle segmentation has been published, or the quality of published datasets is deemed inadequate.

On the contrary, there are many facial datasets without annotated wrinkles. These datasets may contain images of people in various contexts, such as "in the wild", with multiple individuals in one image, or individuals in multiple facial expressions. Additionally, the resolution of images in these datasets can vary. Because of the numerous datasets, we will not discuss them individually, but we refer the reader to websites that list the best ones, allowing them to select a dataset fit for their purpose [70–72].

### 3.6.2 Prior Work in Wrinkle Segmentation

Wrinkles ordinarily manifest as dark curves on a human face, although light conditions can rarely cause them to be lighter than their surrounding pixels. This specific dark-light-dark (light-dark-light) transition sets them apart from image edges, which are just light-dark or dark-light transitions.



**Figure 3.16:** Visualization of Deeplab v3. Upsampling is done by Atrous Spatial Pyramid Pooling, which applies fractionally strided convolution of different sizes and sums these responses together. [66].

### Basic Image Techniques for Wrinkle Segmentation

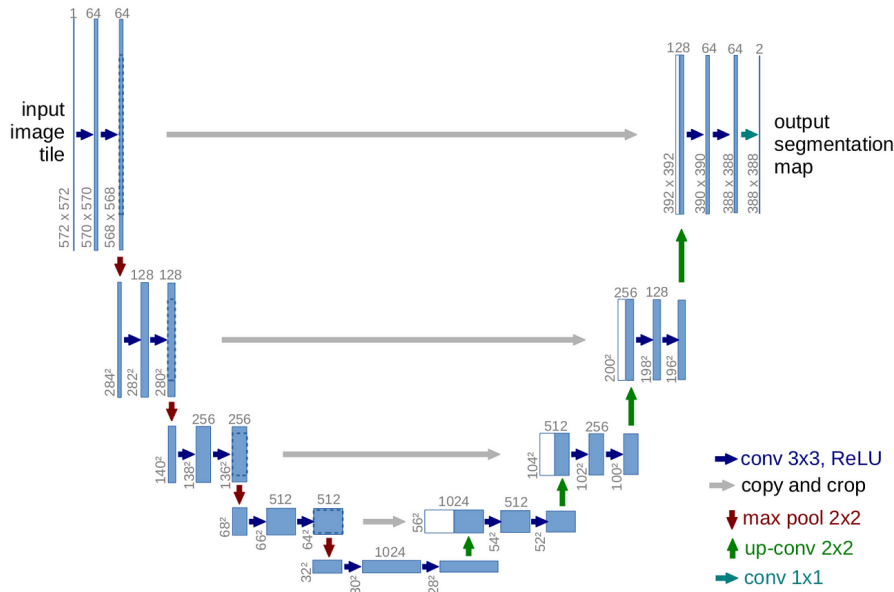
Two surveys for wrinkle detection algorithms have been published, one in 2016 [73] and the other in 2021 [74]. Both surveys agree that basic image methods for edge detection (Canny Edge Detector, Marr-Hildreth Edge Detector, Gabor filters, Sobel windows) fail to detect wrinkles or produce too many false positives. Other approaches are harder to evaluate because they, for example, detect points along the wrinkle, and/or the authors did not provide a quantitative measurement for their performance.

One algorithm that stands out the most in both reviews is called Hybrid Hessian Filtering (HHF). HHF is a slight extension of Frangi filter [48], which we described in 3.4.5. It shows that vessels in the eye retina are in a way similar to wrinkles. The main advantages of this approach are (i) simplicity, (ii) fast computation, and (iii) being publicly available. The authors reported an average JSI of 75.57 % for the Bosphorus dataset's [75] forehead segments [76]. This method does not generalize properly to the whole face where the direction of wrinkles changes [77].

### CNNs for Wrinkle Segmentation

There are more recent approaches that make use of CNNs for wrinkle detection. In a study from 2021, authors used the FACES dataset [78] to evaluate their detection approach. They used U-NET structured CNN with dice loss function and reported a mean accuracy of 98.9 %. Accuracy was computed from the JSI of each image, where successful detection meant 80 % or higher JSI value. However, the authors only segmented nasolabial folds [79].

A newer article from 2022 uses a semiautomatic labeling strategy; a combination of human-annotated labels and a computed texture map from an image function. A UNet++ architecture with dice loss is trained to detect wrinkles. Another alternation from classical UNet was computing loss for each level of the UNet++ on resized ground truth labels, which is known as deep supervision. They reported 47 % and 44 % JSI in the forehead and the eye area, respectively. They used their original dataset acquired by skin analysis



**Figure 3.17:** Visualization of UNet. On the left of each layer is the size  $H \times W$ , and above it is the number of channels  $C$ . With each layer down the number of channels doubles and the size of the output is roughly halved. Note that the convolution in the original UNet does not use padding so the output is smaller than the input. Image source: [67].

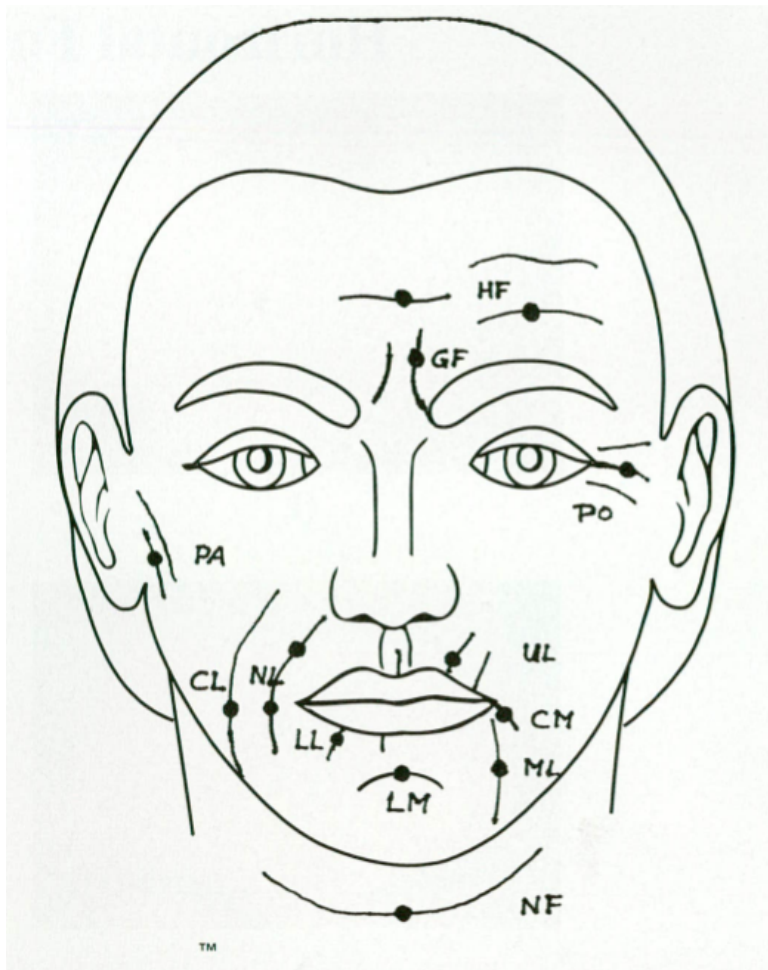
device and the images were then annotated by three human labelers [77].

Another study utilized UNet++ for segmenting wrinkles for subsequent wrinkle removal, however, even when reporting a high JSI, their segmentation results did not look plausible [80].

### Visual Transformers for Wrinkle Segmentation

A very fresh contribution to wrinkle detection by neural networks is from 2024. The authors used a new dominant type of network based on visual transformers (ViT) with cross-entropy loss combined with auxiliary losses and named their model Striped WriNet [81]. The authors reported a very high JSI of 65.54 % for wrinkles on the whole face for their public data set and 46.51 % for their private dataset. Wrinkles were annotated by professional dermatologists and for the facial dataset, they used the Flickr-Faces HQ dataset [82] combined with the CelebA-HQ dataset [83]. It is important to mention that authors deployed multiple architectures for wrinkle detection and all the other networks showcased lower, but fairly close JSI to Striped WriNet architecture, ranging from 63.01 % to 64.92 % on the public dataset, and 43.65 % to 45.63 % on the private dataset citewrinet. The architecture of Striped WriNet is available online on the authors' GitHub page [🔗](#).





**Figure 3.18:** Wrinkles localization. HF, horizontal forehead lines; GF, glabellar frown lines; PO, periorbital lines; PA, preauricular lines; CL, cheek lines; NL, nasolabial folds; UL, upper radial lip lines; LL, lower radial lip lines; CM, corner of the mouth lines; ML, marionette lines; LM, labiomenal crease; NF, horizontal neck folds. Image and caption source: [69].

### ■ 3.6.3 Wrinkle Segmentation Summary

The main obstacle to wrinkle segmentation is a missing publicly available dataset of human-annotated wrinkles [74,81]. The reason for this is that there can be a variety of wrinkles on a human face and it is very time-consuming to annotate them all correctly. Even though all the articles previously mentioned show promising results no authors made their models or datasets public. They evaluated models on their private datasets, and despite the authors of Striped WriNet naming one of their datasets "public", it was unfortunately unavailable. A good dataset would help with training neural networks and could be used to validate algorithms. Out of classical approaches, Hybrid Hessian Filtering provides the best results, but more recent algorithms using CNNs and Visual Transformers seem to surpass it.




**Part II**

**Research Project**

## Chapter 4

### Methodology

We decided to segment wrinkles in a video dataset of Parkinson’s disease patients and healthy controls by neural networks since they show the best performance. For this purpose, we trained multiple architectures and appointed the most fitted for video wrinkle segmentation. With a public wrinkle dataset not being available, we assembled our own containing a high amount of 674 annotated images for training and validation. With the selected trained network we segmented wrinkles in the videos and obtained wrinkle characteristics which were statistically tested. Characteristics that emerged significantly different between Parkinson’s disease patients and healthy controls were then used to fit a logistic regression model and evaluate its performance.

All programming for this diploma thesis was done in Python language with free-to-use libraries, including NumPy, PyTorch, OpenCV, MONAI, SciPy, MediaPipe, etc. Additionally, we also used solutions published to GitHub  by other authors with adherence to relevant licenses. We published our methods for video analysis and wrinkle dataset onto a public [GitHub !\[\]\(a75296508989caaa77a08d26cfccd4e5\_img.jpg\)](#) repository.

#### 4.1 Datasets

Two datasets were used for this diploma thesis, the first dataset contains videos of Parkinson’s disease patients and healthy controls, and the second dataset consists of images with labeled wrinkles.

##### 4.1.1 Dataset of Parkinson’s Disease Patients and Healthy Controls

To study the changes in the early stages of Parkinson’s disease, videos of 100 de-novo-diagnosed Parkinson’s disease patients (PN) were recorded. This group was matched in terms of age and gender with a healthy controls (HC) group and the dataset totals 200 one-minute-long video segments. Table 4.1 provides basic clinical data about the study participants. PN were recorded between 2015 and 2020 and diagnosed by MDS-UPDRS standards [9]. The exclusion criteria from the study were:

Name	PN
Count (-)	100
Males/Females	61/39
Age mean (y)	61.2
Age standard deviation (y)	11.9

**Table 4.1:** Clinical data of PN included in the PNHC dataset

- previously used treatment for PD
- history of hypomimia-inducing disease in the family tree
- depression
- therapy for hypomimia or speech improvement

The study received approval from the ethics committee of General University Hospital in Prague, Czechia, and has been performed following the ethical principles laid down by the Declaration of Helsinki.

Videos were recorded by a digital camera Panasonic Handycam HDR-CX410 placed approximately 1 meter in front of the examined study participant. The resolution of the videos is  $1440 \times 1080$  pixels and the sampling frequency is 25 frames per second. All the videos contain monologues on an arbitrary topic and they were provided to the author of this thesis by the Signal Analysis, Modelling, and Interpretation (SAMI) team in .mp4 format. For future distinctness, we will call this dataset of videos the PNHC dataset.

### ■ 4.1.2 Wrinkle Dataset

As previously mentioned, no publicly available datasets for wrinkle annotation that adhere well to wrinkles are available. Thus, if we wanted to obtain a human-labeled dataset, we had to assemble our own. For this purpose, we needed a free-to-use face dataset. Among many choices, we picked the Flickr Faces HQ (FFHQ) dataset [82].

