# A New Approach for Pediatric Posterior Fossa Semantic Segmentation in Magnetic Resonance Images

Larissa de Oliveira Penteado Dias

Thesis text
presented to the
Instituto de Matemática e Estatística
of the
Universidade de São Paulo
as part of the Master's Program

Program: Computer Science

Supervisor: Prof. Dr. Roberto Marcondes Cesar Junior

During the development of this work, the author received
financial support from FAPESP (grants #2018/07386-5, #2019/16112-9, #2017/50236-1 and
#2015/22308-2) and from CAPES (grant #1767508)

São Paulo, July 2022

This is the submitted version
of the master's thesis text
by Larissa de Oliveira Penteado Dias

Examining Body:

- Prof. Dr. Roberto Marcondes Cesar Junior (Supervisor) - IME-USP
- Prof. Dr. Sergio Shiguemi Furuie - POLI-USP
- Profa. Dra. Regina Celia Coelho - UNIFESP

This work received financial funding from the São Paulo Research Foundation (FAPESP), grants #2018/07386-5, #2019/16112-9, #2017/50236-1 and #2015/22308-2, and from CAPES grant #1767508. We thank both foundations for their support.

## Abstract

DIAS L. O. P. **A New Approach for Pediatric Posterior Fossa Semantic Segmentation in Magnetic Resonance Images**. 2022. 80f. Thesis - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2022.

The analysis of brain magnetic resonance imaging (MRI) exams is an essential task for the diagnosis and treatment of various diseases. The manual examination of such images is time-consuming and prone to inter observer variability. Moreover, the analysis of neonatal and pediatric exams poses intrinsic challenges due to the smaller size of the brain structures and the greater inter patient variability, because of the children's neurological development, especially during the first two years of life. Therefore, the development of automatic methods to perform the semantic segmentation of MRI data is important to aid the doctors at examining such images. In order to automatically obtain the segmentation of a MRI volume, there are both 2D and 3D methods. Fully Convolutional Neural Networks (FCN) have been presenting increasingly better results at the segmentation of both natural and medical images. In this project, we developed a new approach to perform the segmentation of the posterior fossa and the fourth ventricle regions on pediatric brain MRI data, using the FCN called LiviaNet, which is a patch 3D approach. These are the regions of occurence of the medulloblastoma, a common cancer that affects children's brains. The identification of this tumor is of interest for the doctors from the Children's Institute (HC-FMUSP). They provided 32 MRI volumes for this project, from children with ages ranging from less than a year to 18 years. Our method was able to identify the region of interest with a mean dice score of 0.74, thus showing the potential of the proposed approach.

**Keywords:** Magnetic Resonance Imaging, Semantic Segmentation, Fully Convolutional Neural Networks, Pediatric Brain Segmentation, Posterior Fossa

## Resumo

DIAS, L. O. P. **Uma Nova Abordagem para Segmentação Semântica da Fossa Posterior em Imagens Pediátricas de Ressonância Magnética**. 2022. 80f. Dissertação - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2022.

A análise de exames de Ressonância Magnética (RM) cerebral é essencial para o diagnóstico e tratamento de diversas doenças. O estudo manual destas imagens é demorado e suscetível a variações entre especialistas. Além disso, a análise de exames neonatais e pediátricos apresenta desafios intrínsecos devido ao menor tamanho das estruturas cerebrais e à maior variabilidade interpaciente, que ocorre por causa do desenvolvimento neurológico das crianças, principalmente durante os primeiros dois anos de vida. Deste modo, o desenvolvimento de métodos automáticos para segmentar os exames de RM é importante para auxiliar os médicos ao examinar estas imagens. Redes Neurais Totalmente Convolucionais (do inglês, FCN) têm apresentado resultados cada vez melhores na segmentação de ambas imagens naturais e médicas. Neste projeto, desenvolvemos uma nova abordagem para realizar a segmentação das regiões da fossa posterior e do quarto ventrículo em dados de ressonância magnética de cérebro pediátrica, utilizando a FCN denominada LiviaNet. Essas são as regiões de ocorrência do meduloblastoma, um câncer comum que afeta o cérebro de crianças. A identificação desse tumor é de interesse dos médicos do Instituto da Criança (HC-FMUSP). Eles forneceram 32 volumes de ressonância magnética para este projeto de crianças com idades variando de menos de um ano a 18 anos. Nosso método foi capaz de identificar as regiões de interesse atingindo um dice score médio de 0.74, mostrando, deste modo, o potencial da abordagem proposta.

**Palavras-chave: Ressonância Magnética, Segmentação Semântica, Redes Neurais Totalmente Convolucionais, Segmentação de Cérebro Infantil, Fossa Posterior.**

# Contents

# List of Abbreviations

| | |
|---|---|
| MRI | Magnetic Resonance Imaging |
| FCN | Fully Convolutional Network |
| CNN | Convolutional Neural Network |
| ROI | Region of Interest |
| MB | Medulloblastoma |
| ADC | Apparent Diffusion Coefficient |
| CT | Computed Tomography |
| PReLU | Parametric Rectified Linear Unit |
| IoU | Intersection over Union |
| FP | False Positives |
| TP | True Positives |
| FN | False Negatives |
| TN | True Negatives |
| LS | Lovász-Softmax |
| EM | Expectation-Maximization |
| MRF | Markov Random Field |
| dHCP | Developing Human Connectome Project |
| GANs | Generative Adversarial Networks |
| GN | Generative Network |
| LR | Low Resolution (images) |
| CSF | Cerebrospinal Fluid |
| IBSR | Internet Brain Segmentation Repository |
| BET | Brain Extraction Tool |
| BFC | Bias-Field Normalization |
| CC | Connected Component |
| ICR-HC | Children's Institute from Hospital das Clínicas |
| FAPESP | São Paulo Research Foundation |
| ANR | Agence Nationale de la Recherche |
| IME-USP | Instituto de Matemática e Estatística da Universidade de São Paulo |

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

Magnetic Resonance Imaging (MRI) is a non invasive and non ionizing type of image acquisition based on the emission, resonance and absorption of radio-frequency (electromagnetic) waves. Such way of obtaining images can be used to exam human internal body parts, by measuring the concentration of hydrogen protons (water molecules) in each tissue of the area being examined. MRI exams provide a sequence of slices, which together form a 3D view of the examined region.

MRI exams have many medical applications, especially in the neurological area. It can be used to study strokes, infections, seizures, hemorrhages [14, 65, 62], neuro-degenerative diseases such as Alzheimer's and Parkinson's [10, 58], multiple sclerosis and various brain tumors. Also, it can be used for the radiotherapy planning [16], to determine the exact location of the area where the treatment should be applied. Similarly, it is used for surgical planning and guidance, being an important pre-operative image tool.

These images can also be used to monitor the neuro-development of neonates (premature or not) and small children. Since the human brain takes about two years to be fully developed, through the usage of MRI, it is possible to study some maturation processes, such as the myelination [59]. Also, it can be used to identify some congenital malformations, hydrocephalus, infections and infarction [17, 41, 4]. Another important application is the detection and classification of medulloblastoma [53], an invasive brain tumor that affects children.

In order to analyze MRI exams, the slices must be individually examined so that it is possible to identify the regions of interest and diseases that may be present. The manual analysis of such images requires a trained expert. It is also time-consuming and prone to intra and inter-observer variability [39]. Therefore, efforts have been made during the past decades to develop computational methods that are capable of analysing these images in an automatic (or semi-automatic) manner.

Since such techniques were developed mainly to deal with adult and healthy patients, or when the subject has a specific disease, they tend to fail when applied to pediatric data or when multiple diseases are present. In neonatal and small children brain MRIs, which is the focus of this work, the main reason why it is difficult to directly apply such methods is the

presence of some intrinsic differences. For example, it is possible to cite the smaller size of pediatric brain structures, the lower resolution and signal-to-noise ratio, the inverted contrast between the gray and white matter (due to the developing brain), the greater presence of motion artifacts, and, also the reduced contrast between the tissues [10]. Given the above cited applications that the analysis of brain MRI pediatric data has, the development of automatic methods to deal with such data is important to help in the diagnosis, prognosis and treatment of various diseases and disorders.

The convolutional neural networks (CNNs) have been achieving state-of-the-art results in the tasks of image segmentation and classification. Some of these models were developed to deal with medical image data, including MRI. LiviaNet [11] is a fully convolutional network (FCN) that has achieved state-of-the-art results when performing the segmentation of the cerebellum on MRI data [8]. Since the aim of this work is the semantic segmentation of the posterior fossa (cerebellum and brain stem) and the fourth ventricle, LiviaNet is tested in order to accomplish such task.

This work has been developed as part of the project FAPESP ANR STAP in collaboration with professor Isabelle Bloch (Sorbonne Université), Doctors Marcelo Straus Takahashi and Suely Fazio Ferraciolli (Instituto da Criança do Hospital das Clínicas). We are deeply grateful for their support and collaboration.

## 1.2 Objective

The main objective of this work is to develop a methodology in order to perform the semantic segmentation of the posterior fossa (composed of brain stem and cerebellum) and fourth ventricle. In the proposed approach we use the state-of-the-art fully convolutional network LiviaNet to perform the segmentation. An example of the region of interest (ROI) can be seen in Figure 1.1. It is important to segment this area, because medulloblastomas usually occur inside it. The identification and segmentation of these tumors is an important task for the doctors at the Children's Institute of the Hospital das Clínicas. This work provides an initial methodology to identify the ROI, where this disease occurs.

## 1.3 Contibutions

The main contributions achieved by this project are:

- the literature review about the medical image segmentation area, which is useful for the STAP[1] project group – of which this work is part of;
- the dataset formation, including part of its annotation and the preprocessing pipeline;
- the methodology proposed to perform the automatic segmentation of pediatric posterior fossa, since the existing methods up to now were mainly based on the manual annotation of this region;

[1] `https://perso.telecom-paristech.fr/bloch/STAP-jobs.html`

- Besides the proposed approach, the work also includes a preliminary research on using supersegmentation algorithms for automatic segmentation of pediatric images. Although this research has not yet lead to conclusive findings, the work is described here to help future initiatives.

## 1.4 Organization

This document is organized as follows. In Chapter 2, a description of the most relevant concepts regarding this work –such as the MRI technical principles, the medulloblastoma tumor, neural networks building blocks, semantic segmentation techniques –is presented along with a review of papers about the same subjects. In Chapter 3, the proposed approach to accomplish the objective is described, detailing the preprocessing, semantic segmentation model and postprocessing steps. In Chapter 4, the experiments conducted are presented and discussed. Finally, Chapter 5 presents the discussion of the contents of this thesis, along with future work suggestions.