The FFHQ dataset was introduced in 2019 by NVIDIA. It contains 70,000 images of people collected from Flickr at  $1024 \times 1024$  resolution. The dataset also includes a .json file with links to the original images, their landmarks, and the distribution license for each image. These images are centered and depict a wide variety of people of different ages and races. The advantages of this set are mainly high resolution and the number of available faces. The disadvantage for our purposes is an excess of photographs of children or people with make-up that do not have many wrinkles. The dataset includes faces in different poses and under various light conditions. All images displayed from the FFHQ dataset in this thesis are under PDM 1.0 Deed of CC0 1.0 Creative Commons licenses.

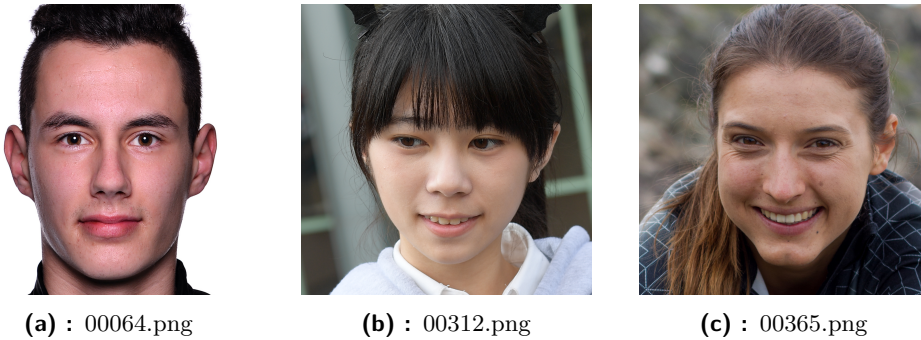
The wrinkle dataset was completed in five stages.

### ■ Stage 1 - Selection from Images 0-999

In the first stage out of the first 1000 images of the FFHQ dataset, 192 images were selected. The exclusion criteria were:

- image of non-person
- image of another face covering the main face
- image of a child
- image of a person with excessive make-up
- image under difficult light conditions
- image of people facing too much to a side
- images with sunglasses
- image that was too blurry

No gender or race was preferred during the selection. The number of wrinkles was also not a selection criterion, because the PNHHC dataset includes people with various wrinkle counts. The reader can see examples of the selected images in Figure 4.1.



**Figure 4.1:** Samples of selected images from FFHQ dataset in Stage 1. The names of the images are the same as in the FFHQ database. Images source: [82].

### ■ Stage 2 - Fully Manual Wrinkle Annotation

The author of this thesis manually labeled the 192 selected images from the first stage. No additional images were added or discarded. Initially, pre-annotating the images via Hybrid Hessian Filter was attempted to help with the speed of labeling. However, this approach was abandoned due to a high wrinkle false positive rate. We used image equalization for wrinkle annotation to increase the contrast of the wrinkles. Background pixels and pixels of eyebrows, mouth, and eyes were not used when calculating the image's cumulative distribution function.

### ■ Stage 3 - Training First Model

For the first model, we utilized the implementation of a UNet-like neural network with dice loss from MONAI python library [84]. It differs from the original UNet in (i) a lowered number of channels per level, (ii) single convolution instead of double convolution, and (iii) residual blocks are added alongside the convolutions. In Python, we call the model as follows:

```

1 model = monai.networks.nets.UNet(
2     spatial_dims=2,
3     in_channels=3,
4     out_channels=1,
5     channels=(32, 64, 128, 256, 512),
6     strides=(2, 2, 2, 2),
7     num_res_units=2)

```

For augmentation of the data horizontal flip was used at a 50 % rate, and also rescaling ranging from 0.95 to 1.05 with rotation up to  $7^\circ$  was used at a 30 % rate. One epoch consisted of 160 images from the first stage, 32 were used for validation. The learning rate of Adam optimizer was set to 0.01 with an exponential learning rate scheduler  $\gamma = 0.95$ .

### ■ Stage 4 - Selection from Images 1000-2999

Criteria for selection from these images were similar to Stage 1, but the criteria for exclusion were less strict - we allowed for images with more difficult light conditions and blurrier images. In total new 482 images were selected out of 2000. We tried to include more faces with their eyes closed, as in the PNHC dataset videos people blink and we did not want our network to overfit to images where people have their eyes open. Examples of images selected from Stage 4 are displayed in Figure 4.2.



(a) : 01222.png



(b) : 02434.png



(c) : 02497.png

**Figure 4.2:** Samples of selected images from FFHQ dataset in Stage 4. (a) image with no difficult conditions; (b) an item present in front of woman's face; (c) closed eyes and more difficult light condition. Images source: [82].

## ■ Stage 5 - Semi-Manual Annotation of Wrinkles

Images selected from Stage 4 were first labeled by the network from Stage 3 and later these labels were manually adjusted again. Upon completion, the wrinkle dataset contained 674 manually annotated faces, automatically cropped to remove excessive background.

## ■ 4.2 Neural Network Training

We trained multiple network architectures to see which performs the best: (i) Classical UNet with padding; (ii) UNet with single convolution and ResNet blocks; (iii) UNet++ with deep supervision as used in [68]; (iv) Striped WriNet as used in [81]; (v) Deeplab v3 with ResNet 101 backbone; and (vi) FCN with ResNet 101 backbone. For the loss function, we used dice loss, in case of (iii) four dice losses, one for each level. In our code, we called the models for reproducibility as follows:

```

1 from monai.networks.nets import BasicUNet, UNet,
  ↪ BasicUnetPlusPlus
2 from striped_wriNet import *
3 from torchvision.models.segmentation import fcn_resnet101,
  ↪ deeplabv3_resnet101
4
5 model_1 = BasicUNet(spatial_dims=2, in_channels=3,
  ↪ out_channels=1, features=(64, 128, 256, 512, 1024, 64))
6
7 model_2 = monai.networks.nets.UNet(spatial_dims=2,
  ↪ in_channels=3, out_channels=1, features=[32, 64, 128, 256,
  ↪ 512], strides=[2, 2, 2, 2], num_res_units=4)
8
9 model_3 = BasicUNetPlusPlus(spatial_dims = 2, in_channels = 3,
  ↪ out_channels = 1, features=(32, 64, 128, 256, 512, 32),
  ↪ deep_supervision=True)
10
11 model_4 = StripedWriNet(n_channels=3, n_classes=1)
12
13 model_5 = deeplabv3_resnet101(num_classes = 1)
14
15 model_6 = fcn_resnet101(num_classes = 1)

```

The Monai library in Python is part of the the MONAI project, which is set to provide open-source user-friendly worksheets for neural network training with a focus on biomedicine [84]. `Striped_wriNet` is the Python file containing WriNet’s architecture provided by their authors [81] and Torchvision is a library from the PyTorch creators containing neural network models and other powerful tools for NN training [85].

We split the wrinkle dataset into training and testing datasets at a 90 %-10 % ratio. The learning rate of Adam optimizer was set to  $10^{-2}$  for `model_2` and to  $10^{-3}$  for other models heuristically with exponential decay learning rate

scheduler,  $\gamma = 0.95$ . The images and labels were resized to  $H = 928 \times W = 768$ , for them to be divisible by 32 which is required by UNet architecture. [65]. It was also attempted to train networks with histogram-equalized images as it was beneficial for manual wrinkle annotation or to train them only with segmented face patches without background. For augmentation of the data horizontal flip and affine transform were used at the same rate as in Stage 3 of wrinkle dataset creation, the augmentation was computed in Python Albumentations library [86]. Models were trained on T4 GPU on Google Colab.

For video analysis, the same model architecture as for model\_2 was trained on resized images and labels to  $H = 640 \times W = 480$  to approximately match the size of faces from the PNHC dataset's videos.

## 4.3 Video Processing

Apart from the author of this thesis's bachelor's work, there has been no comprehensive automatic evaluation study of segmented wrinkles and how they relate to hypomimia in PD. Therefore, the video processing was designed to produce characteristics representing hypomimia manifestation debated in Section 2.7.

### 4.3.1 Segmentation of Face Images

For face image segmentation we decided to use Google MediPipe [87] face landmarks, a free-to-use solution that produces 478 landmarks with  $x$ ,  $y$ , and estimated  $z$  coordinates. The predicted face coordinates of a single image are depicted in Figure 4.3. These landmarks were used for face segmentation into (i) the upper half; (ii) the bottom half; (iii) the left half; (iv) the right half; (v) the left nasolabial fold; (vi) the right nasolabial fold. The masks were created as a convex polygon containing landmarks bordering particular regions. Parts of the segmented face are depicted in Figure 4.4.

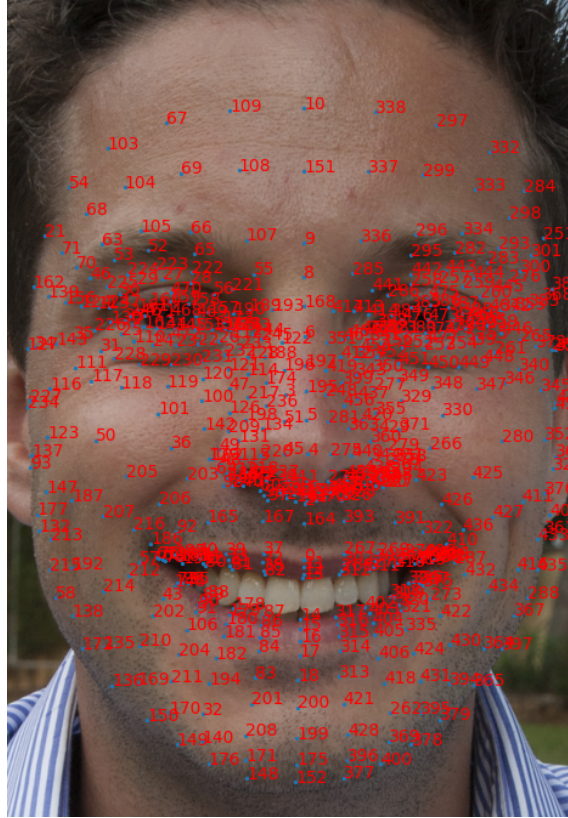
### 4.3.2 2D Face Alignment

Face alignment is important for wrinkle movement analysis because when faces are not aligned, it is impossible to tell if segmented wrinkles' movement originates in facial expression change or head movement.

We have a source image  $I_s$  and a target image  $I_t$  to which we want the source image aligned. We predict landmark coordinates for both images  $crd_s$  and  $crd_t$ , for 2D affine transformation we need only the  $x$  and  $y$  coordinate generated by MediaPipe, thus  $crd_s, crd_t \in \mathbb{R}^{478 \times 2}$ . The 2D affine transformation can be defined by a matrix  $T \in \mathbb{R}^{2 \times 3}$  by the following equation:

$$\begin{bmatrix} \hat{x}_t \\ \hat{y}_t \end{bmatrix} = T \cdot \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} \quad (4.1)$$