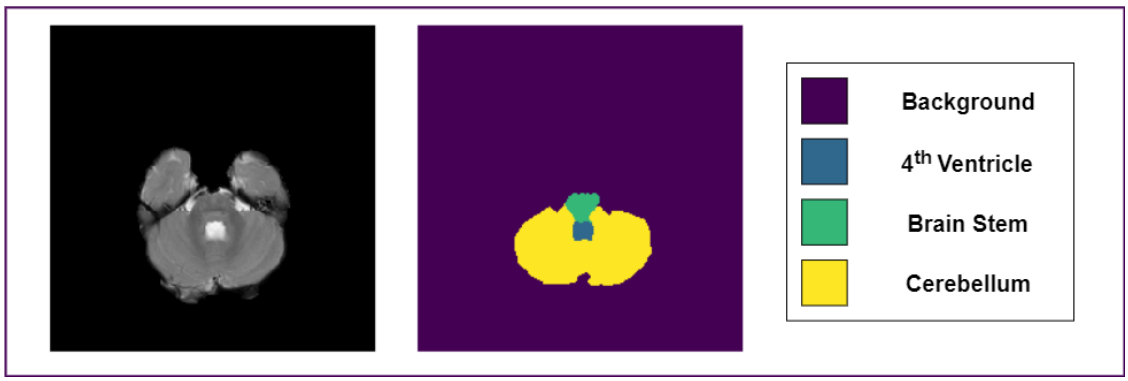



Figure 1.1: Example of a brain T2-weighted MRI slice (left) and the corresponding ROI segmentation (right).

# Chapter 2

# Concepts and Literature Review

## 2.1 MRI principles

This section is based on the work by Stacke *et al.* [50, chapter 2].

Charged particles, such as protons, spin around its own axis, which generates a small magnetic field. Each individual field is very small to be detected, however the magnetization net ($\overrightarrow{M_0}$) formed by a set of particle is detectable. When not influenced by external fields, each proton spins around a different axis (as shown in Figure 2.1 (a)), because of this $\overrightarrow{M_0} = 0$.

When exposed to a strong external magnetic field ($\overrightarrow{B_0}$), the resulting magnetic moments of the protons align in the same direction as the field, either in the parallel or the anti-parallel orientation (Figure 2.1 (b)) – there are more protons aligned in the parallel orientation, because it is the lowest energy state. Therefore, the resulting magnetization net $\overrightarrow{M_z}$ generated has the same direction as $\overrightarrow{B_0}$. Then, the particles start to precess with frequency $\omega = \gamma\overrightarrow{B_0}$ (Figure 2.2 (a)), where $\gamma$ is the gyromagnetic ratio, which is specific to each chemical element. The frequency $\omega$ is also known as Larmor precession frequency.

The receiver coils used in MRI, capture the signals coming from an oscillating magnetic field. Therefore, in order to measure the net magnetization of the region being imaged, it is necessary to oscillate the field. To do so, a radio frequency (RF) signal with frequency equal to the Larmor frequency of the protons we want to measure (usually, the hydrogen protons) is used. The RF causes the particles spins to flip to the plane orthogonal to the z-axis, as shown in Figure 2.2(b).

The protons precessing in the $xy-$plane will return to their lower energy state (aligned to the $\overrightarrow{B_0}$). Such movement releases energy, which is perceived and measured by the receiver coils. The time taken for this relaxation process to occur is called $T_1$, Equation 2.1.

$$M_z = M_0(1 - \exp^{\frac{-t}{T_1}}) \tag{2.1}$$

The net magnetization in the $xy-$plane ($\overrightarrow{M_{xy}}$), after the RF-pulse slowly disappears. The time required for it to completely decay is called $T_2$, given by Equation 2.2. The $T_1$ and $T_2$ times depends both on the tissue being examined – one proton's neighborhood spins influence its own spin – and the strength of $\overrightarrow{B_0}$.

$$M_{xy} = M_0 \exp^{\frac{-t}{T_2}} \tag{2.2}$$

Additionally, gradient coils are used to encode spatial information for each axis, during the images acquisition. The coils for axis $z$ use a varying magnetic gradient; for axis $y$, it is phase-encoding; and for axis $x$, it is frequency encoding. Figure 2.3 shows the three different views –axial, coronal and sagittal– obtained when taking a patient's MRI.

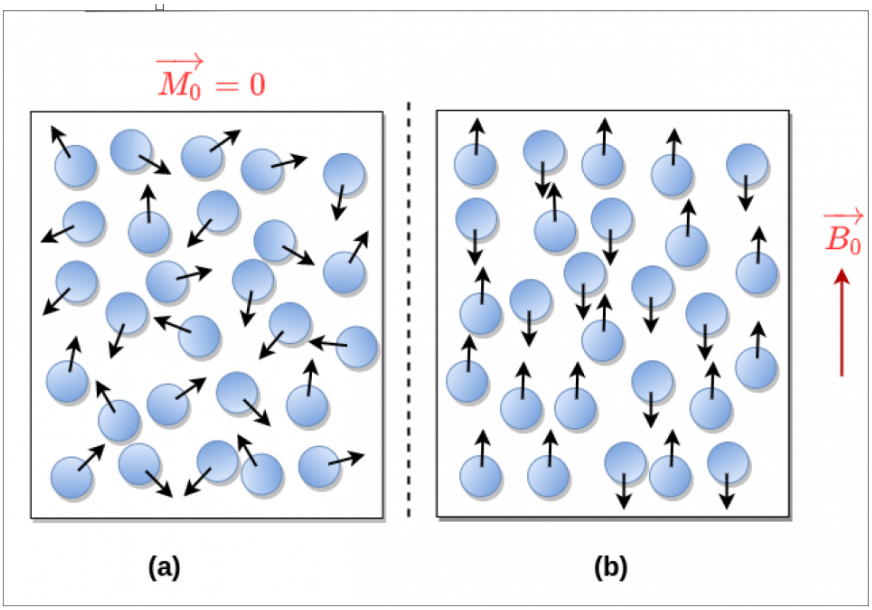



Figure 2.1: Particle's spins before (a) and after (b) the influence of the external magnetic field $\overrightarrow{B_0}$.

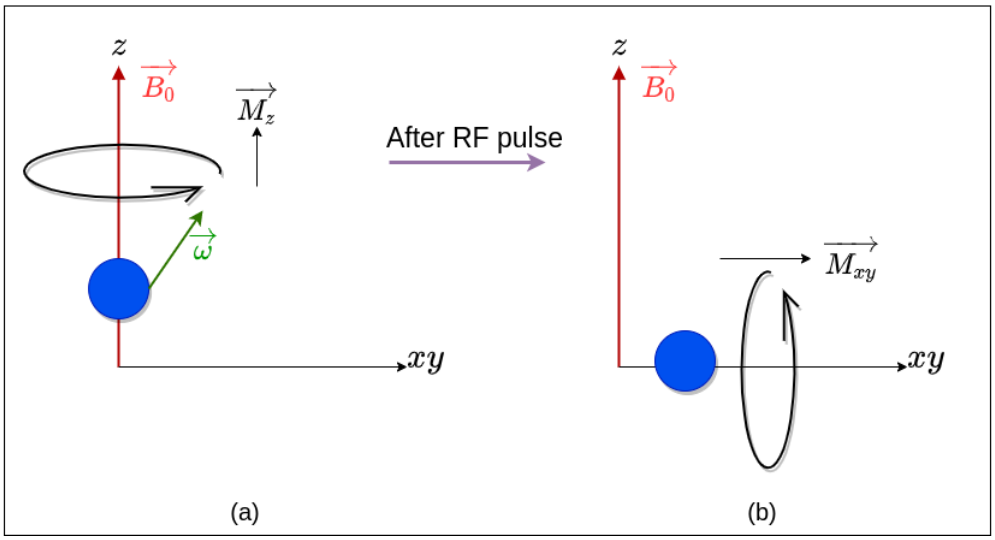



Figure 2.2: Proton precessing around $z-$axis before the RF-pulse (a); then, in the $xy-$plane after the pulse.

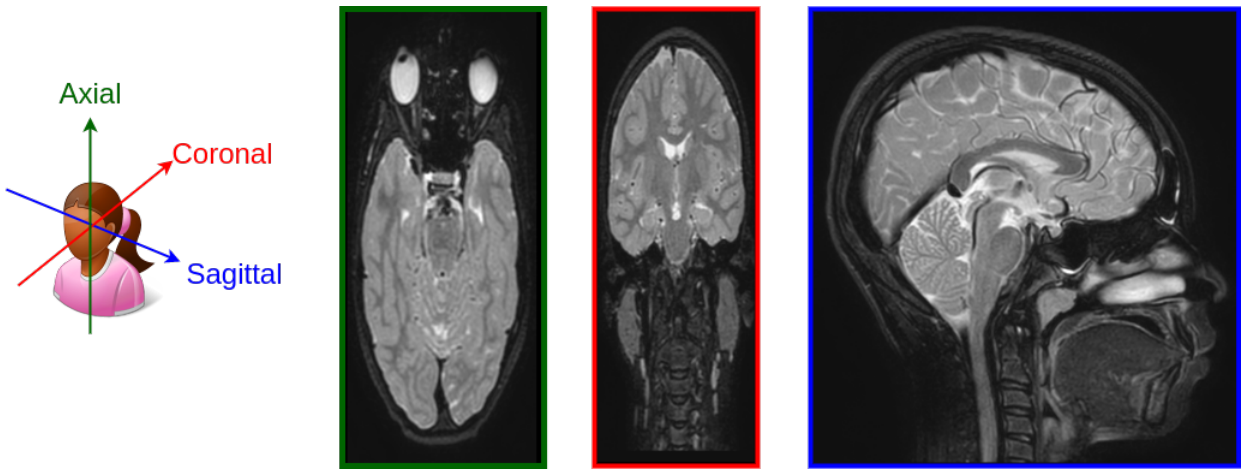



Figure 2.3: Different MRI views. With the green border is the axial one, the red corresponds to the coronal and the blue represents the sagittal view.

## 2.2 Medulloblastoma

Medulloblastoma (MB) is an invasive, malignant brain tumor. It affects mainly pediatric patients, being the most common type of cancer that affects them. MRI exams are important for the diagnosis of MB. It also plays a significant role in its treatment and surgical planning, since it may help determine the tumor's molecular subtype.

This cancer occurs in the posterior fossa, which is formed of the cerebellum and the brain stem, with 99% of times being located on the cerebellum [12]. However, in some cases it can also occupy part of the fourth ventricle [53]. Therefore, the segmentation of the posterior fossa and fourth ventricle is the aim of the present work.

In Yeom *et al.* [61] some MRI features presented in the exams of children affected with MB are investigated in order to determine if and how they relate with the tumor's histologic subtype. By analyzing the MRI diffusion sequence, the mean and minimum apparent diffusion coefficient (ADC) presented to be correlated with the MB subtype. Furthermore, by analyzing other sequences, such as T1-weighted and T2-weighted, the presence of cysts, ring enhancement and the contrast enhancement patterns also were correlated with tumor's subtype.

Eran *et al.* [12] pointed out that some findings in MRI and Computed Tomography (CT) may help discriminate MB from other tumors of the posterior fossa and also help in predicting its subtype. For example, if the tumor presented extensive nodularity in MRI images, it typically has a favorable outcome. The limited ADC helps to differentiate it from other cancers. Moreover, MRI is the best technique for discriminating an hemorrhage from a MB, which is not possible using CT images. Therefore, analysis of MRI exams plays an important and decisive role on the diagnosis and prognosis assessment of MB in children.

Shan *et al.* [47] presents a method based on rigid-body registration and active contour for the automatic segmentation of the cerebellum in children with MB. First, a template is constructed from exams of 10 patients, in which the cerebellum was manually annotated. Then, a volume of interest is registered to this template for spatial alignment. The cerebellum delineation of the template is used as the initial contour for the active contour methods,

which adjusts the boundaries in order to obtain the cerebellum segmentation of the volume of interest.

## 2.3 Neural Network Building Blocks

### 2.3.1 Convolution

A convolution consists of an operation between two functions. The operation can be both continuous as showed in Equation 2.3 or discrete Equation 2.4 –where the $\star$ represents the convolution operation symbol–, depending on the functions type. In the mentioned equations, we show the 2D version of the convolutions, but the concept can be extended for other dimensions as well. In the case of images, it is used the discrete version, and one of the functions is the image $I$ itself and the other is a mask $M$. The mask $M$ is flipped by 180 ° in all the axes depending on the dimension, then it is slided by the image and the result is the sum at each position of products between the image and the mask at each position where the kernel is centered, as showed by the example in Figure 2.4.

$$w(x,y) \star f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w(\tau_u, \tau_v) f(x - \tau_u, y - \tau_v) d\tau_u d\tau_v \tag{2.3}$$

$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x - s, y - t) \tag{2.4}$$

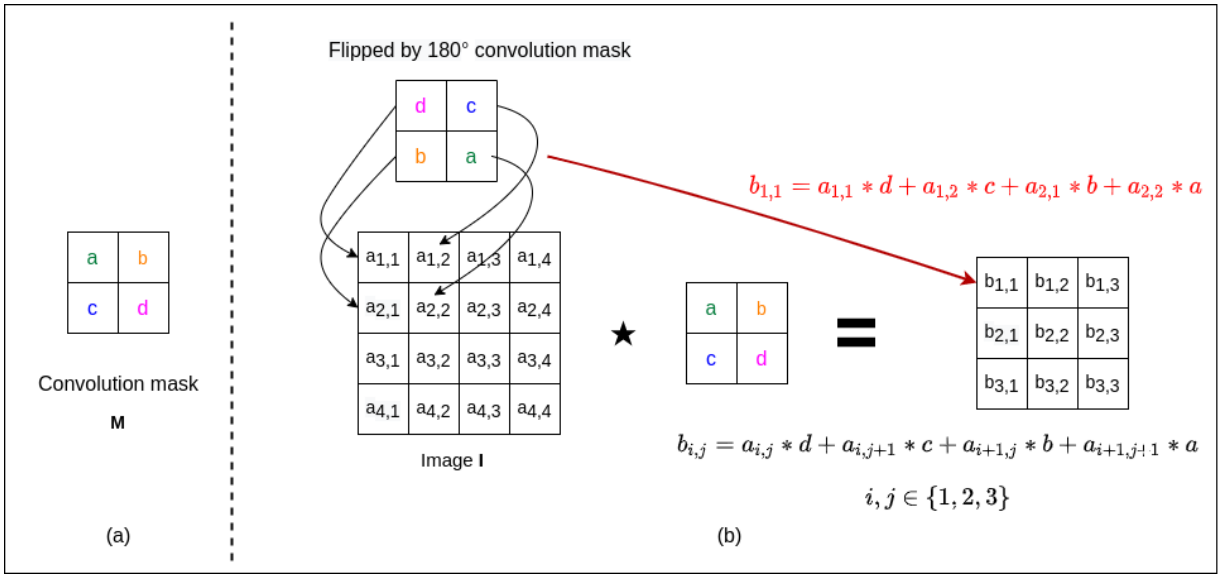



Figure 2.4: Convolution illustration. (a) A 2× 2 mask$M$ and (b) showing the result of the convolution of the image $I$ by $M$.

Suppose the image $I$ has size $H \times W$ and the mask $M$ is $k \times l$. The result of the convolution between $I$ and $M$ will have size $(H - k + 1) \times (W - l + 1)$. This happens because the window cannot be centered around the perimeter of the image. For some applications, it is desirable that the output has the same size of the input. In order to achieve it, padding methods

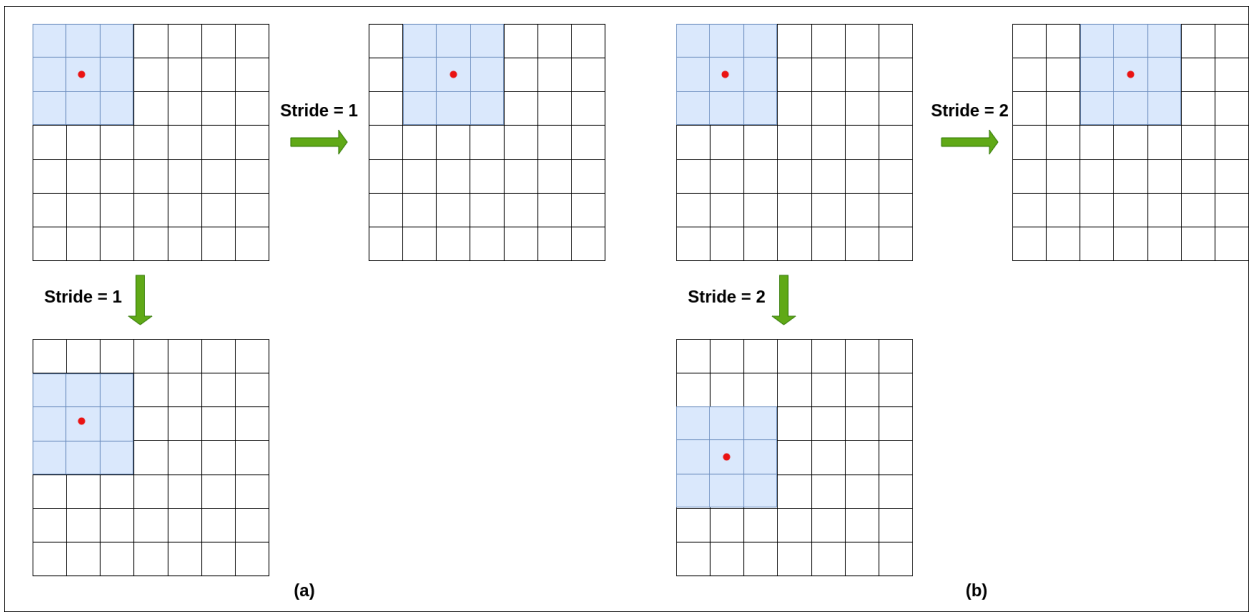



Figure 2.5: Example of a convolution step using (a) Stride 1, (b) Stride 2, both in vertical and horizontal direction. The red dot represents where the window (the blue square) is centered.

are applied to add $(k-1)/2$ rows above and below the horizontal borders and $(l-1)/2$ columns to the right and left of the original image. There are several ways of padding the image, for example, in constant-padding, these new rows and columns are composed of a constant number $c$ –zero is the most common used one, also known as zero-padding; in mirror-padding, the rows and columns from the border of the image are mirrored.

Another important concept for convolution application is the stride. It dictates the step when sliding the mask through the image. Therefore, if the convolution is being applied with stride $n$, the mask will be shifted $n$ columns or rows (if it is moving either horizontal or vertically, respectively) from the present pixel, where it is centered. Figure 2.5 shows a window moving vertical and horizontally with stride 1 and stride 2.

When the convolution operation is used in Convolutional Neural Networks (CNNs), the mask – also known as filter or kernel – have the same depth as the feature maps used as input for the convolutional layer 2.6. For example if the kernel has dimension $3 \times 3$, its depth will be $3 \times 3 \times 1$, in the case of gray scale images; $3 \times 3 \times 3$, for RGB images (the ones that have the red, green and blue channels); or $3 \times 3 \times 128$ if the input of the layer is a block of feature maps with depth 128. Therefore, each convolutional kernel produces one feature map as output. Usually, multiple kernels are used in each layer to produce more than one feature map.

### 2.3.2 $1 \times 1$ Convolution

The depth of the feature maps tend to grow in deeper layers of the CNNs. This makes the calculation of convolutions with bigger filters (e.g., $5 \times 5$, $7 \times 7$) to be computationally costly, due to the increased number of parameters. In order to alleviate this issue, $1 \times 1$ convolutions can be used in CNNs to summarize the channels of a block of feature maps [33]. As explained above, each convolution operation produces a feature map, therefore, the convolution of a

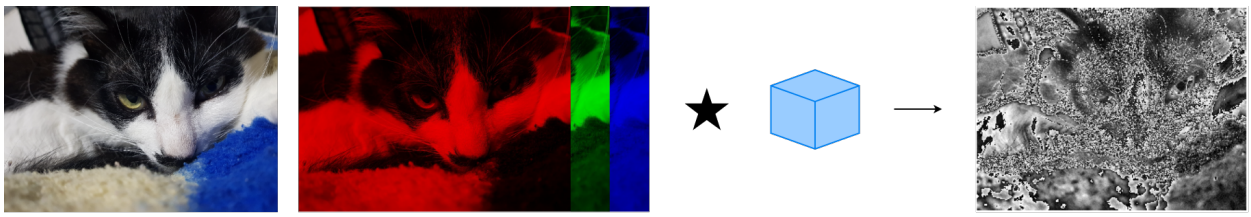



Figure 2.6: At any given layer of the network, the convolution kernel has the same depth as the input block feature maps, in order to produce one single feature map as output. Example of the convolution of a RGB image with a 3 channel kernel, resulting in an one channel image.

block of feature maps with a $1 \times 1$ kernel produces a single feature map with the height and width of the original ones. Thus, $1 \times 1$ convolutions can be interpreted as cross-channel pooling layers, used for dimensionality reduction [51].