**Figure 4.3:** 478 Google MediaPipe landmarks detected for one person's face.

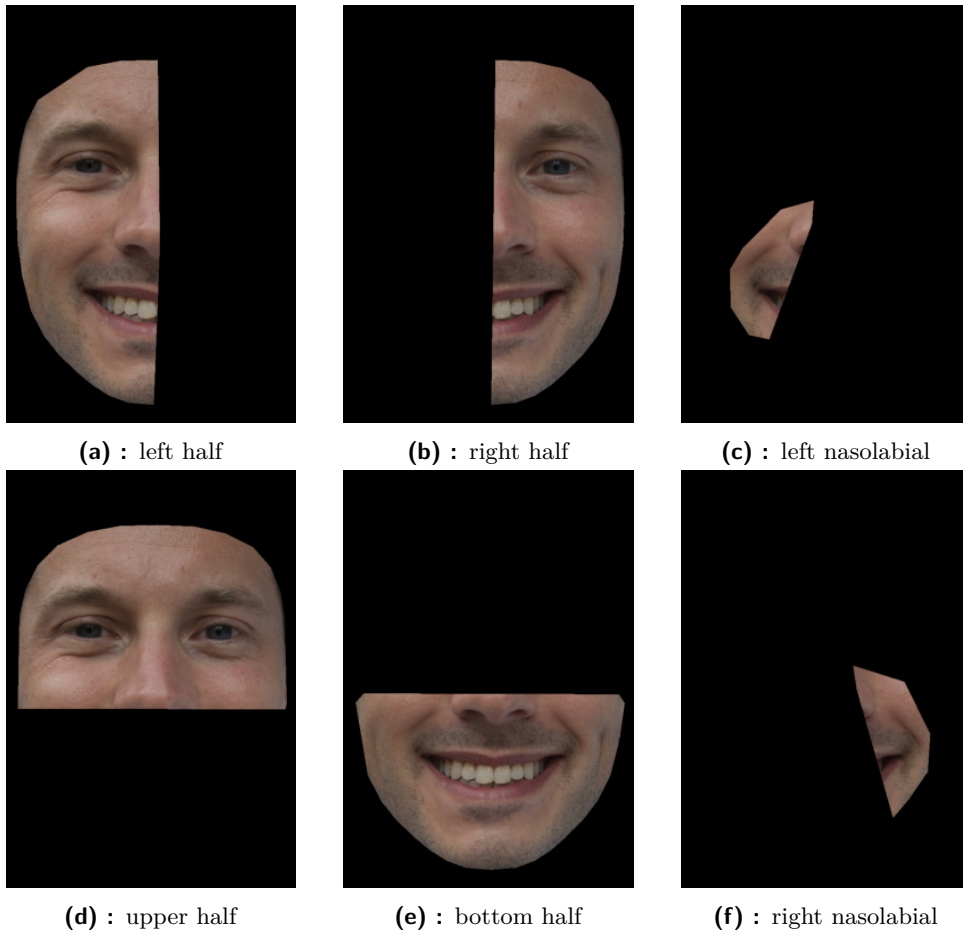
where  $(x_s, y_s)$  are the coordinates in source image and  $(\hat{x}_t, \hat{y}_t)$  is their coordinates after applying the transformation. The goal is to minimize the sum

$$\sum_{i=0}^{478-1} \left\| \begin{bmatrix} \hat{x}_{t;i} \\ \hat{y}_{t;i} \end{bmatrix} - \begin{bmatrix} x_{s;i} \\ y_{s;i} \end{bmatrix} \right\|^2 \quad (4.2)$$

which can be solved in the least square sense. After calculating matrix  $T$  we can apply it to  $I_s$ . This is implemented in the OpenCV's [44] *warpAffine* function. This affine transformation contains rotation in the image plane, scaling, and shearing, which can be sufficient for aligning images that are close to each other, for example, when  $I_s$  and  $I_t$  are subsequent frames of a video.

### 4.3.3 3D Face Alignment

Assessing depth from a single image is an ill-posed problem. However, when a priori knowledge is incorporated,  $z$  coordinate of facial landmarks can be estimated. 3D transformation relies on interpolating  $z$  coordinate value from landmarks to target image regular grid. Then the target grid is transformed by inverse transformation matrix  $T^{-1}$  to obtain estimates of the coordinates in the source image domain, and finally, the corresponding RGB values can



**Figure 4.4:** Six segmented parts of one face.

be found. Matrix  $T$  can be estimated by least squares minimization, which is generalized previous case to three dimensions, or the following equations can generate a  $T$  of user's choice:

$$T = \left[ \begin{array}{c|c} R_z \cdot R_y \cdot R_x & \begin{matrix} x_{shift} \\ y_{shift} \\ z_{shift} \end{matrix} \end{array} \right] \quad (4.3)$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{bmatrix} \quad (4.4)$$

$$R_y = \begin{bmatrix} \cos(\theta_x) & 0 & \sin(\theta_x) \\ 0 & 1 & 0 \\ \sin(\theta_x) & \cos(\theta_x) & 0 \end{bmatrix} \quad (4.5)$$

$$R_z = \begin{bmatrix} \cos(\theta_x) & -\sin(\theta_x) & 0 \\ \sin(\theta_x) & \cos(\theta_x) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (4.6)$$

where  $\theta_x, \theta_y, \theta_z$  are rotation angles corresponding to coordinate axes. The 3D transformation of a grid was applied in an article about facial landmark estimation [88].

#### 4.3.4 Hidden Landmarks Elimination

We offer a procedure for eliminating face landmarks occluded by the face surface since some of the landmarks predicted can be localized on parts of the face not visible from the camera viewpoint. Transformations involving occluded landmarks may cause ambiguities and artifacts in the result. We transform the face landmarks into vertices of a graph and give each vertex  $v$  the following attributes:

- $x$  -  $x$  coordinate: set to  $x$  coordinate of the corresponding landmark
- $y$  -  $y$  coordinate: set to  $y$  coordinate of the corresponding landmark
- $z$  -  $z$  coordinate: set to  $z$  coordinate of the corresponding landmark
- $neighs$  - list of vertices: set to list of  $v$ 's neighbors, this information is provided by MediaPipe. Sort them by  $z$  coordinate
- $surface$  - condition: set to *False*, indicates if  $v$  is on surface
- $queue$  - condition: set to *False*, indicates if  $v$  ever got onto queue

---

#### Hidden Landmarks Elimination Algorithm:

1. Create a mask  $M$  corresponding to image grid, set all pixels of  $M$  to 0
2. Create queue  $Q$
3. Push vertex  $v^{min}$  with lowest  $z$  to  $Q$ 
  - $v_{queue}^{min} = True$
  - $v_{surface} = True$ , because the vertex with the lowest  $z$  coordinate must be on visible surface
4. While  $Q$  is not empty:
  - a. pop vertex  $v^i$  from  $Q$
  - b. if  $M_{v_y^i, v_x^i} = 0$ :
    - $v_{surface}^i = True$

- find all vertices  $v^j$  in  $v_{neighs}^i$  so that  $v_{surface}^j = True$
- **if** at least 3 such  $v^j$  are found:
  - compute the convex polygon of  $x$  and  $y$  coordinates  $v^j$  and  $v^i$  vertices
  - set all pixels of  $M$  inside the polygon to 1
- **for**  $v^k \in v_{neighs}^i$  push  $v^k$  on  $Q$  and set  $v_{queue}^k = True$  **if**  $v_{queue}^k = False$

Upon terminating, all vertices will have attribute  $v_{surface}$  set to *True* when they belong to the face surface visible from the camera point and vice versa. The algorithm is based on a breadth-first search with nodes sorted by their  $z$  coordinate.

More flexible non-linear transformations (Thin-plate spline, piece-wise affine) that work with image coordinates can be a powerful tool. Thin-plate spline (TPS) is a transformation that simulates physical forces induced by bending a metal plate [89] and piece-wise affine applies a 2D affine to each triangle from a grid formed by image landmarks.

When transforming by a non-linear transformation from  $crd_s$  to  $crd_t$ , many artifacts can occur. To make this transformation more rigid, we can multiply  $crd_t$  by  $T$  to receive  $\hat{crd}_t$  and apply a non-linear transformation from  $crd_s$  to  $\hat{crd}_t$ , furthermore we reduce the artifacts by elimination of hidden landmarks. Piece-wise affine transformation used is implemented in the scikit-image Python library and interpolation for TPS and 3D grid transformation was calculated with the help of the SciPy interpolate functions.

### 4.3.5 Wrinkle Parameters Calculation

We denote  $W_k$  the detected wrinkle mask for the whole image  $I_k$ , with  $H_k$  denoting the Hybrid Hessian Filter response. Then we define  $L_k, R_k, U_k, B_k, NL_k$ , and  $NR_k$  as a mask for the face left, right, upper, bottom, left nasolabial, and right nasolabial parts respectively, masks applied to an image were shown in Figure 4.4. In our work, we partially utilized the implementation of the Frangi filter found online, our modification is in splitting 2D convolution kernel into two 1D kernels for computational speed up. From one image  $I_k$  we can calculate the following parameters:

$$ALL_k := \sum \sum W_k \quad (4.7)$$

$$U - B_k := \frac{\sum \sum (W_k \odot U_k - W_k \odot B_k)}{ALL_k + 1} \quad (4.8)$$

$$L - R_k := \frac{\sum \sum W_k \odot L_k - W_k \odot R_k}{ALL_k + 1} \quad (4.9)$$

$$TUBU := \sqrt{\frac{\sum \sum W_k \odot (H_k \odot NL_k + H_k \odot NR_k)}{(\sum_i \sum_j W_k \odot (NL_k + NR_k)) + 1}}, \quad (4.10)$$

$\odot$  denotes element-wise matrix multiplication,  $\sum \sum$  denotes summation of all elements. For two images  $I_k$  and  $I_l$  we define these parameters:

$$JSI_{k,l}^{ALL} := JSI(W_k, W_l) \quad (4.11)$$

$$JSI_{k,l}^{U-B} := JSI(W_k \odot U_k, W_l \odot U_l) - JSI(W_k \odot B_k, W_l \odot B_l) \quad (4.12)$$

$$JSI_{k,l}^{L-R} := JSI(W_k \odot L_k, W_l \odot L_l) - JSI(W_k \odot R_k, W_l \odot R_l) \quad (4.13)$$

$$TUBU_{k,l}^{DIFF} := \sqrt{\frac{\sum \sum W_k \odot (|H_k - H_l| \odot NL_k + |H_k - H_l| \odot NR_k)}{(\sum_i \sum_j W_k \odot (NL_k + NR_k)) + 1}}, \quad (4.14)$$

$JSI$  is defined in Equation 3.10, and  $|\cdot|$  in this case also denotes the element-wise absolute value of a matrix

The next list explains the reasoning behind choosing these parameters:

- *ALL*: Low variability of *ALL* parameter should represent lack of appearing and disappearing wrinkles due to hypomimia
- *U - B*: It has been suggested that when smiling, there is a lack of movement in the upper part of the face. Higher peaks of *U - B* compared to baseline could be associated with hypomimia present.
- *L - R*: Asymmetry has not been associated with hypomimia, but tremor in PD patients mostly predominates on one side. A high variability of *L - R* would suggest asymmetric wrinkle arising.
- *TUBU*: This variable measures the average tubularity of nasolabial folds. Lower tubularity change would suggest the presence of hypomimia because hypomimia induces flattened nasolabial folds.
- *JSI<sup>ALL</sup>*: The Jaccard Similarity Index between image wrinkles is a way to compare how much the wrinkles changed between frames. Lower *JSI<sup>ALL</sup>* implicates higher movement of wrinkles between frames.
- *JSI<sup>U-B</sup>*: This measures the same phenomenon as *U - B*, higher *JSI<sup>U-B</sup>* means a higher discrepancy in movement between the lower and upper half of the image face.
- *JSI<sup>L-R</sup>*: If this value differs significantly from zero, it suggests asymmetry in wrinkle movement.
- *TUBU<sup>DIFF</sup>* Lower difference in tubularity between frames can be potentially a key parameter for hypomimia detection [5].

---

**Parameter Calculation Procedure for One Video:**

1. Load NN model for wrinkle segmentation (to GPU if possible)
2. Set batch size = 5, W = 480, H = 640
3. Create variables for saving calculated parameters
4. Allocate memory for the batch as  $B \times C \times H \times W$  array
5. load video and while video loader is not empty:
  - read 5 video frames
  - calculate landmarks
  - resize images to  $H \times W$ , transform the landmarks' coordinates accordingly
  - load images into preallocated memory
  - segment wrinkles through the model
  - calculate single image parameters for each image of the batch
  - calculate two image parameters for each pair of aligned subsequent images
  - calculate two image parameters for the first and last aligned image of the batch
  - save the parameters

This procedure applied to 1500-frame video results in 1500 values for single image parameters ( $ALL$ ,  $U - B$ ,  $L - R$ ,  $TUBU$ ) and 1500 for two image parameters: 1200 for subsequent images ( $JSI^{ALL2}$ ,  $JSI^{U-B2}$ ,  $JSI^{L-R2}$ ,  $TUBU^{DIFF2}$ ) and 300 ( $JSI^{ALL5}$ ,  $JSI^{U-B5}$ ,  $JSI^{L-R5}$ ,  $TUBU^{DIFF5}$ ) for the first and last image of each batch.