### 2.3.3 Parametric Rectified Linear Unit

$$PReLU(x) = \begin{cases} x, & \text{if } x > 0, \\ \alpha_i x, & \text{otherwise.} \end{cases} \tag{2.5}$$

Every activation neuron has its own $\alpha_i$, which is a parameter that is learned by the network. This function is typically used because it helps avoiding vanishing gradients (zero gradients). Also, since $\alpha_i$ is a learnable parameter, it produces activation functions that are more specialized [22], that is, they can emphasize or attenuate the output of some neurons.

### 2.3.4 Softmax

Given a vector $v$, with $|v| = n$ the softmax of $v$ is expressed by Equation 2.6.

$$Softmax(v) = \frac{\exp v_k}{\sum_i \exp v_i}, k \in \{1, ..., n\} \tag{2.6}$$

The softmax [48] function is used in order to express the vector values as probabilities, note that $\sum_i v_i = 1$, after the operation. This is useful in classification and segmentation tasks –where the task is to attribute a class to an object– since the values represent the probability of each class.

### 2.3.5 Batch Normalization

The batch normalization technique, proposed by [24], is employed in neural networks in order to normalize the inputs of the layers. This operation helps to alleviate some issues, such as sensibility to random initial weights and learning rate. It also helps to speed up the optimization step of the network.

Let $Z = \{z^{(1)}, z^{(2)}, ..., z^{(m)}\}$, where $z_i \in \mathbb{R}^d$, be a set of intermediate values in the neural network. The empirical mean $\mu_Z$ and variance $\sigma_Z^2$ of $Z$ is given by the Equations 2.7 and 2.8, respectively.

$$\mu_Z = \frac{\sum_i^m z^{(i)}}{m} \tag{2.7}$$

$$\sigma_Z^2 = \frac{\sum_i^m (z^{(i)} - \mu_Z)^2}{m} \tag{2.8}$$

Then we can calculate the normalized value of $z_i$, according to Equation 2.9.

$$z_{norm}^{(i)} = \frac{z^{(i)} - \mu_Z}{\sqrt{\sigma_Z^2 + \epsilon}} \tag{2.9}$$

Finally, the $z_{norm}$ value is transformed using the parameters $\gamma$ and $\beta$, which are learned by the network during the optimization step, Equation 2.10.

$$\tilde{z}^{(i)} = \gamma z_{norm}^{(i)} + \beta \tag{2.10}$$

### 2.3.6 Loss Function

A loss function, in the context of the neural networks, compares the prediction made by the model with the ground truth, producing a real number that quantifies their difference. Since the objective is to minimize the loss, the gradient of the function is used to optimize the network. Commonly, in order to achieve it, the gradient descent optimization technique is applied to the network weights to update them in every iteration (also called, epoch).

The original training of the LiviaNet, proposed by [11], used the cross-entropy loss. In this project, other losses are also considered for training the network, due to some specific characteristics of the problem. For example, the Dice-score based loss is commonly used for medical image problems. Also, the volumes of some structures are smaller than that of the others. The cerebellum is bigger than the brain stem and the fourth ventricle. Therefore, a loss that can deal well with multiclass imbalance, as the Lovász-Softmax one, is used.

#### Cross-Entropy

The cross-entropy function comes from the field of information theory. It is a measure of difference between two probability distributions [43]. It reflects the total entropy between the two distributions and is given by Equation 2.11.

$$CE(P, Q) = -\sum_{x \in X} P(x) log(Q(x)) \tag{2.11}$$

where $P, Q$ are probability distributions and and $X$ is a set of random variables.

The cross-entropy can be used as a loss function in classification problems, in order to optimize the models, such as neural networks. In the classification setting, each example in the dataset ($D$) has a class label ($g(x)$). The model predicts the probability ($p(x)$) for each example to belong to a class. Therefore, the binary cross-entropy loss (where $c = 1$) can be written as:

$$ceLoss_c(g, p) = -\sum_x g(x)log(p(x)) \tag{2.12}$$

For the multi-class classification setting, it is possible to do the one-hot encoding of every label of the dataset, Thus, multi-class cross-entropy can be represented by Equation 2.13.

$$ce_{loss} = \sum_{c \in \mathcal{C}} ceLoss_c(g, p) \tag{2.13}$$

where $\mathcal{C} = \{0, 1, 2, ..., n\}$ are the classes in the classification problem.

It is possible to interpret the semantic segmentation task as a multi-label classification problem, in which the aim is to classify every pixel (or voxel) in the image. Therefore, the cross-entropy loss function can be used to optimize semantic segmentation algorithms. This is done in the original training of the LiviaNet by [11].

**Dice-Based Score**

In the semantic segmentation case, the dice score is a metric that evaluates the degree of agreement between the prediction and the ground truth. Then, let $P \in [0, 1]^N$ be the labelmap predicted by a model for an image and $G \in \{0, 1\}^N$, the ground truth segmentation. The dice score between $P$ and $G$ is given by Equation 2.14.

$$Dice(P, G) = \frac{2|P \cap G|}{|P| + |G|} \tag{2.14}$$

where $|.|$ indicates the cardinality of the set.

In [37], Milletari *et al.* presents a continuous extension of this function (Equation 2.15). This is necessary in order to derive a loss function that can be used in the training of neural networks.

$$Dice_{cont}(P, G) = \frac{2\sum_i p_i g_i}{\sum_i p_i^2 + \sum_i g_i^2} \tag{2.15}$$

with $P, Q \in \mathbb{R}^N$. This function can be differentiated w.r.t. every voxel $p_j \in P$ according to Equation 2.16.

$$\frac{\partial Dice_{cont}}{\partial p_j} = 2\frac{g_j(\sum_i p_i^2 + \sum_i g_i^2) - 2p_j(\sum_i p_i g_i)}{(\sum_i p_i^2 + \sum_i g_i^2)^2} \tag{2.16}$$

Then, the loss function based on the dice score can be derived from Equation 2.15. Let $P \in \mathbb{R}^N$ be the output of a model, and $G \in \{0, 1\}^N$ be the reference ground-truth. The soft-dice loss between the two images is given by Equation 2.17.

$$Dice_{loss}(P, G) = 1 - Dice_{cont}(P, G) \tag{2.17}$$

For multiclass semantic segmentation the total dice loss can be calculated by the average between the dice loss for each class (Equation 2.18).

$$Dice_{multi}(P,G) = \frac{\sum_{c\in\mathcal{C}} Dice(P_c,G_c)}{|C|} \tag{2.18}$$

where $P_c, G_c$ are the prediction map and ground truth for class $c$ and $\mathcal{C}$ is the set of labels for each class in the segmentation task.

**Lovász**

The intersection over union (IoU, also known as Jaccard index) is a common metric used to evaluate the quality of the prediction provided by segmentation models. Let then $P, G \in \{0,1\}^N$ be the prediction and segmentation maps. Then, the Jaccard index [25] between $P, G$ for class $c$ is given by:

$$Jaccard_c(P,G) = \frac{\{P=c\} \cap \{G=c\}}{\{P=c\} \cup \{G=c\}} \tag{2.19}$$

Equation 2.19, can also be rewritten in terms of false and true positives (FP and TP, respectively), and false and true negatives (FN, TN, respectively) for class $c$:

$$Jaccard_c(P,G) = \frac{|TP_c|}{|TP_c| + |FN_c| + |FP_c|} \tag{2.20}$$

Note that $Jaccard_c(P,G) \in [0,1]$. Then, the total Jaccard index (Equation 2.21) is given by the average of the score obtained by each class $c \in \mathcal{C}$, where $\mathcal{C}$ is the set all classes to be identified in the images.

$$Jaccard_{total}(P,G) = \frac{\sum_{c\in\mathcal{C}} Jaccard_c(P,G)}{|\mathcal{C}|} \tag{2.21}$$

A corresponding loss function (Equation 2.22) for $P, G \in \{0,1\}^N$ can be extracted from the previous equations.

$$\Delta_{Jaccard_c} = 1 - Jaccard_c(P,G) \tag{2.22}$$

Observe that such equations are designed for binary sets. However, during training, the CNNs outputs are typically probability maps, which belong to the set $[0,1]^N$ (real numbers) not in $\{0,1\}^N$. Therefore, in order to use the Jaccard index as a loss function to train CNNs, it is necessary to use its continuous extension.

The authors in [5] provide this extension by using the property that the $\Delta_{Jaccard_c}$ is submodular [63]. Consider the set $M_c(P,G) = \{G = c, P \neq c\}$ of mispredictions. Equation 2.22 can be rewritten in terms of $M_c(P,G)$ as:

$$\Delta_{Jaccard_c} : M_c \in \{0,1\}^N \mapsto \frac{|M_c|}{|\{G=c\} \cup M_c|} \tag{2.23}$$

In a continuous optimization scheme, it is necessary to be able to attribute a loss value to any vector $m \in \mathbb{R}^N$. Thus, since $\Delta_{Jaccard_c}$ is submodular, then its continuous extension

is given by its convex closure, which is computable in polynomial time and is given by its Lovász Extension (Equation 2.3.6).

The Lovász extension of a set function $\Delta : \{0,1\}^N \mapsto \mathbb{R}$, such that $\Delta(0) = 0,$ is defined by :

$$\bar{\Delta} = m \in \mathbb{R}^N \mapsto \sum_{i=1}^{N} m_i g_i(m),$$

$$\text{with } g_i(m) = \Delta(\{\pi_1, \pi_2, ..., \pi_i\}) - \Delta(\{\pi_1, \pi_2, ..., \pi_{i-1}\})$$

$\pi$ is a permutation ordering the components of $m$ in decreasing order: $x_{\pi_1} \geq x_{\pi_2} \geq ... \geq x_{\pi_N}$

The final loss for multi-class semantic segmentation is given by the following scheme:

1. Use the softmax operation to map the scores of the model to probability distributions;
2. Construct the vector of mis-segmentations $m(c)$ for class $c \in \mathcal{C}$:

$$m_i(c) = \begin{cases} 1 - f_i(c), \text{ if } G_i = c \\ f_i(c), \text{ otherwise.} \end{cases}$$

   where, $f_i(c)$ is the $i$-th voxel in the prediction volume to class $c$.

3. Use m(c) to construct the loss surrogate to $\Delta_{Jaccard_c}$:

$$loss(f(c)) = \bar{\Delta}_{Jaccard_c}(m(c))$$

4. Calculate the Lovász-Softmax (LS) loss:

$$loss(f) = \frac{1}{|C|} \sum_{c \in \mathcal{C}} \bar{\Delta}_{Jaccard_c}(m(c))$$

The study performed by [5] showed that CNNs trained with the Lovász-Softmax (LS) loss exhibited better segmentation results than the ones trained with cross-entropy. Also, the LS function is able to deal well with multi-class imbalance. Therefore, it is an interesting loss function to be considered on this project.

## 2.4 MRI Semantic Segmentation

Semantic segmentation consists in attributing a label to each pixel (or voxel) in the image (or volume) [15]. Atlas-based and Deep Learning are automatic techniques able to perform such task. The latter are the convolutional neural networks based methods, which have been successful in the task of image classification and segmentation for both natural and medical images. The former are methods based on the registration of the target volume to one or a set of template images. Then, the labels are transferred to the target sequence in order to obtain its segmentation. The atlas-based methods have disadvantages as the time

spent to register the images. which is typically long, and they tend to fail for not healthy patients or subjects with different anatomy, when such cases are not covered by the template set (the variability of the subjects is limited by that of the template set). In the following subsections, some atlas-based and CNN methods for MRI segmentation are reviewed.

### 2.4.1 Atlas-Based Approaches

The techniques that use an atlas –a pre-segmented image– as guidance for segmenting brain MRI are called atlas-based [10]. These images may be useful in helping to distinct areas with low contrast and identify different tissues that have similar intensities. Given the lack of existing atlas for neonates and small children, most studies create their own version of an atlas for such images. Below, some methods that use an atlas to segment MRI brain images of neonates and infants are described.

In Anbeek *et al.* [2], a coordinate system called "average brain" is developed. First, a collection of neonate brain MRI is manually annotated, and then iteratively registered in order to create the mentioned system. The volume of interest is registered to this atlas and then a K-Nearest-Neighbors (k-NN) classifier, using as features the voxels intensity and coordinate, provides a probabilistic segmentation map. Lastly, the average brain is registered back to the volume of interest providing a segmentation for this latter.

Another example can be found in Weisenfeld *et al.* [59]. This method also uses a set of manually annotated images (that they call *templates*), which are associated with a set of intensity value samples for each class (called *prototypes*) of the segmentation. A probabilistic segmentation map is obtained for a volume of interest by registering it with the templates. This probabilistic map is iteratively refined by using the prototypes to eliminate errors. This is repeated until the segmentation obtained converges.

A cortical reconstruction is proposed in Xue *et al.* [60]. In order to achieve this result, the brain and other deep tissues are extracted by using atlas-based label propagation. The voxel classification is performed by an Expectation-Maximization (EM) algorithm [55]. Furthermore, Markov Random Fields (MRFs) are used to deal with partial volume and spatial heterogeneity issues.

IMAPA, a method based on the use of the multi-atlas framework, is presented on [52]. A linear combination of the image to be segmented $I$ and a set $E$ of $n$ annotated images (multi-atlas) is used in order to obtain the segmentation $S$ of $I$. This combination is obtained from weights calculated between patches of image $I$ and patches of each example from $E$. Such weights are updated in an iterative way, which aims to minimize an energy function that calculates the distance among the patches centered in a given voxel $x$. This approach was trained on data from the Developing Human Connectome Project (dHCP) and obtained state-of-the-art results, when compared to other atlas based techniques.

In the method presented by Ceschin *et al.* [9] the brain of neonates is segmented by registrating the input volume to ALBERT neonatal parcellation dataset [19, 20]. Then, the cerebellum is extracted from all the volumes and the binary labelmaps are registered into a standard system. Afterwards, these images are used as input to train a 3D CNN for the classification of the structure into dysplastic or not. This classification is important in

order to find malformation in neonates with congenital heart disease. The SLANT method proprosed by Huo *et al.* [23] combines multi-atlas and FCNs in order to perform whole brain 3D segmentation.

Due to the fast development of neonate and small children brain structures and their great variability, it is hard to use atlas based methods for this kind of image. So, some techniques that do not make use of atlas during the segmentation process were developed.

### 2.4.2 Machine Learning Approaches

With the increasing success of Neural Networks, some methods based on Convolutional Neural Networks have been developed to tackle the medical images segmentation task. Below some of them are briefly introduced.

The U-Net [44] has an encoder-decoder-like architecture. That is, its first layers extract features from the image by performing a series of convolutions with stride 2 (downsampling), which means the image size is reduced by a factor of 2 every time the operation is applied. Then, the last layers perform transpose convolution (upsampling) to grow the image resolution back to its original size. So, a softmax layer provides the probability for each pixel of being part of the background or the foreground. This network was tested for the segmentation of both 2D neuronal structures and cell images of microscopic biomedical images.

Another convolutional network called V-Net was developed to segment volumetric MRI [37]. It was designed to perform the semantic segmentation of lower abdominal MRIs, in order to identify the prostate from the background (that is, the rest of the volume, which is not the object of interest). It was the first end-to-end fully-convolutional network (FCN) developed to segment 3D images. Before, 3D images that were processed by neural networks had their slices analyzed one by one (since each slice is a 2D image) by 2D CNNs. This approach is also called "2.5D representation" [46]. When it participated on the PROMISE2012 challenge [36], V-Net obtained state-of-art results.

Li *et al.* (2017a) [32] also propose an end-to-end network for semantic segmentation of volumetric images. HighRes3dNet is composed of 20 layers and in order to avoid the encoder-decoder architecture, residual connections and dilated convolutions are applied. This approach allows the network to have different receptive fields without having to reduce the size of the feature maps. It was trained and tested for the brain parcellation task, which consists of dividing the brain into 155 structures.

Khalili *et al.* [28] presents a neural network to segment seven brain tissues (including brain stem and cerebellum) from fetal (*in utero*) MRI. These images, as the neonatal ones, commonly suffer from intensity inhomogeneity artifacts due to patient movement during scan. In order to deal with it, this technique performs data augmentation by introducing simulated intensity inhomogeneity images during network training. The network architecture used is U-NET [44], thus 2D segmentation is implemented. The mean Dice coefficient improved from 0.3 to 0.8 for the cerebellum segmentation when the network was trained with data augmentation. Such augmentation could be used to replace or complement preprocessing techniques that are typically used to deal with intensity inhomogeneity, such as

bias field corrections and volumetric reconstruction, without requiring the acquisition of additional volumes.

In [40], a method based on the use of Generative Adversarial Networks (GANs) is proposed in order to perform both the super-resolution reconstruction and cortical gray matter segmentation of neonatal brain MRI volumes. The GAN used is a 3D end-to-end one. The generative network (GN) is responsible for the reconstruction and segmentation, while the adversarial has the role of discriminating between simulated and real data from the GN output. The network was trained on simulated low resolution (LR) images from dHCP and was tested both for this dataset and real LR data. When compared to methods that perform super-resolution reconstruction, such as cubic spline interpolation, this network obtain more realistic results rather than oversmoothed ones. For the segmentation analysis, it was compared to IMAPA and performed better for the LR images.

A CNN is presented in Moeskops *et al.* [38] to perform the semantic segmentation of brain MRI. This network architecture is based on a multi-scale approach. It is composed of branches, and each branch is trained on different sized 2D patches of the images. The network is separately trained on data obtained from different acquisition protocols and ages. There are five datasets used to train and test the network, 3 are composed of neonates images –T2w, coronal view (30 weeks and 40 weeks) and axial view (40 weeks)–, one of ageing adults volumes (T1w axial view, 70 years), and of young adults (T1w coronal view, 23 years). It is capable of obtaining accurate mean dice results for every brain tissue segmented, except for myelinated white matter for neonates data.

Rosati *et al.* [45] combines the Region Growing and K-Means Clustering methods in order to obtain the brain tissue segmentation –which includes the white matter, the grey matter and the cerebrospinal fluid (CSF) – of neonates T1-weighted MRI exams.

LiviaNet [11] is a 3D patch fully-convolutional neural network (FCN) developed to perform subcortical structures parcellation. The network is composed of thirteen layers (9 convolutional, 3 fully-connected and a classification one). It was trained and tested on the Internet Brain Segmentation Repository (IBSR[1]), which contains 18 T1-weighted MRI images. The model obtained state-of-the-art performance in the segmentation of both the right and left sides of the pallidium, the thalamus, the caudate and the putamen.

The atlas-based methods for semantic segmentation lacks generalization ability and have high time requirements due to the registration processes needed. Among the CNNs methods presented, LiviaNet was chosen to be used in our pipeline, because of its 3D approach. Also it was the best one to delineate the whole cerebellum structure on the study carried out by [8], to compare automatic methods performance in the parcellation of the cerebellum. Based on this result and its ability to deal with 3D images, the network was the chosen one to be trained and applied to the data used in this work. More details will be given in Section 3.2.2.

[1] `http://www.cma.mgh.harvard.edu/ibsr`

## 2.5 Supersegmentation-based approaches

To reduce memory requirements and computational costs of some image analysis algorithms, different techniques use a superpixel representation (or supersegmentation) of the image as input. Such representation corresponds to a partition of the original image into regions (superpixels) that are homogeneous regarding some aspects, such as color, surface and texture [1]. A desirable characteristic of these methods is that the generated superpixels should adhere to original image boundaries.

An approach based on such methods was studied in the beginning of this Master's project. However, the patch based technique described in Section 3.2.2 presented better and more promising results at that moment. Thus the results of the supersegmentation techniques are shown in Section 4.1.1. Below we do a bibliographic review of the topic.

Supersegmentation techniques can be categorized as graph-based and gradient-ascent-based. In the first category are the methods that treat each pixel in the image as a vertex in a graph and adjacent pixels have weighted edges connecting them. These weights represent the similarity between the nodes. The superpixels are generated by merging nodes which is accomplished by minimizing a function defined on the graph. The gradient-ascent ones are iterative. They start from an initial pixel clustering and refine it until a certain criteria is met.