We denote variables which will be a single number calculated from parameters across the images for one video.

$$normstd^{ALL} := \frac{std(ALL)}{mean(ALL)} \quad (4.15)$$

$$std_{U-B} := std(U - B) \quad (4.16)$$

$$std^{L-R} := std(L - R) \quad (4.17)$$

$$normstd_{TUBU} := \frac{std(TUBU)}{mean(TUBU)} \quad (4.18)$$

$$mean_{JSI}^{ALL2} := mean(JSI^{ALL2}), mean_{JSI}^{ALL5} := mean(JSI^{ALL5}) \quad (4.19)$$

$$mean_{JSI}^{U-B2} := mean(|JSI^{U-B2}|), mean_{JSI}^{U-B5} := mean(|JSI^{U-B5}|) \quad (4.20)$$

$$mean_{JSI}^{L-R2} := mean(|JSI^{L-R2}|), mean_{JSI}^{L-R5} := mean(|JSI^{L-R5}|) \quad (4.21)$$

$$\begin{aligned} normstd_{TUBU}^{DIFF2} &:= \frac{std(TUBU^{DIFF2})}{mean(TUBU^{DIFF2})}, \\ normstd_{TUBU}^{DIFF5} &:= \frac{std(TUBU^{DIFF5})}{mean(TUBU^{DIFF5})} \end{aligned} \quad (4.22)$$

## 4.4 Data analysis

The procedure above parametrized all videos of 100 PN and 100 HC and then variables from the Equations 4.15 - 4.22 for each study participant were calculated. The Kolmogorov-Smirnov test was performed to assess the normality of the data. Consequently, depending on the result, an unpaired two-sample t-test or Mann-Whitney U-test was performed to evaluate the statistical difference of computed variables between PD and HC. Computed variables were also used to classify HC from PD with a logistic regression classifier. Because we performed 12 tests, we adjusted  $\alpha_{adj} = \frac{0.05}{12}$  by the Bonferroni correction for significance value.

# Chapter 5

## Results

This section shows the results of the procedures described in Chapter 4.

### 5.1 Dataset Results

First, we display the results of the wrinkle dataset composure. Figure A.1 illustrates the result of pre-labeling image with HHF filter. In Figure A.2 we show the contrast enhancement by histogram equalization to ease the manual wrinkle annotation. In Figure A.3 we show the results of wrinkle pre-annotation by CNN trained in Stage 3 of wrinkle dataset assembling. Lastly, we display manually annotated examples from the wrinkle dataset in Figure A.4.

### 5.2 Model Training Results

Table 5.1 evaluates the performance of models used for wrinkle segmentation. For the PNHC dataset evaluation, we chose model 1-Conv UNet because it had a lower computation time and memory size. A new model of the same architecture was trained again on resized images of the wrinkle dataset with a mean JSI of 31.24 %. The results of wrinkle segmentation of the validation dataset are displayed in Figure A.5. Image enhancements such as histogram equalization, removing the background, or changing the augmentations did not result in a noticeable improvement.

### 5.3 Face Alignment Results

Figure A.6 illustrates the elimination of facial landmarks occluded by the face surface. The entire algorithm ran for  $\sim 0.5$  seconds for one image. Figure A.7 provides a visual comparison of the used alignment methods. Because the FFHQ dataset does not intentionally contain multiple images of the same person, we used a stock video (attributions: Free Broll by Videezy!). There was a wide disparity in the time needed for each transformation. 2D affine transformation required  $\sim 0.25$  seconds, 3D grid transformation  $\sim 2.23$



Model Type	Mean JSI (%)	Epoch Time (min:sec)
UNet	32.12	6:57
1Conv UNet	31.53	2:50
UNet++	30.70	8:53
Striped WriNet	31.90	6:37
Deeplab v3	31.56	13:16
FCN ResNet 101	28.51	07:12

**Table 5.1:** Performance of models for the validation dataset. Mean JSI is the mean JSI for each image in the validation dataset. Epoch Time is a time of evaluating the first training epoch, other epochs took a similar amount of time.

seconds, piece-wise transformation  $\sim 4$  seconds and TPS took a very long time  $\sim 40$  seconds. Because of the duration of the alignments and present artifacts, we decided to use 2D Affine transformation, despite its low elasticity.

## 5.4 Video Evaluation Results

The videos were evaluated using AMD Radeon RX 590 GPU. The alignment chosen for two image parameter estimation was a 2D affine transformation. Evaluation of a one-minute video ran for  $\sim 12$  minutes and the whole evaluation process for all the videos took 41 hours with no breaks. Frames where people could not be segmented because they accidentally moved too close to the camera were omitted from the calculations. Upon visual inspection of the model performance on the PNHC videos, there were noticeable changes in wrinkle segmentation in some parts of the videos, even when the participants did not change their position or expression considerably.

## 5.5 Data analysis Results

Kolmogorov-Smirnov test for normality and subsequent t-test or Mann-Whitney U-test were performed to reveal significant differences between PN and HC groups. Results of statistical tests are provided in Table A.1 and means and standard deviations of the variables are available in Table A.2. Figures A.8 - A.11 display histograms and estimated probability density function of the 4 most significant variables. A logistic regression model was trained on the 5 most significant variables with leave-one-out cross-validation. Figures A.12 and A.13 show the Receiver Operating Characteristic (ROC) curve and confusion matrix of the logistic regression model. The accuracy of the logistic regression model was 74 %, with sensitivity of 72 %, and specificity 77 %.

## Chapter 6

### Discussion

In this chapter, we discuss our results, compare them with state-of-the-art methods, and mention limitations and possible improvements.

#### 6.1 Dataset

We completed a wrinkle dataset with a total count of 672 wrinkle-annotated images selected from the FFHQ database. Selected images cover faces in various poses, emotions, and wrinkle counts, making it challenging for correct segmentation. It also included people with glasses and objects like a microphone in front of their faces. Because a missing available dataset is one of the major issues for wrinkle segmentation, we decided to make this dataset public. Bear in mind that the dataset is imperfect, the main reason being wrinkles in some images were difficult even for a human labeler, and the labeler was not a professional dermatologist. However, we hope the provided dataset could benefit research towards more precise wrinkle segmentation models. The dataset purposely contained a higher variety of images because the networks trained for this dataset should evaluate monologue videos, where people do not sit perfectly still.

#### 6.2 Wrinkle Segmentation Model Training

The JSI of the trained networks on the validation dataset ranged from 28.51 to 32.12 %. The state-of-the-art Striped WriNet study showed JSI of 63.01 to 64.92 % for their "public" dataset and 43.65 to 45.63 % for their "private" dataset [81]. This may appear to be an underperformance from our model; however, it is hard to tell, without having access to the same datasets as other authors had. Since the WriNet article showed that using a different dataset lowered the performance of their networks by  $\sim 20\%$  it seems like the bottleneck of current wrinkle segmentation approaches is the dataset used, not the architecture of the neural network. We argue that our dataset included harder images to correctly segment, but it was crucial as the model was subsequently used for frames of PNHC videos.

Our trained architectures correctly managed not to detect many objects like hair and glass frames as false positive wrinkles, the predictions also did not contain an excess of little noisy segments but consisted of mostly compact wrinkles. However, the main drawback was that the trained networks predicted wrinkles to have almost the same width, whether the ground truth wrinkles were finer or broader.

When it came to video evaluation, the performance of these networks was not comparable with their results for the wrinkle dataset, with cropped faces from videos being  $\frac{2}{3}$  of the size the architectures were originally trained on. This is why we trained the 1-conv UNet again on smaller reshaped faces, which helped significantly. To improve the results we tried to use histogram-equalized images with no greater success even though this helped manual wrinkle annotation. Removing image content beyond the convex polygon of face landmarks produced more true positive wrinkles. Still, missing context also resulted in more hair being denoted as a false positive wrinkle, meaning JSI stayed approximately the same. Utilizing larger neural network architectures probably would not make the wrinkle segmentation more precise and it would not be suitable for video evaluation which can be time-sensitive.

### 6.3 Facial Alignment and Video Evaluation

We attempted to align faces via more elastic transformations, which can create artifacts on a face image. Artifacts were reduced when hidden face landmarks were eliminated from the transformation computations. The process took 0.5 seconds to eliminate hidden landmarks and 2 seconds for the fastest elastic transformation to be estimated. Unfortunately, even when a lot of effort was put into more flexible transformations, because of time constraints, 2D affine transformation was selected in the end. Adding 1 second to each frame alignment would result in an 84-hour increase in the duration for evaluating the entire PNHC dataset.

Despite reshaping being able to distort desired image features [65], it was necessary for image processing by UNet-like models in batches. Some of the segmented wrinkles differed too significantly between very similar image frames, we partially attribute the reason for this to the role of image compression noise. Blurring images with a 2D Gaussian partially solved the problem with compression noise, but also made the finer wrinkles indistinguishable, so we decided not to use it.

### 6.4 Data Analysis

Statistical tests confirmed significant differences in many variables computed; after adjusting the level of significance value  $\alpha_{adj}$ , 5 variables emerged as significant.

- **normstd<sub>TUBU</sub>**: This variable measured the fluctuation of the depth of nasolabial wrinkles. PN had values significantly lower which we expected

from the literature.

- $\text{mean}_{\text{JSI}}^{\text{ALL2}}$  and  $\text{mean}_{\text{JSI}}^{\text{ALL5}}$  : These parameters reflected lower wrinkle movement in PN patients resulting in higher mean JSI. The mean JSI calculated from images 5 frames apart was overall lower, which is expected as the frames are not that similar to subsequent frames. This variable indicated the differences between HC and PN the most.
- $\text{mean}_{\text{JSI}}^{\text{U-B5}}$  : Higher values of this variable for the PN group hinted at the more disproportional movement of the lower and upper part of the face. However,  $\text{mean}_{\text{JSI}}^{\text{U-B2}}$  did not confirm this phenomenon.
- $\text{mean}_{\text{JSI}}^{\text{L-R2}}$  : Higher values of this variable for HC could indicate more asymmetry in facial expressions in HC. On the contrary,  $\text{mean}_{\text{JSI}}^{\text{L-R5}}$  resulted in almost the same distribution for both HC and PN, which is a more expected result.

The three most significant variables behaved as we expected,  $\text{mean}_{\text{JSI}}^{\text{U-B5}}$  and  $\text{mean}_{\text{JSI}}^{\text{L-R2}}$  gave results that were not confirmed by their counterparts  $\text{mean}_{\text{JSI}}^{\text{U-B2}}$  and  $\text{mean}_{\text{JSI}}^{\text{L-R5}}$ . We fitted and evaluated logistic regression with leave-one-out cross-validation to obtain an accuracy of 74 %, sensitivity of 72 %, and specificity of 77 %. This is slightly worse than in [5]. We attribute this to neural networks showing unstable results between similar frames, probably because there is still not a sufficient amount of annotated data. This instability was further amplified by the video compression rate.

These results show that even the earliest stage of PD can be detected from wrinkle movement. Only one variable showed a significant difference in terms of single image parameters, other variables came from differences between aligned image pairs. Choosing a more sophisticated alignment could benefit the hypomimia detection from wrinkles, however, at this time it would prolong the computations to an immense amount of time.

## 6.5 Meeting the Thesis Goals

In this section, we reflect on how we fulfilled the goals of this thesis.

- **Study the topic of facial expressivity disruption in people with Parkinson’s disease.**  
Disruption of facial expressivity is a common symptom of Parkinson’s disease which is called hypomimia, hypomimia signs are discussed in Section 2.7. Recent studies have been focusing on the automatic of hypomimia and a review of these approaches is concluded in Table 2.1.
- **Study the topic of facial wrinkle assessment.** Wrinkle segmentation is getting more popular in the scientific community and is receiving fresh contributions from various authors. Best-performing algorithms make use of neural networks with higher JSI values than basic image segmentation methods. JSI is the most commonly used metric regarding wrinkle

segmentation evaluation. Wrinkle segmentation was discussed in Section 3.6.2.

- **Design an automatic approach for the assessment of facial expressivity disruption based on the evaluation of facial wrinkles.** Like the other authors, we decided as well to use wrinkle segmentation via a neural network. For this purpose, we completed a new wrinkle dataset and trained neural network architectures on the dataset. A selected network was then used for frame-by-frame wrinkle segmentation of videos, with wrinkle parameters being calculated with adherence to hypomimia symptoms.
- **Analyze differences in the facial characteristics of people with Parkinson’s disease and healthy control group.** For two groups of 100 de-novo-diagnosed Parkinson’s disease patients and 100 healthy controls we conducted statistical tests of the parameters established by the previous steps. 5 parameters emerged with a very high significance and showed that early symptoms of facial expressivity disruption can be detected even in the earliest stages of Parkinson’s disease from wrinkle parameters.


# Chapter 7

## Conclusion

The goal of this thesis was to study the concept of loss in facial expressivity (hypomimia) in patients with PD, wrinkle assessment approaches, and to design an automatic approach to evaluate hypomimia from wrinkles and analyze the differences found by this approach in PD patients and healthy controls.

Hypomimia is an early symptom of Parkinson's disease with new approaches for its evaluation still appearing. Many authors used face landmarks for hypomimia evaluation which met with great success, we provided a summary of approaches used in Table 2.1. Wrinkles as a mark of emotional expression could be also used to evaluate hypomimia in PD, but no study directly segmented wrinkles for this purpose.

Wrinkle assessment has risen in popularity because of the recent expansion of neural networks, which outperform prior basic image segmentation techniques. Wrinkle datasets are challenging to obtain as a result of the wide disparity of wrinkles and the precision required for correct annotation. Authors used their private datasets to evaluate different neural network architectures with Jaccard Similarity Index (JSI) being the most frequent segmentation metric. From the studies, one can assume that a wrinkle dataset can impact JSI more than the neural network architecture used.

The authors of wrinkle segmentation techniques did not provide any of their datasets online, which is why we assembled our dataset containing 672 annotated images from the Flickr-Faces HQ dataset and made it publicly available on GitHub . Architectures trained for this dataset achieved a JSI of 32% for the validation set, which is lower than the state-of-the-art, most likely as a consequence higher variety of the images selected for the dataset.

A selected trained network was used for the wrinkle segmentation of videos. Estimated parameters from wrinkles were then statically analyzed to find the differences between the PN and HC groups. Parameters that set these groups apart were utilized to fit a logistic regression classifier. The p-value of three parameters was  $< 0.001$  and the classifier performed classification of HC from PN with an accuracy of 74 %.

Even though we showed that wrinkle formation in HC and PN significantly differs, wrinkle parameters seem not to be the most informative criteria obtainable from the videos and need a large amount of resources to estimate.

Room for improvement lies in making models for wrinkle segmentation more reliable and stable between video frames. Better results from the video classification could be achieved by higher quality videos or by combining the parameters obtained from wrinkle segmentation with ones from other approaches that evaluate hypomimia.



## Bibliography

- [1] W. Poewe, K. Seppi, C. M. Tanner, G. M. Halliday, P. Brundin, J. Volkman, A.-E. Schrag, and A. E. Lang, “Parkinson disease,” *Nature reviews Disease primers*, vol. 3, no. 1, pp. 1–21, 2017.
- [2] B. R. Bloem, M. S. Okun, and C. Klein, “Parkinson’s disease,” *The Lancet*, vol. 397, no. 10291, pp. 2284–2303, 2021.
- [3] J. Jankovic *et al.*, “Parkinson’s disease: clinical features and diagnosis,” *Journal of Neurology Neurosurgery and Psychiatry*, vol. 79, no. 4, pp. 368–376, 2008.
- [4] M. Bologna, G. Fabbrini, L. Marsili, G. Defazio, P. D. Thompson, and A. Berardelli, “Facial bradykinesia,” *Journal of Neurology, Neurosurgery & Psychiatry*, 2012.
- [5] J. Vaník, “Detekce vrásek pro popis hypomimie u parkinsonovy nemoci,” B.S. thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2022.
- [6] M. M. McGregor and A. B. Nelson, “Circuit mechanisms of parkinson’s disease,” *Neuron*, vol. 101, no. 6, pp. 1042–1056, 2019.
- [7] A. M. Graybiel, “The basal ganglia,” *Current biology*, vol. 10, no. 14, pp. R509–R511, 2000.
- [8] C. Frank, G. Pari, and J. P. Rossiter, “Approach to diagnosis of parkinson disease.,” *Canadian family physician*, vol. 52, no. 7, pp. 862–868, 2006.
- [9] C. G. Goetz, B. C. Tilley, S. R. Shaftman, G. T. Stebbins, S. Fahn, P. Martinez-Martin, W. Poewe, C. Sampaio, M. B. Stern, R. Dodel, *et al.*, “Movement disorder society-sponsored revision of the unified parkinson’s disease rating scale (mds-updrs): scale presentation and clinimetric testing results,” *Movement disorders: official journal of the Movement Disorder Society*, vol. 23, no. 15, pp. 2129–2170, 2008.
- [10] P.-A. Fall, A. Saleh, M. Fredrickson, J.-E. Olsson, and A.-K. Granérus, “Survival time, mortality, and cause of death in elderly patients with parkinson’s disease. a 9-year follow-up,” *Movement disorders: official*



- journal of the Movement Disorder Society*, vol. 18, no. 11, pp. 1312–1316, 2003.
- [11] T. Thervupettagam, “TNPSC Current affairs, Monthly TNPSC Current affairs, TNPSC Portal Current affairs in English — tnpssc-thervupettagam.com.” <https://www.tnpssc-thervupettagam.com/currentaffairs-detail/world-parkinsons-day-april-11?cat=important-days>. [Accessed 28-01-2024].
- [12] R. B. Postuma, D. Berg, M. Stern, W. Poewe, C. W. Olanow, W. Oertel, J. Obeso, K. Marek, I. Litvan, A. E. Lang, *et al.*, “Mds clinical diagnostic criteria for parkinson’s disease,” *Movement disorders*, vol. 30, no. 12, pp. 1591–1601, 2015.
- [13] A. Grammatikopoulou, N. Grammalidis, S. Bostantjopoulou, and Z. Katsarou, “Detecting hypomimia symptoms by selfie photo analysis: for early parkinson disease detection,” in *Proceedings of the 12th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, pp. 517–522, 2019.
- [14] T. Maycas-Cepeda, P. López-Ruiz, C. Feliz-Feliz, L. Gómez-Vicente, R. García-Cobos, R. Arroyo, and P. J. García-Ruiz, “Hypomimia in parkinson’s disease: what is it telling us?,” *Frontiers in Neurology*, vol. 11, p. 603582, 2021.
- [15] G. Su, B. Lin, J. Yin, W. Luo, R. Xu, J. Xu, and K. Dong, “Detection of hypomimia in patients with parkinson’s disease via smile videos,” *Annals of Translational Medicine*, vol. 9, no. 16, 2021.
- [16] M. Novotny, T. Tykalova, H. Ruzickova, E. Ruzicka, P. Dusek, and J. Rusz, “Automated video-based assessment of facial bradykinesia in de-novo parkinson’s disease,” *NPJ digital medicine*, vol. 5, no. 1, p. 98, 2022.
- [17] L. Marsili, R. Agostino, M. Bologna, D. Belvisi, A. Palma, G. Fabbrini, and A. Berardelli, “Bradykinesia of posed smiling and voluntary movement of the lower face in parkinson’s disease,” *Parkinsonism & related disorders*, vol. 20, no. 4, pp. 370–375, 2014.
- [18] B. Jin, Y. Qu, L. Zhang, and Z. Gao, “Diagnosing parkinson disease through facial expression recognition: video analysis,” *Journal of medical Internet research*, vol. 22, no. 7, p. e18697, 2020.
- [19] Y. Liqiong, C. Xiangling, G. Quanhao, J. Zhang, L. Man, C. Xiaqing, W. Yanxia, Z. Xianwei, and X. Fan, “Changes in facial expressions in patients with parkinson’s disease during the phonation test and their correlation with disease severity,” *Computer Speech & Language*, vol. 72, p. 101286, 2022.

- [20] A. Bandini, S. Orlandi, H. J. Escalante, F. Giovannelli, M. Cincotta, C. A. Reyes-Garcia, P. Vanni, G. Zaccara, and C. Manfredi, "Analysis of facial expressions in parkinson's disease through video-based automatic methods," *Journal of neuroscience methods*, vol. 281, pp. 7–20, 2017.
- [21] M. Rajnoha, J. Mekyska, R. Burget, I. Eliasova, M. Kostalova, and I. Rektorova, "Towards identification of hypomimia in parkinson's disease based on face recognition methods," in *2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 1–4, IEEE, 2018.
- [22] D. Kempler and D. Van Lancker, "Effect of speech task on intelligibility in dysarthria: A case study of parkinson's disease," *Brain and language*, vol. 80, no. 3, pp. 449–464, 2002.
- [23] S. Skodda, W. Grönheit, N. Mancinelli, U. Schlegel, *et al.*, "Progression of voice and speech impairment in the course of parkinson's disease: a longitudinal study," *Parkinson's disease*, vol. 2013, 2013.
- [24] D. A. Rahn III, M. Chou, J. J. Jiang, and Y. Zhang, "Phonatory impairment in parkinson's disease: evidence from nonlinear dynamic analysis and perturbation analysis," *Journal of Voice*, vol. 21, no. 1, pp. 64–71, 2007.
- [25] J. Ruzs, T. Tykalová, R. Krupička, K. Zárubová, M. Novotný, R. Jech, Z. Szabó, and E. Ržička, "Comparative analysis of speech impairment and upper limb motor dysfunction in parkinson's disease," *Journal of Neural Transmission*, vol. 124, pp. 463–470, 2017.
- [26] F. Amato, L. Borzi, G. Olmo, C. A. Artusi, G. Imbalzano, and L. Lopiano, "Speech impairment in parkinson's disease: acoustic analysis of unvoiced consonants in italian native speakers," *IEEE Access*, vol. 9, pp. 166370–166381, 2021.
- [27] I. Suttrup and T. Warnecke, "Dysphagia in parkinson's disease," *Dysphagia*, vol. 31, no. 1, pp. 24–32, 2016.
- [28] M. D. Lewek, R. Poole, J. Johnson, O. Halawa, and X. Huang, "Arm swing magnitude and asymmetry during gait in the early stages of parkinson's disease," *Gait & posture*, vol. 31, no. 2, pp. 256–260, 2010.
- [29] X. Huang, J. M. Mahoney, M. M. Lewis, G. Du, S. J. Piazza, and J. P. Cusumano, "Both coordination and symmetry of arm swing are reduced in parkinson's disease," *Gait & posture*, vol. 35, no. 3, pp. 373–377, 2012.
- [30] K. L. Chou, M. Evatt, V. Hinson, and K. Kompoliti, "Sialorrhea in parkinson's disease: a review," *Movement Disorders*, vol. 22, no. 16, pp. 2306–2313, 2007.

- [31] N. Miller, M. Walshe, and R. W. Walker, “Sialorrhea in parkinson’s disease: Prevalence, impact and management strategies,” *Research and Reviews in Parkinsonism*, pp. 17–28, 2019.
- [32] J. Nienstedt, C. Buhmann, M. Bihler, A. Niessen, R. Plaetke, C. Gerloff, and C. Pflug, “Drooling is no early sign of dysphagia in parkinson s disease,” *Neurogastroenterology & Motility*, vol. 30, no. 4, p. e13259, 2018.
- [33] S. Hall, Y. Surova, A. Öhrfelt, S. B. Study, K. Blennow, H. Zetterberg, and O. Hansson, “Longitudinal measurements of cerebrospinal fluid biomarkers in parkinson’s disease,” *Movement Disorders*, vol. 31, no. 6, pp. 898–905, 2016.
- [34] L. P. Oosterveld, I. M. Verberk, N. K. Majbour, O. M. El-Agnaf, H. C. Weinstein, H. W. Berendse, C. E. Teunissen, and W. D. van de Berg, “Csf or serum neurofilament light added to  $\alpha$ -synuclein panel discriminates parkinson’s from controls,” *Movement Disorders*, vol. 35, no. 2, pp. 288–295, 2020.
- [35] B. F. Boeve, “Idiopathic rem sleep behaviour disorder in the development of parkinson’s disease,” *The Lancet Neurology*, vol. 12, no. 5, pp. 469–482, 2013.
- [36] C. H. Schenck, B. F. Boeve, and M. W. Mahowald, “Delayed emergence of a parkinsonian disorder or dementia in 81% of older men initially diagnosed with idiopathic rapid eye movement sleep behavior disorder: a 16-year update on a previously reported series,” *Sleep medicine*, vol. 14, no. 8, pp. 744–748, 2013.
- [37] G. Su, B. Lin, J. Yin, W. Luo, R. Xu, J. Xu, and K. Dong, “Detection of hypomimia in patients with parkinson’s disease via smile videos,” *Annals of Translational Medicine*, vol. 9, no. 16, 2021.
- [38] I. R. I. Haque and J. Neubert, “Deep learning approaches to biomedical image segmentation,” *Informatics in Medicine Unlocked*, vol. 18, p. 100297, 2020.
- [39] M. Basavarajaiah, “6 basic things to know about Convolution — bdhuma.” <https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411>, 2019. [Accessed 02-05-2024].
- [40] M. Kumar, “How Padding helps in CNN ? — numpyninja.com.” <https://www.numpyninja.com/post/how-padding-helps-in-cnn>, 2020. [Accessed 03-05-2024].
- [41] jun94, “[CV] 2. Gaussian and Median Filter, Separable 2D filter — medium.com.” <https://medium.com/jun94-devpblog/cv-2-gaussian-and-median-filter-separable-2d-filter-2d11ee022c66>, 2020. [Accessed 02-05-2024].