In Zhou *et al.* [64], a graph neural network (GNN) is defined as a way of extending neural networks for processing the graph-structured data. Existing CNN are designed for Euclidean data – pixels (or voxels) distributed in a rigidly defined and ordered grid. Therefore, they are not capable of dealing with graph data, which is non-Euclidean, properly. For example, some operations of CNNs (for instance, convolution and pooling ones) need the data information to be ordered and have a fixed size, however graph nodes are unordered and have a variable number of neighbors. Also, CNNs do not make direct use of nodes dependency information, they only incorporate it as a feature of each node. To circumvent these drawbacks, GNNs are defined in a way that its operations are calculated by propagating on each node (therefore, making their output invariant for node order), and are guided by graph structure (depending on node's neighborhood values).

In Landrieu *et al.* [30], it is also presented a method for the semantic segmentation of 3D clouds. In order to perform this task, the image is first partitioned into simple geometric shapes called "superpoints", which is achieved by finding the minimum arguments of the energy function defined in [21]. Then, from this partition, a graph is built (called Super Point Graph - SPG), by connecting nearby super points. After this stage, the nodes of the graph are embedded in order to obtain a compact representation for them, which is done using the PointNet [42]. Finally, a Gated Graph Neural Network is used in order to obtain the label for each super point, using its embedded representation.

The superpixel techniques can be combined with the graph neural nets (GNNs) in order to obtain an image's (or volume's) semantic segmentation, similarly to what is done in the previous cited work by Landrieu *et al.* [30]. By seeing each region of the supersegmentation as a vertex in a graph, where neighbor regions are connected by an edge, it is possible to

use a node classification GNN in order to attribute a label to each vertex. Given all this, in the beginning of this project, we tested some superpixel segmentation methods in order to assess how appropriate these techniques would be for our problem. Below, we detail some commonly used supersegmentation algorithms and show the results we obtained by applying them to our images.

### 2.5.1 Watershed

Watershed is a well defined concept in the topographic area. It is a elevated border between two (hydrographic) catchment basins. These basins are defined by lower spots to which the water flows. By treating images as topographic reliefs, it is possible to think about water flowing in the image and apply some definitions and techniques to obtain their segmentation. First, it is important to define some terms for the 2D gray-scale image space ($\mathbb{Z}^2$) [56].

**Definition 2.5.1.** ***(Path [56])*** *A path of $P$ of length $l$ between $p$ and $q$ in and image $I$ is a sequence of pixels $(p_0, p_1, ..., p_l)$, such that $p_0 = p$ and $p_l = q$. Denote the length of a path $P$ by $l(P)$.*

**Definition 2.5.2.** ***(Minimum [56])*** *A minimum $M$ is a connected set of pixels, which are iso-intensive and have a lower elevation (that is, lower intensity, darker pixels) than surrounding pixels.*

$$\forall p \in M, \forall q \notin M, \text{ such that } I(p) \leq I(q)$$

$$\forall \text{ path } P = (p_0 = p, p_1, ..., p_l = q) \text{ between } p \text{ and } q$$

$$\exists i \in [1, l], \text{ such that } I(p_i) > I(p_0)$$

**Definition 2.5.3.** ***(Catchment basins [56])*** *A catchment basin is a region defined by a minimum $M$. All pixels $p \in I$ from which the (simulated) water flows and reaches the minimum $M$ belong to the catchment basin of $M$.*

**Definition 2.5.4.** ***(Watershed [56])*** *The watershed can be defined as a set of paths of connected pixels separating two catchment basins. It is the region where water from two catchment basins meet.*

The immersion simulation is given by supposing that there is a hole on each regional minimums $M$ of $I$ and the image (seen as surface due the altitude values of each pixel) is gradually immersed on water. The catchment basins of $I$ begin to fill and the pixels, where water from different catchment basin meet, are called dams. At the end, each minimum is surrounded by dams, which are the watershed lines.

Let $I$ be the image of interest and $h_{min}, h_{max}$ be, respectively, the smallest and largest values of $I$. Also, let the threshold of $I$ at level $h$ be $T_h(I) = p \in I, I(p) \leq h$, where $I(p)$ is the intensity of pixel at location $p$. Let's also define the concepts of geodesic distance (Definition 2.5.5) and geodesic influence zone (Definition 2.5.6).

**Definition 2.5.5.** *(**Geodesic distance [56]**) The geodesic distance between two pixel $x$ and $y$ in a set $A \subseteq I$ is the infimum of the length of the paths between $x$ and $y$ that are completely contained in $A$.*

$$d_A(x, y) = \inf\{l(p),\ P \text{ path between } x, y \text{ which is totally included in } A.\} \tag{2.24}$$

**Definition 2.5.6.** *(**Geodesic influence zone [56]**) Let $B \subset A$, such that $B = \{B_1, B_2, ..., B_k\}$, where each $B_i$ is a connected component. The geodesic influence zone of a connected component $B_i$ of $A$ is the location of points of $A$ whose geodesic distance to $B_i$ is smaller than their geodesic distances to any other component of $B$.*

$$iz_A(B_i) = \{p \in A, \forall j \in [1, k] \setminus \{i\}, d_A(p, B_i) < d_A(p, B_j)\} \tag{2.25}$$

The set of points in $A$ that are not part of any geodesic influence zone are the skeleton by influence zones (SKIZ) of $B$ in $A$. Then $SKIZ_A(B) = A \setminus IZ_A(B)$, with $IZ_A(B) = \bigcup_{i\in[1,k]} iz_A(B_i)$. Let $X_{h_{min}} = T_{h_{min}}(I)$ the set of minima points with lowest altitude. The catchment basins of $I$ can be found by implementing the following recursion:

1. a) $X_{h_min} = T_{h_{min}}(I)$.
2. b) $\forall h \in [h_{min}, h_{max} - 1], X_{h+1} = \min_{h+1} \bigcup IZ_(T_{h+1}(I))$

The catchment basins will be given by the final set $X_{h_{max}}$, and therefore, the watershed lines are the complement of this set in $I$, $I \setminus X_{h_{max}}$.

The algorithm to implement such recursion is based on breadth-first search of the pixels associated with each threshold level of the image intensities $T_h$. The first step corresponds to sorting the pixels locations in increasing order of their corresponding intensities. Then, the influence zone of each $T_h$ are computed to find the catchment basins. The following scheme was adapted from [56, 6].

1. All pixels at level $h_{min}$ receive a unique label each one. An empty priority queue ($PQ$) is created.
2. The pixels that are part of the neighborhood of each pixel in item (1) are inserted in a priority queue ($PQ$), whose priority function is the magnitude of the gradient of the pixels.
3. The highest priority pixel $p$ is extracted from $PQ$. The pixel $p$ receives the same label as its neighbors, if all of them have the same label. All the neighbors $q$ of $p$ that do not have a label are put into $PQ$ (if $q \notin PQ$).
4. Repeat item (3) until $PQ$ is empty. The pixels that are unlabelled are the watershed lines.

Since this algorithm usually results in an oversegmentation of the image, in item (1) some pixels that are known by markers, receive a label. The markers can either be user-defined

or set automatically. Commonly, a set of markers are local minima of the gradient of the image. This algorithm has complexity $\mathcal{O}(n)$, where $n$ is the number of pixels in the image. And it does not provide mush control over the compactness of the superpixels obtained.

### 2.5.2 Simple Linear Interactive Clustering

Simple Linear Interactive Clustering (SLIC) [1] is a method developed to obtain the super-segmentation of an image. It is based on the K-Means algorithm, with some modifications that makes it efficient to be applied even to large images.

First, a set of $k$ seed points are randomly chosen in a grid of $S \times S$ ($S$ is $\sqrt{\frac{N}{k}}$, where $N$ is the number of pixels on the image) equally-spaced pixels of the image. It is assumed that the image is represented in the CIELAB color space, therefore, the points are $[l_i, a_i, b_i, x_i, y_i]$ for $i \in \{1, ..., k\}$, where $(l_i, a_i, b_i)$ are the color components, and $(x_i, y_i)$ are the pixel coordinates.

Then, each pixel of the image is assigned to the same cluster as the nearest center within a $2S \times 2S$ neighborhood. This limitation on the search space of the nearest center reduces the computational cost of the algorithm when compared to classic K-Means. Also, the distance between a pixel $j$ and a cluster center $C_i$ used is a combination of the distance between the color ($d_c$, Equation 2.27) of the pixels and the one between their coordinates ($d_s$, Equation 2.28). It is given by Equation 2.26.

$$D = \sqrt{d_c^2 + \left(\frac{d_s}{S}\right)^2 * m^2}, \tag{2.26}$$

where

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2}, \tag{2.27}$$

and

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}. \tag{2.28}$$

$S = \sqrt{\frac{N}{k}}$ is the maximum spatial distance expected within the cluster and $m$ is a compactness constant defined by the user. If it is large, spatial proximity is more relevant, then more compact and regular superpixels are produced. If it is small, color proximity is more important and the resulting superpixels adhere better to image boundaries. It is important to notice that if the image is in the grayscale domain, then the color distance $d_c$ is given by $d_c = \sqrt{(l_j - l_i)^2}$.

Finally, cluster centers are updated by calculating the mean among all the points within the cluster. Thus, each center $c_i$ (of cluster $C_i$) is $\frac{1}{|C_i|}\sum_{c \in C_i} c$, where $|.|$ represents set cardinality. This step and the assignment one are repeated until the error $E = ||c_i^{new} - c_i^{old}||_2$ converges. That is, the difference between the old and new cluster centers is not altered. Then, a postprocessing step is performed in order to assign disjoint pixels to nearest cluster and ensure connectivity.

Due to the limited search space on the assignment step, less distance calculations are carried out. Therefore, the complexity of this algorithm is $\mathcal{O}(N)$.

### 2.5.3 Normalized Cuts

In this method, first presented in [49], a graph $G = (V, E)$ is extracted from the image. The nodes are the pixels and there is a weighted edge between each node, therefore, the graph is complete. Also, the weight on each edge is given by Equation 2.29.

$$w_(i,j) = e^{\frac{-||F(i)-F(j)||_2^2}{\sigma_I}} * \begin{cases} e^{\frac{-||X(i)-X(j)||_2^2}{\sigma_X}}, & \text{if } ||X(i)-X(j)||_2 < r, \\ 0, & \text{otherwise.} \end{cases} \tag{2.29}$$

where, $X(i)$ are the coordinates of node $i$ in the image, and $F(i)$ is a feature vector. In the case of gray-scale images, $F(i) = I(i)$, the intensity value of the pixel represented by node $i$.