- [42] L. Dierkop, “Histograms and Exposure Compensation — ldierkop.” <https://medium.com/@ldierkop/histograms-and-exposure-compensation-3b7557f6c1be>, 2019. [Accessed 02-05-2024].
- [43] M. Kaur, J. Kaur, and J. Kaur, “Survey of contrast enhancement techniques based on histogram equalization,” *International Journal of Advanced Computer Science and Applications*, vol. 2, no. 7, 2011.
- [44] Itseez, “Open source computer vision library.” <https://github.com/itseez/opencv>, 2015.
- [45] R. C. Gonzalez, *Digital image processing*. Pearson education india, 2009.
- [46] J. R. Movellan, “Tutorial on gabor filters,” *Open source document*, vol. 40, pp. 1–23, 2002.
- [47] F. Yi and I. Moon, “Image segmentation: A survey of graph-cut methods,” in *2012 international conference on systems and informatics (ICSAI2012)*, pp. 1936–1941, IEEE, 2012.
- [48] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, “Multiscale vessel enhancement filtering,” in *Medical Image Computing and Computer-Assisted Intervention—MICCAI’98: First International Conference Cambridge, MA, USA, October 11–13, 1998 Proceedings 1*, pp. 130–137, Springer, 1998.
- [49] C. Banerjee, T. Mukherjee, and E. Pasiliao Jr, “An empirical study on generalizations of the relu activation function,” in *Proceedings of the 2019 ACM Southeast Conference*, pp. 164–167, 2019.
- [50] S. Sharma, S. Sharma, and A. Athaiya, “Activation functions in neural networks,” *Towards Data Sci*, vol. 6, no. 12, pp. 310–316, 2017.
- [51] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [52] M. Yani, S. M. Budhi Irawan, S, and M. Casi Setiningsih, ST, “Application of transfer learning using convolutional neural network method for early detection of terry’s nail,” in *Journal of Physics: Conference Series*, vol. 1201, p. 012052, IOP Publishing, 2019.
- [53] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A survey of convolutional neural networks: analysis, applications, and prospects,” *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.
- [54] “Conv2d &x2014; PyTorch 2.3 documentation — pytorch.org.” <https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html>. [Accessed 05-05-2024].

- [55] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.
- [56] X. Li, X. Sun, Y. Meng, J. Liang, F. Wu, and J. Li, “Dice loss for data-imbalanced nlp tasks,” *arXiv preprint arXiv:1911.02855*, 2019.
- [57] M. D. Zeiler, “Adadelta: an adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.
- [58] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [59] K. Y. Li, “How to choose a learning rate scheduler for neural networks.” <https://neptune.ai/blog/how-to-choose-a-learning-rate-scheduler>, 2023. [Accessed 13-05-2024].
- [60] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.
- [61] P. Ostwal, “Data Augmentation for Computer Vision — pranjal-ostwal.medium.com.” <https://pranjal-ostwal.medium.com/data-augmentation-for-computer-vision-b88b818b6010>, 2023. [Accessed 07-05-2024].
- [62] S. Bangar, “VGG-Net Architecture Explained — siddheshb008.” <https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f>, 2022. [Accessed 08-05-2024].
- [63] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [64] S. Bangar, “Resnet Architecture Explained — siddheshb008.” <https://medium.com/@siddheshb008/resnet-architecture-explained-47309ea9283d>, 2022. [Accessed 08-05-2024].
- [65] X. Liu, L. Song, S. Liu, and Y. Zhang, “A review of deep-learning-based medical image segmentation methods,” *Sustainability*, vol. 13, no. 3, p. 1224, 2021.
- [66] I. Berrios, “DeepLabv3 — itberrios6.” <https://medium.com/@itberrios6/deeplabv3-c0c8c93d25a4>, 2023. [Accessed 16-05-2024].
- [67] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International*

- Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241, Springer, 2015.
- [68] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, “Unet++: A nested u-net architecture for medical image segmentation,” in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pp. 3–11, Springer, 2018.
- [69] G. Lemperle, R. E. Holmes, and S. S. M. Lemperle, “A classification of facial wrinkles,” *Plastic and reconstructive surgery*, vol. 108, no. 6, pp. 1735–1750, 2001.
- [70] jian667, “GitHub - jian667/face-dataset: Face related datasets — github.com.” <https://github.com/jian667/face-dataset>, 2021. [Accessed 04-02-2024].
- [71] O. Zuk, “Top 10 Face Datasets for Facial Recognition and Analysis — datagen.tech.” <https://datagen.tech/blog/face-datasets/>, 2022. [Accessed 04-02-2024].
- [72] Shaip-admin, “15 Best Face Datasets to Train Your Facial Recognition Model — shaip.com.” <https://www.shaip.com/blog/15-free-image-datasets-to-train-facial-recognition-models/>, 2022. [Accessed 08-05-2024].
- [73] A. Mawale and A. Chaugule, “Facial wrinkles detection techniques and its application,” *International Journal of Computer Applications*, vol. 134, no. 7, pp. 5–8, 2016.
- [74] M. H. Yap, N. Batool, C.-C. Ng, M. Rogers, and K. Walker, “A survey on facial wrinkles detection and inpainting: datasets, methods, and challenges,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 5, no. 4, pp. 505–519, 2021.
- [75] A. Savran, B. Sankur, and M. T. Bilge, “Comparative evaluation of 3d vs. 2d modality for automatic detection of facial action units,” *Pattern recognition*, vol. 45, no. 2, pp. 767–782, 2012.
- [76] C.-C. Ng, M. H. Yap, N. Costen, and B. Li, “Automatic wrinkle detection using hybrid hessian filter,” in *Computer Vision—ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part III 12*, pp. 609–622, Springer, 2015.
- [77] S. Kim, H. Yoon, J. Lee, and S. Yoo, “Semi-automatic labeling and training strategy for deep learning-based facial wrinkle detection,” in *2022 IEEE 35th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 383–388, IEEE, 2022.

- [78] N. C. Ebner, M. Riediger, and U. Lindenberger, “Faces—a database of facial expressions in young, middle-aged, and older women and men: Development and validation,” *Behavior research methods*, vol. 42, pp. 351–362, 2010.
- [79] U. Sabina and T. K. Whangbo, “Nasolabial wrinkle segmentation based on nested convolutional neural network,” in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 483–485, IEEE, 2021.
- [80] M. Sanchez, G. Triginer, C. Ballester, L. Raad, and E. Ramon, “Photorealistic facial wrinkles removal,” in *Asian Conference on Computer Vision*, pp. 117–133, Springer, 2022.
- [81] M.-Y. Yang, Q.-L. Shen, D.-T. Xu, X.-L. Sun, and Q.-B. Wu, “Striped wrinet: Automatic wrinkle segmentation based on striped attention module,” *Biomedical Signal Processing and Control*, vol. 90, p. 105817, 2024.
- [82] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.
- [83] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of gans for improved quality, stability, and variation,” *arXiv preprint arXiv:1710.10196*, 2017.
- [84] W. MONAI Consortium *et al.*, “Project monai,” *Zenodo*. Available online: <https://zenodo.org/record/4323059#>. *YXaMajgzaUk* (accessed on 25 May 2020), 2020.
- [85] T. maintainers and contributors, “Torchvision: Pytorch’s computer vision library.” <https://github.com/pytorch/vision>, 2016.
- [86] E. K. V. I. I. A. Buslaev, A. Parinov and A. A. Kalinin, “Albumentations: fast and flexible image augmentations,” *ArXiv e-prints*, 2018.
- [87] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, *et al.*, “Mediapipe: A framework for building perception pipelines,” *arXiv preprint arXiv:1906.08172*, 2019.
- [88] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, “Face alignment across large poses: A 3d solution,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 146–155, 2016.
- [89] F. L. Bookstein, “Principal warps: Thin-plate splines and the decomposition of deformations,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 11, no. 6, pp. 567–585, 1989.



## Appendices



## Appendix A

### Results

In this Appendix, we display results that did not fit into the main text.

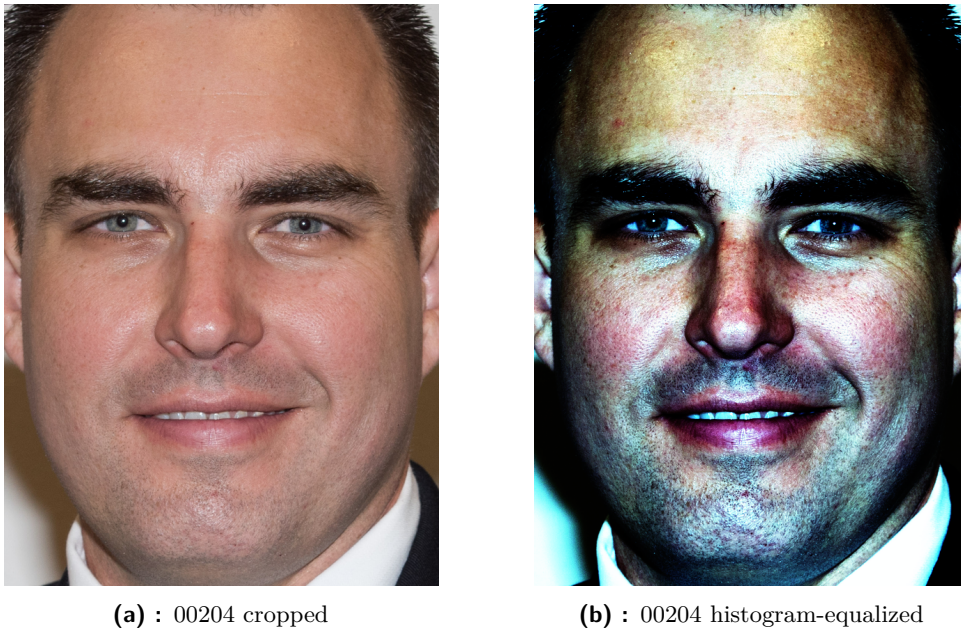


(a) : 00365 cropped



(b) : 00365 HHF

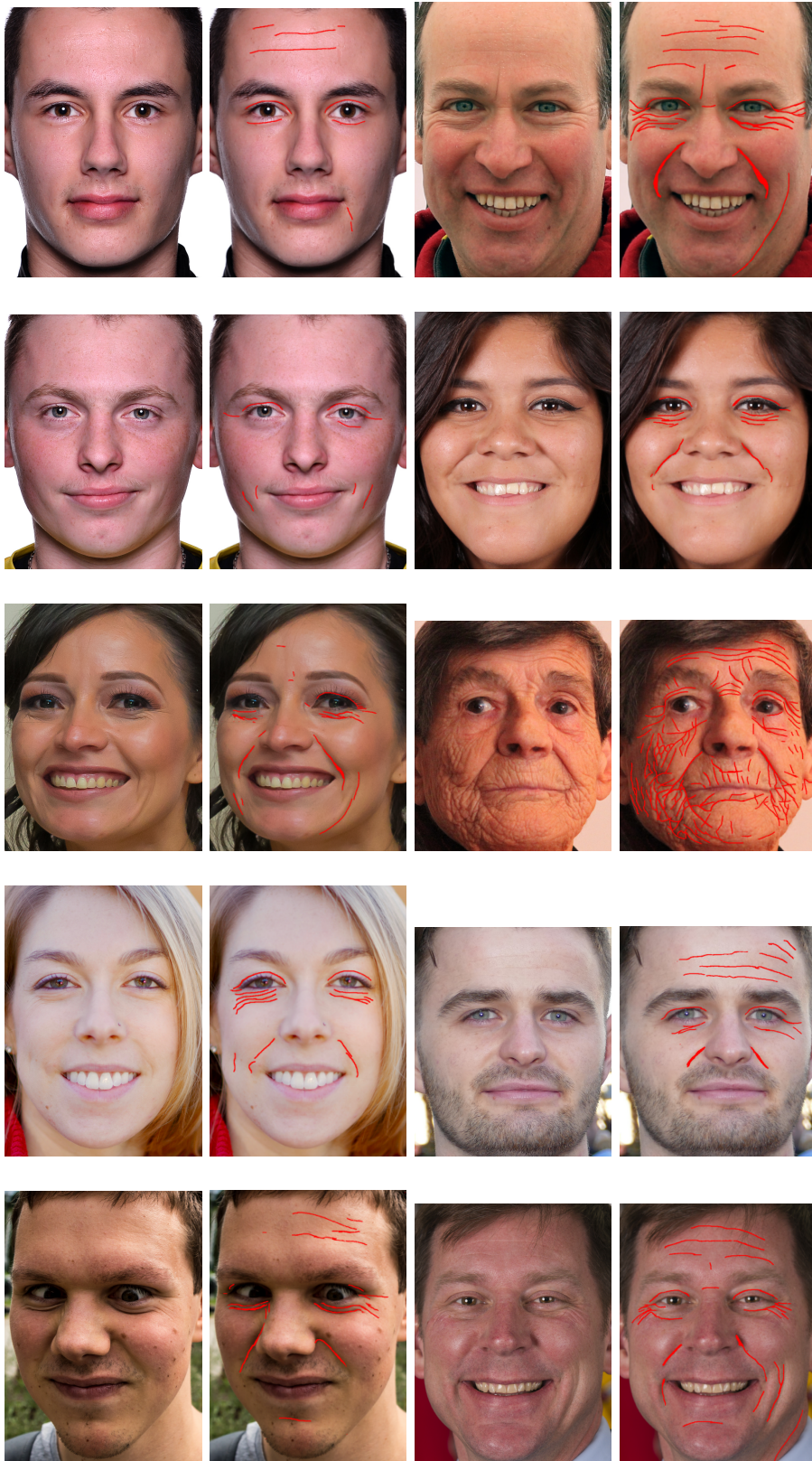
**Figure A.1:** Image 00365.png. (a) original cropped; (b) HHF labeled. HHF response produces too many false positives.



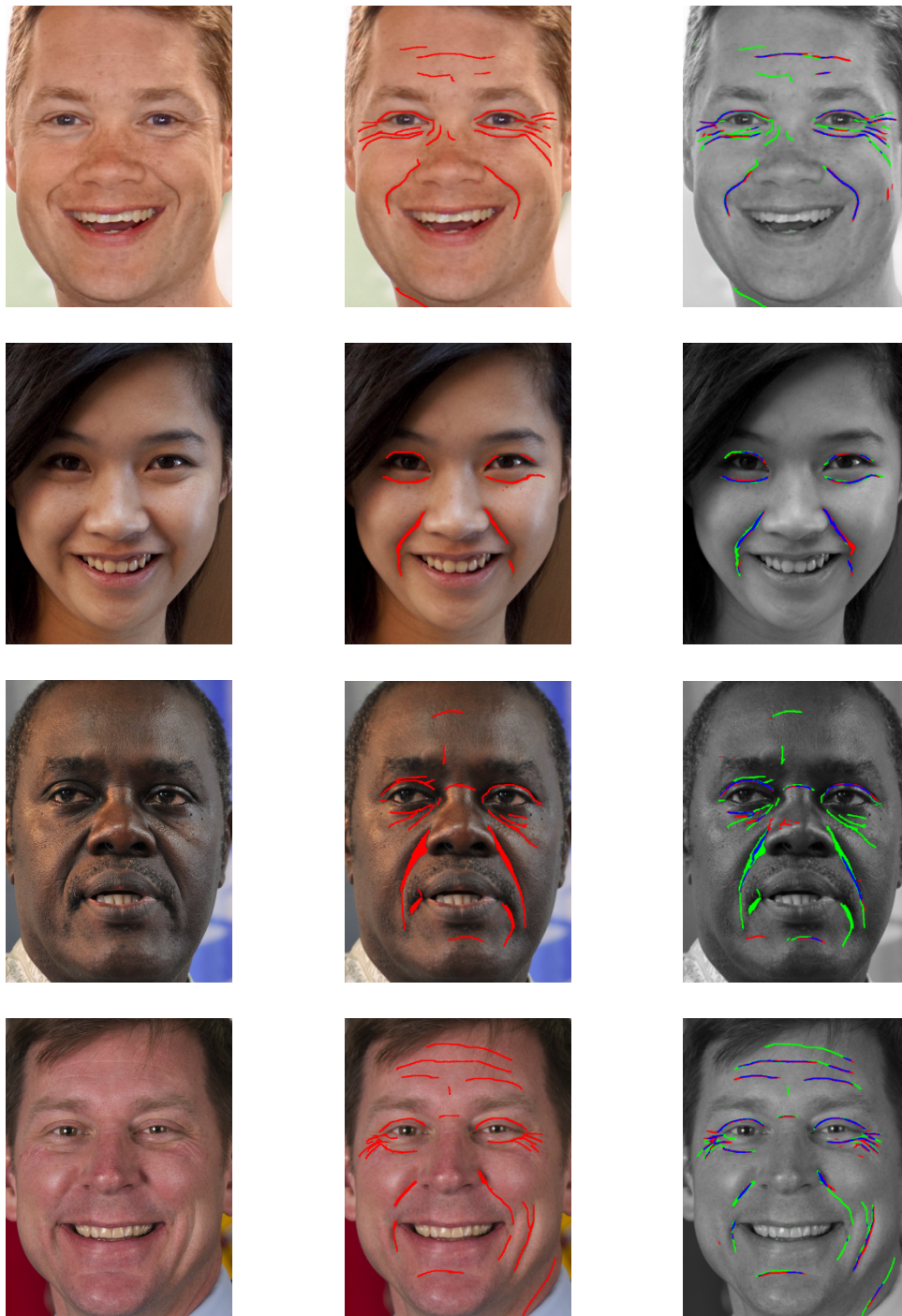
**Figure A.2:** Image 00204.png. (a) original cropped; (b) histogram-equalized. The histogram-equalized image has more apparent wrinkles.



**Figure A.3:** Images with pre-annotated wrinkles from Stage 3. Wrinkles are depicted in red. On the top left image, the model fails; on the top right, pre-annotated wrinkles adhere well and can be manually adjusted.



**Figure A.4:** Manually annotated images. The reader can observe a wide disparity of human faces in terms of total wrinkle count, position of head, and illumination.

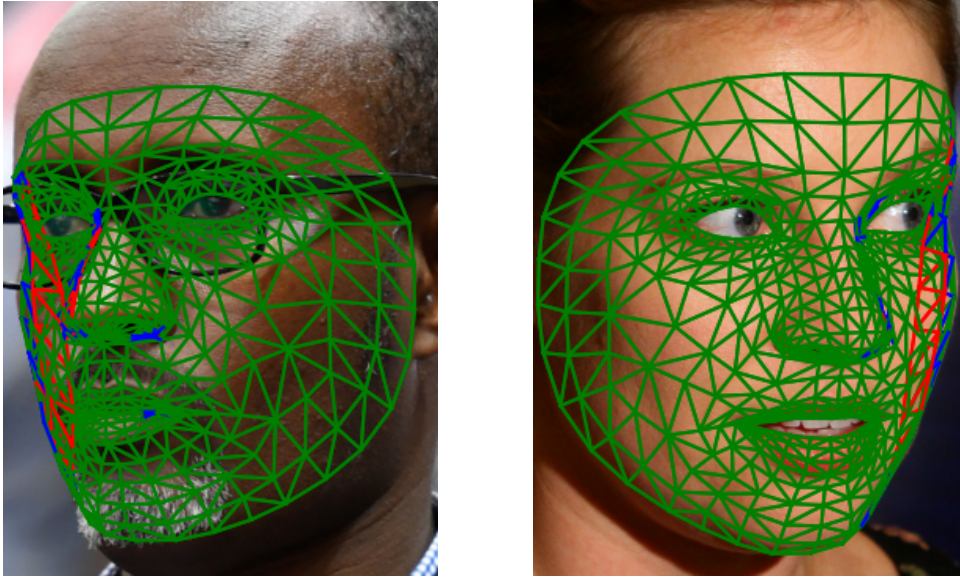


(a):

(b):

(c):

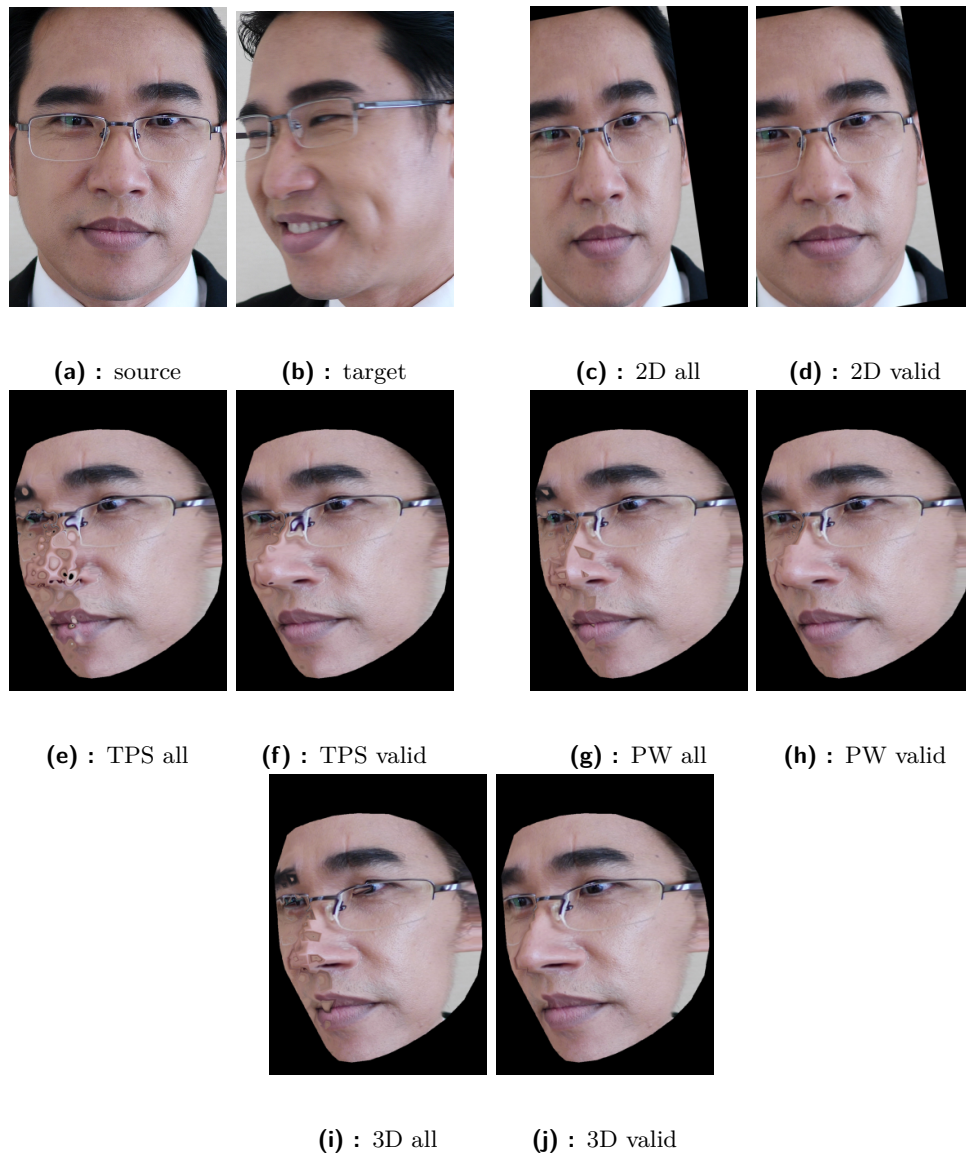
**Figure A.5:** Results of wrinkle segmentation. (a) original images; (b) manually labeled wrinkles; (c) manual labeling and prediction comparison, red = automatically segmented wrinkles, green = manually labeled wrinkles, blue = intersection of red and green.



**Figure A.6:** Results of hidden face landmarks elimination. Green lines connect landmarks on the surface, red lines connect landmarks not on the surface, and blue lines connect a landmark on the surface with a not-on-the-surface landmark.