A desirable partition $\{V_1, V_2..., V_m\}$ of a graph $G$ is the one that yields $V_i$ with high intra-similarity and each pair $V_i, V_j, i \neq j$ with low similarity. Then, let $A, B$ be a partition such that $A \cup B = V$ and $A \cap B = \emptyset$, that was obtained by removing edges that connected $A$ and $B$. So, the degree of disassociation of this partition can be written as in Equation 2.30.

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v). \tag{2.30}$$

In order to avoid some undesirable effects that can result from minimizing the Equation 2.30, such as producing partitions where a part is composed of only one element, the normalized-cuts measure have been proposed and is presented on Equation 2.31.

$$N_{cut}(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)}. \tag{2.31}$$

where $assoc(A, V) = \sum_{u \in A, v \in V} w(u, t)$, that is the sum of all edges from $A$ to the nodes of $G$.

The problem of minimizing Equation 2.31 is NP-complete. Therefore, an approximated solution has been proposed. Let $d(i) = \sum_j w(i, j)$, for all node $i \in V$, and $D$ is a diagonal matrix, with $D(i, i) = d(i)$. Also, define $W \in \mathbb{R}^{N \times N}$ a symmetrical matrix with $W(i, j) = w(i, j)$. Then, the graph cut algorithm can be described as follows:

1. From the input image, create $G(V, E)$, the matrices $D$ and $W$.
2. Solve $(D-W)x = \lambda Dx$ to find the eigenvectors with the smallest associated eigenvalues.
3. Use the eigenvector associated with the second smallest eigenvalue to bipartition the graph.
4. Recursively repeat for the parts of the partition, if necessary.

The eigenvector obtained in the item 3 can be used to extract the partition in various ways. The most common ones is deciding that either the median value or zero will be the threshold. The computational complexity of this algorithm is $\mathcal{O}(nm)$, where $m$ is the number of steps required to the Lanczos [29] method to converge (the method to calculate

the eigenvectors and eigenvalues), and $n$ is the number of pixels in the image. In [31], a more efficient implementation is shown and it is said that the complexity can be $\mathcal{O}(n^{\frac{3}{2}})$.

### 2.5.4 Efficient Graph-based Felzenszwalb method

Due to the limiting time complexity of the graph-based approaches for obtaining the superpixel representation of images, Felzenszwalb *et al.* [13], created a more efficient method.

From an image, a graph $G = (V, E)$ is created. Let $|V| = n$, where $n$ is the number of pixels in the image, and $|E| = m$, then $m = \mathcal{O}(n)$. Each pixel $p_i$ corresponds to a vertex $v_i \in V$, and the edge set $E$ is formed by connecting the pixels that are in an eight-connected neighborhood. Each edge of the graph has an associated weight given by $w(v_i, v_j) = |I(p_i) - I(p_j)|$, where $I(p_i)$ is the intensity of pixel $p_i$.

The aim of the method is to find a partition $S = \{C_1, C_2, ..., C_r\}$ of V, such that elements within a part $C_i$ are similar and elements in $C_i, C_j$, with $i \neq j$, are dissimilar. That is, the weights of edges within $C_i$ are small and between $C_i, C_j$ are large.

In order to obtain such a partition, a predicate $D$ is defined in order to decide if there is a boundary between two components in $S$ or not. First, let the internal difference of a component $C \in V$ be the largest weight in the Minimum Spanning Tree (MST) of $C$ defined as in Equation 2.32. Then, minimum internal difference is defined by Equation 2.33.

$$Int(C) = \max_{e \in MST(C,E)} w(e) \tag{2.32}$$

$$MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2)) \tag{2.33}$$

where, $\tau$ is a threshold function that controls how much more different from each other two components must be, when compared to their internal differences, in order to consider that there is a boundary between them. The function is given by $\tau(C) = k/|C|$, where $k$ is a user-defined constant.

Also, let the difference between two components $C_1, C_2 \in V$ be the smallest weight of an edge among the edges that connect $C_1$ and $C_2$, as in Equation 2.34.

$$Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j \in E)} w(v_i, v_j) \tag{2.34}$$

Then the predicate $D$ is defined by Equation 2.35.

$$D(C_1, C_2) = \begin{cases} \text{true}, & \text{if } Dif(C_1, C_2) > MInt(C_1, C_2) \\ \text{false}, & \text{otherwise.} \end{cases} \tag{2.35}$$

Therefore, a greedy algorithm can be devised using the predicate $D$ in order to obtain the segmentation $S$. The following algorithm scheme was extracted from [13].

1. From an image, create the graph $G = (V, E)$ as described above.

2. Sort the edges of $E$ by non-decreasing edge weight, obtaining $\pi = \{o_1, o_2, ..., o_m\}$.

3. Start with segmentation $S^0$, where each vertex $v_i$ is in its own component.

4. Repeat step 5 for $q = 1, 2, ..., m$.

5. Construct $S^q$ given $S^{q-1}$ as follows. Let $v_i$ and $v_j$ denote vertices connected by the $q - th$ edge in the ordering, that is $o_q = (v_i, v_j)$. Also, let $C_i^{q-1} \in S^{q-1}$, such that $v_i \in C_i^{q-1}$, and $C_j^{q-1} \in S^{q-1}$, such that $v_j \in C_j^{q-1}$. Then, if $C_i^{q-1} \neq C_j^{q-1}$ and $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, then $S^q$ is obtained from $S^{q-1}$ by merging $C_i^{q-1}$ and $C_j^{q-1}$, otherwise $S^q = S^{q-1}$.

6. Return $S = S^m$.

The compactness of the superpixels produced by this segmentation can be controlled by parameter $k$ from the $\tau(C)$ function. The larger this number, larger the components found will tend to be. Moreover, the complexity of this algorithm is $\mathcal{O}(nlogn)$, due to the time spent to order the edges by their weights.

# Chapter 3

# Material and Methods

## 3.1 Data

Tasks involving medical images, specially the volumetric ones such as MRI, often suffer from data shortage. Some reasons are: the lack of public datasets due to patient anonymity issues and the difficulty to obtain the ground truth images, since they have to be manually produced. However, there are some public datasets for brain segmentation, such as ISeg [57], which contains T1- and T2-weighted brain MRI of infants ranging from 2 weeks to 12 months, and BraTS [35], that is composed of multimodal MRIs for brain tumor segmentation in adults. Due to the specific characteristics of these datasets (subjects' ages and structures annotated), they are not suitable for the task proposed in this work. Therefore, we built our own dataset. The acquisition and annotation will be explained below.

The data was provided by the Doctors Marcelo Straus Takahashi and Suely Ferraciolli from the Children's Institute from Hospital das Clínicas (ICr-HC). The dataset consists of 32 T2-weighted MRI obtained from a 1.5T Philips Ingenia machine. All the volumes had their ROI (fourth ventricle, brain stem and cerebellum regions) annotated.

### 3.1.1 Dataset formation

The dataset consists of 32 T2-weighted MRI volumes acquired using a 1.5T Philips Ingenia machine from the ICr, 11 of them being obtained in the axial plane and the other 21 in the sagittal plane. In the beginning of the project, only the data from four patients were available. Then, the doctors were able to provide us more exams from seven other patients. Some experiments were conducted using the data from the four first patients (Table 3.1 - A1), and subsequently, we included the other 7 patients exams into the dataset (Table 3.1 - A2, A3). Near the end of this project, the doctors could provide us more 21 annotated volumes, and we were able to perform experiments with them (Table 3.1 - A4).

The age distribution of the data can be seen in Figure 3.1. The ages vary from 0 to 18 years, with a predominance of patients that are 0 to 4 years. This variety of ages is unusual in the literature, because most of the datasets (public or not) are focused in a specific age group, as it is possible to verify in the previously cited ISeg. Performing automatic semantic segmentation on a dataset with such age distribution is challenging, because the

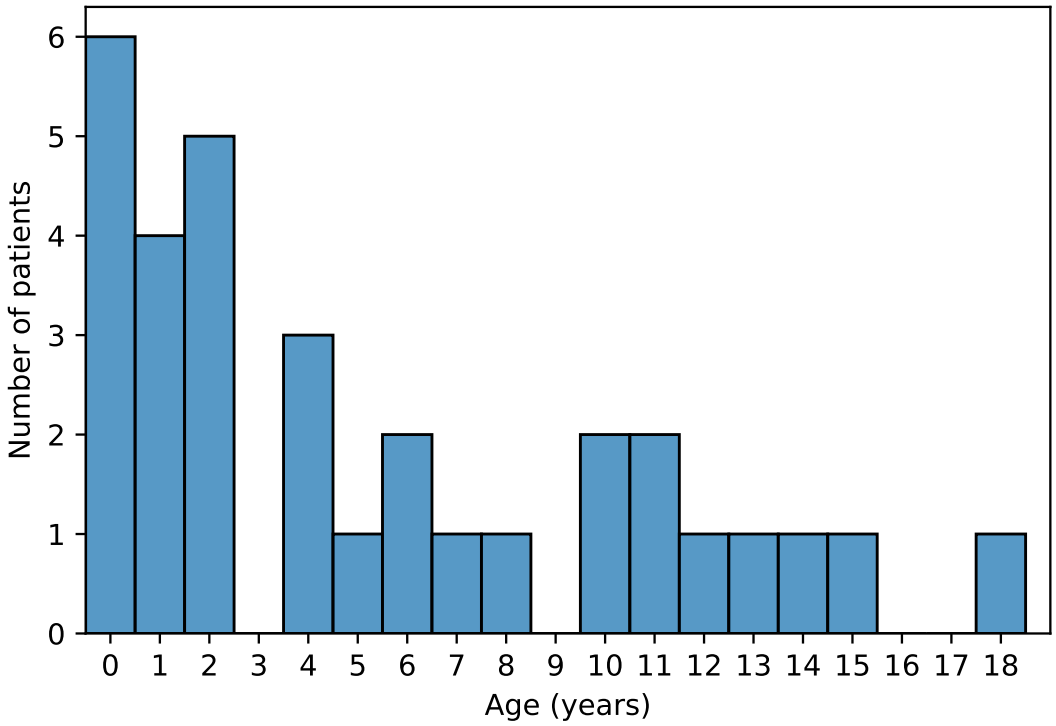



Figure 3.1: Distribution of the ages in the dataset.

brain structures rapidly develop and change during the first years of age of a person (mainly during the first two, when myelination occurs). On the other hand, a method that can deal with such data is important for the doctors in the Children's Institute that have to treat dozens of patients of different ages on a daily basis.