Variable Name	Normal	p-value	t(198)	U
$normstd^{ALL}$	True	0.03*	2.136	-
$std_{U-B}$	True	0.04*	2.03	-
$std_{L-R}$	True	0.013*	2.49	-
$normstd_{TUBU}$	False	<b>&lt;0.001***</b>	-	6737
$mean_{JSI}^{ALL2}$	True	<b>&lt;0.001***</b>	-4.05	-
$mean_{JSI}^{ALL5}$	True	<b>&lt;0.001***</b>	-5.93	-
$mean_{JSI}^{U-B2}$	True	>0.1	1.34	-
$mean_{JSI}^{U-B5}$	True	<b>0.002**</b>	-3.08	-
$mean_{JSI}^{L-R2}$	False	<b>0.003**</b>	-	6224
$mean_{JSI}^{L-R5}$	False	>>0.1	-	5096
$normstd_{TUBU}^{DIFF2}$	False	0.09	-	5695
$normstd_{TUBU}^{DIFF5}$	False	0.015*	-	5994

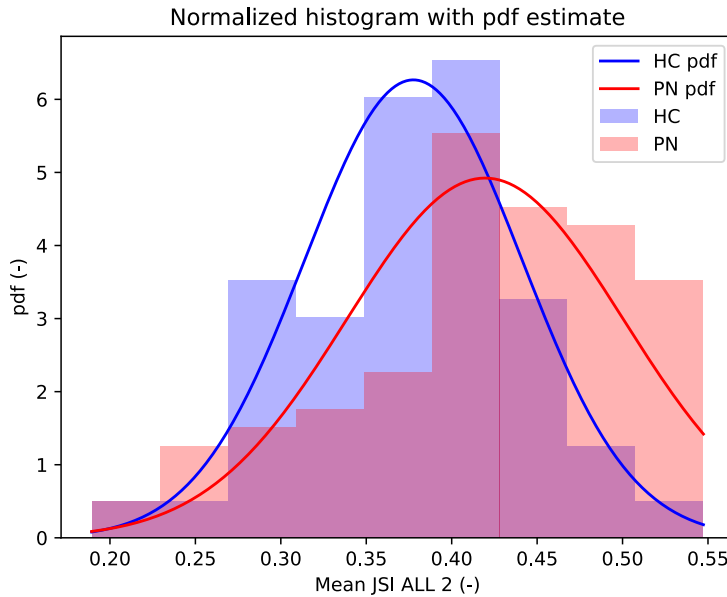
**Table A.1:** Statistical test results. p-value in bold denotes that it is lesser than  $\alpha_{adj} = \frac{0.05}{12}$ . \*  $\implies p \in [0.05, 0.01[$ , \*\*  $\implies p \in [0.01, 0.001[$ , \*\*\*  $\implies p \leq 0.001$



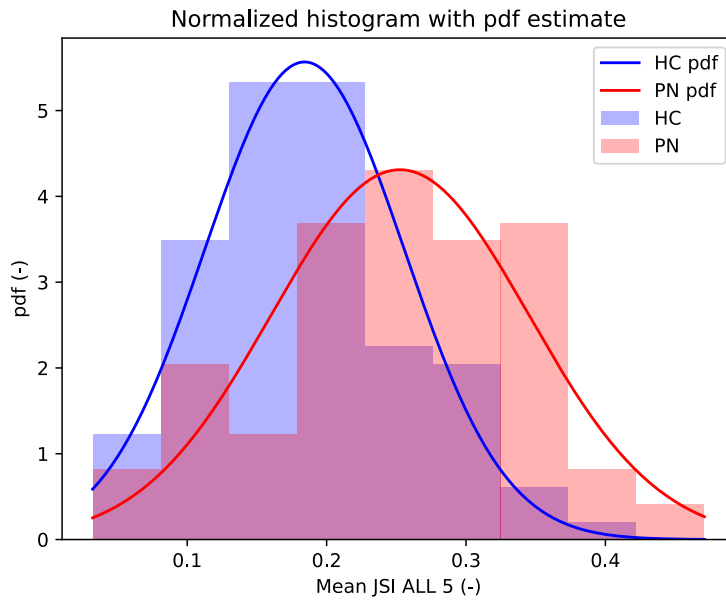
**Figure A.7:** Visualized facial alignment results. source = source image; target = target image; 2D = 2D affine transformation; TPS = thin-plate spline transformation; PW = piece-wise affine transformation; 3D = 3D grid transformation, all = all landmarks used; valid = not eliminated landmarks used. We purposely chose a major rotation angle to illustrate the artifacts.

Variable Name	Mean HC	Std HC	Mean PN	STD PN
$normstd^{ALL}$	0.145	0.049	0.130	0.052
$std^{U-B}$	0.168	0.059	0.150	0.067
$std^{L-R}$	0.163	0.056	0.143	0.053
$normstd_{TUBU}$	0.301	0.176	0.246	0.265
$mean_{JSI}^{ALL2}$	0.378	0.064	0.420	0.081
$mean_{JSI}^{ALL5}$	0.184	0.072	0.253	0.093
$mean_{JSI}^{U-B2}$	0.158	0.040	0.150	0.041
$mean_{JSI}^{U-B5}$	0.132	0.034	0.151	0.051
$mean_{JSI}^{L-R2}$	0.111	0.030	0.102	0.034
$mean_{JSI}^{L-R5}$	0.103	0.030	0.104	0.036
$normstd_{TUBU}^{DIFF2}$	0.00085	0.00030	0.00076	0.00026
$normstd_{TUBU}^{DIFF5}$	0.00114	0.00042	0.00097	0.00034

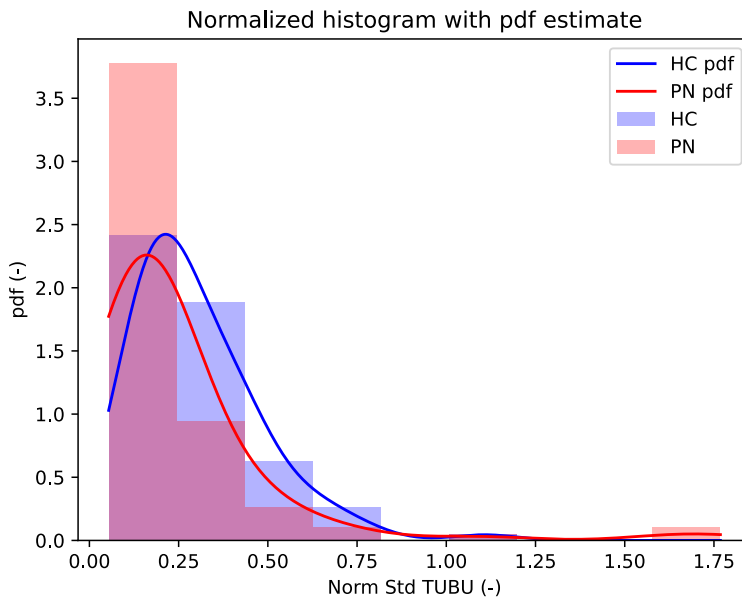
**Table A.2:** Means and standard deviations of calculated values from all 200 videos.



**Figure A.8:** Histogram of  $mean_{JSI}^{ALL2}$  plotted with pdf estimation, pdf was estimated through maximum likelihood. A higher JSI indicates more similarity between frames.

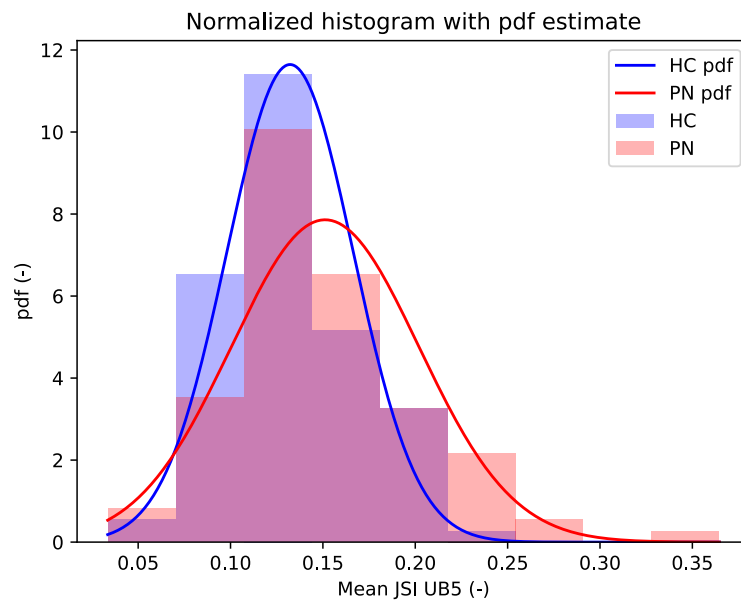


**Figure A.9:** Histogram of  $mean_{JSI}^{ALL5}$  plotted with pdf estimation, pdf was estimated through maximum likelihood. A higher JSI indicates more similarity between frames.

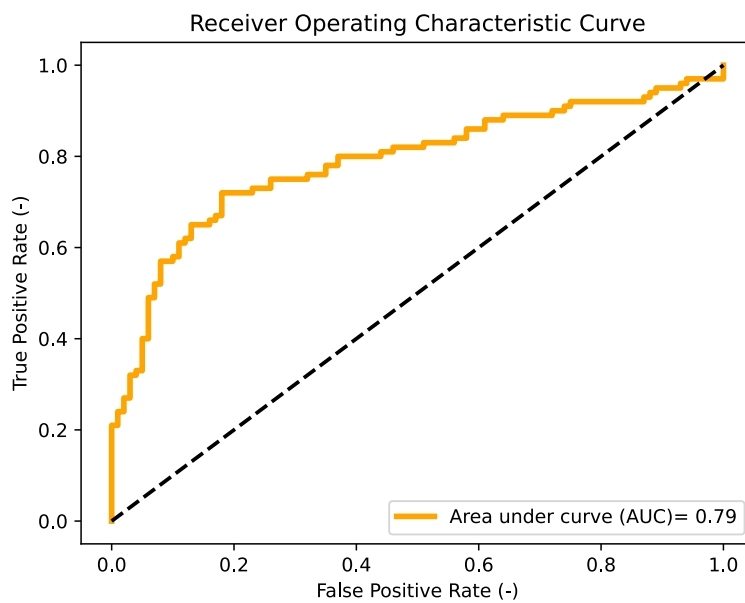


**Figure A.10:** Histogram of  $normstd_{TUBU}$  plotted with pdf estimation, pdf was estimated through kernel estimation. A higher value indicates more variability between frames.

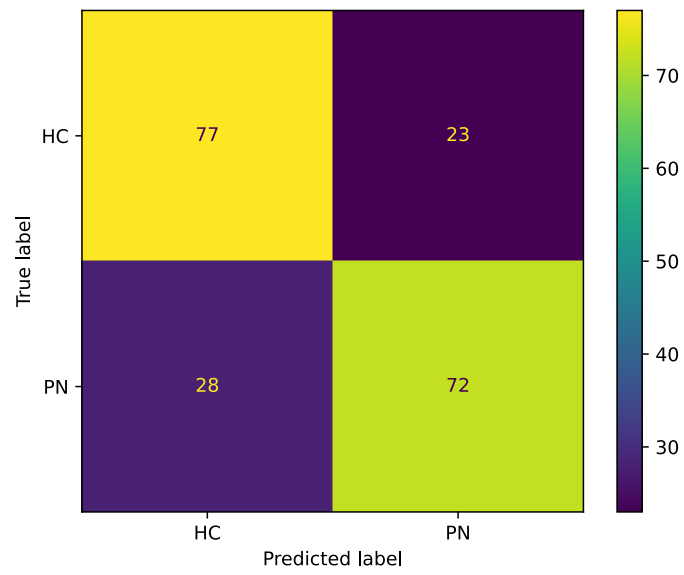




**Figure A.11:** Histogram of  $mean_{JSI}^{U-B5}$  plotted with pdf estimation, pdf was estimated through kernel estimation. A higher value indicates a higher difference in the upper and lower parts of the face.



**Figure A.12:** Receiver Operating Characteristic Curve of logistic regression classifier trained for 5 values with the most significant difference between PD and HC.



**Figure A.13:** Confusion matrix of logistic regression classifier trained for 5 values with the most significant difference between PD and HC.