The first 11 volumes were all obtained in the axial plane, which can be seen as the plane that divides the body into superior and inferior parts –thinking about the head MRI case, it has slices from the top of the head to the neck base Figure 3.2 (a). The spacing of the volumes were $0.9 \times 0.9$ in the $x - y$ plane and [1.4, 2.6] in the $z$ axis. All the images had the same dimensions $240 \times 240 \times 70$. The next 21 exams were volumetric images obtained in the sagittal plane, the one that divides the body into left and right parts –again, in the head MRI case, the one that contains slices from one ear to the other Figure 3.2(b). Their spacings varied from $0.5 \times 0.5$ to $0.9 \times 0.9$ in the $x - y$ plane and [0.8, 1.0] in the $z$ axis. Some volumes were $432 \times 432 \times 167$, others $240 \times 240 \times 160$, and $240 \times 240 \times 140$.

Since we already had volumes in the axial plane, we decided to standardize all the volumes to the axial plane view by transposing the volumes of the 21 sagittal samples, an example can be seen in Figure 3.4. This operations consists of permuting the $x, y, z$ axes of the sagittal volume into $y, z, x$ to obtain the axial view. Therefore, the $x$-axis of the axial corresponds to the $y$-axis of the sagittal, $y$-axis corresponds to the $z$-axis, and the $z$-axis to the $x$-axis. Such standardization aims into facilitating the learning by the network. After some experiments, we also verified that the great variance in the dimensions of the volumes hindered the learning by the network. Therefore, we decided on resampling the

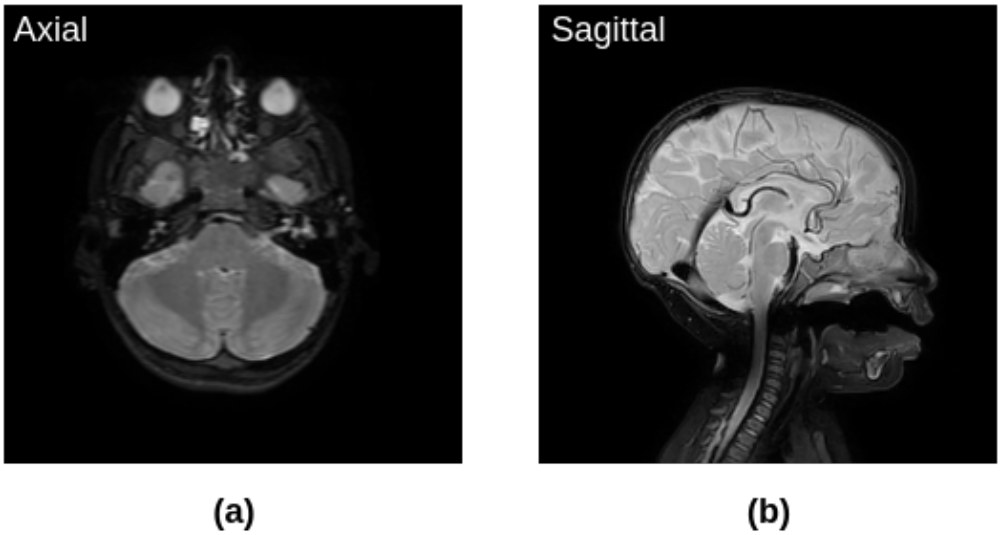



Figure 3.2: Example of an MRI slice in the axial view (a) and another in the sagittal view (b).

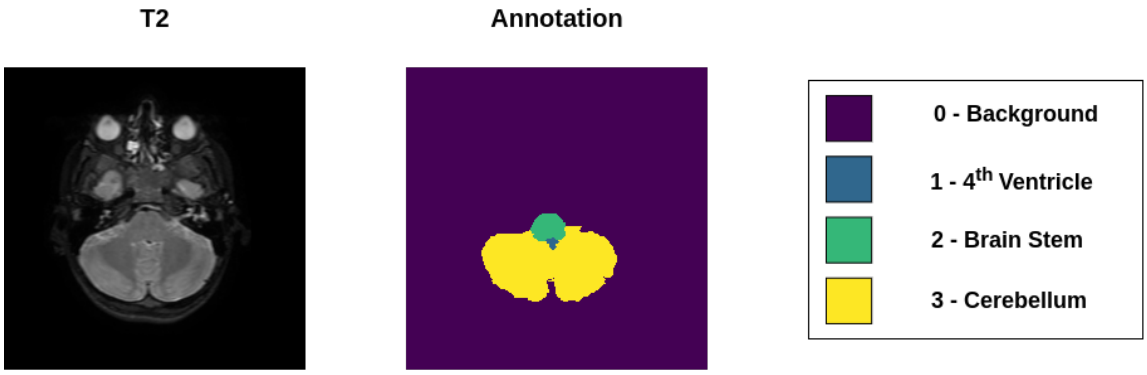



Figure 3.3: Example of an volume slice and the annotation of the $4^{th}$ ventricle, brain stem and cerebellum regions, with their respective numeric labels.

volumes and their annotations, so that they all had the same dimension $240 \times 240 \times 70$. This was implemented using the function *ResampleImageFilter* from SimpleITK library. For the volumes we use a linear interpolation, and the nearest neighbor, for the annotations. This is approximation method was chosen, because it preserves the labels in the ground truth, that is, it does not introduce new labels as the others methods do.

### 3.1.2 Data Annotation

The data were annotated by radiology residents from the ICr-HC, who were trained and supervised by Marcelo Takahashi and Suely Ferracioly (radiology experts). The residents used the Insight Toolkit (ITK[1]) and 3D Slicer[2] softwares to delineate the regions of interest in each slice of the volumes. Then, the experts revised their annotation.

The regions annotated were $4^{th}$ ventricle, the brain stem and cerebellum regions, which were assigned labels 1, 2 and 3, respectively. All the other parts of the volume, which will be referred to as background, were given label 0. An example can be seen in Figure 3.3.

[1]https://itk.org/
[2]https://www.slicer.org/

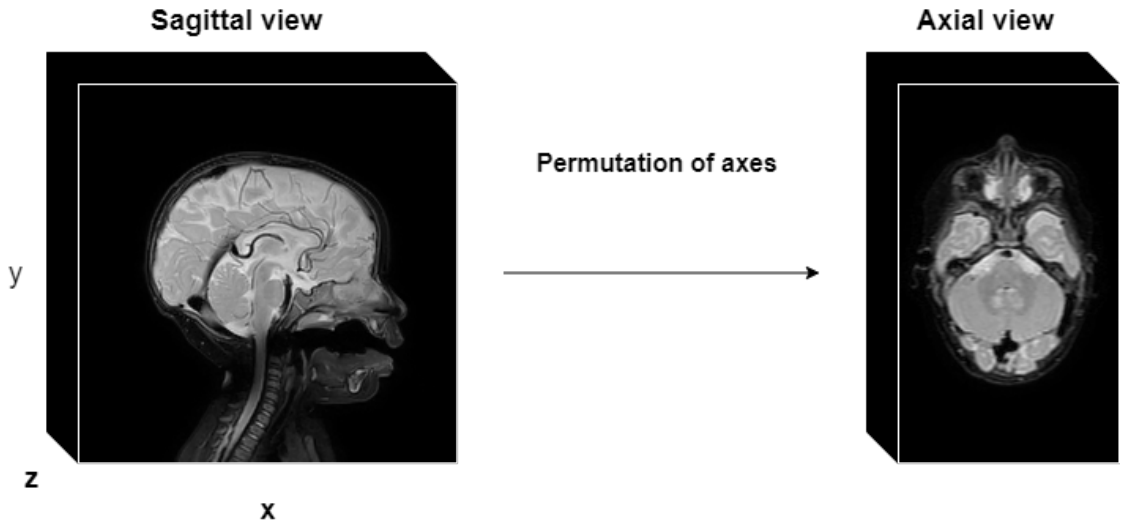



Figure 3.4: The sagittal and axial views of a MRI from the same patient. From the sagittal volume, we can obtain the axial slices by permuting (or transposing) the $x, y, z$ axes into $y, z, x$ axes.

## 3.2 Proposed Approach

Figure 3.5 shows the pipeline developed during this project in order to obtain the semantic segmentation of a dataset of pediatric T2-weighted MRI volumes (Table 3.1). Firstly, the data is preprocessed, by performing skull-stripping and normalization operations. Then, patches of the volume are classified by the LiviaNet model trained to identify the defined region of interest (ROI). Finally, the segmentation obtained is postprocessed to remove mis-segmentations and improve the final result. All these volumes are from the axial view and have dimension $240 \times 240 \times 70$. These steps are detailed in the next sections.

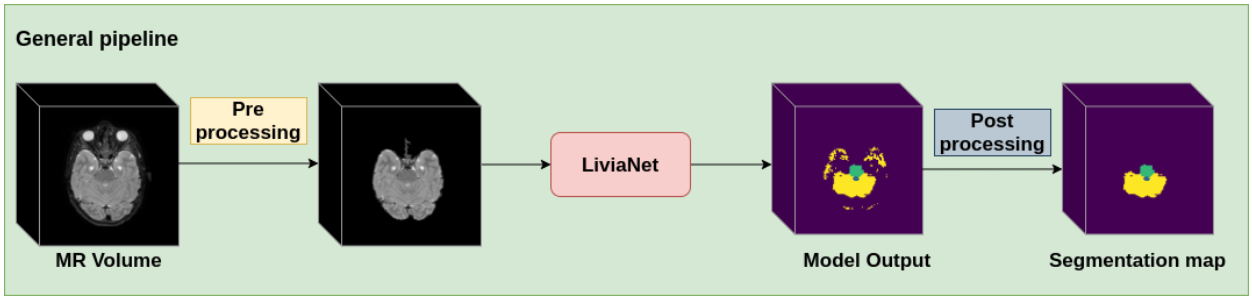



Figure 3.5: Pipeline to obtain the semantic segmentation of MRI data, showing the preprocessing, model application and postprocessing steps.

### 3.2.1 Preprocessing

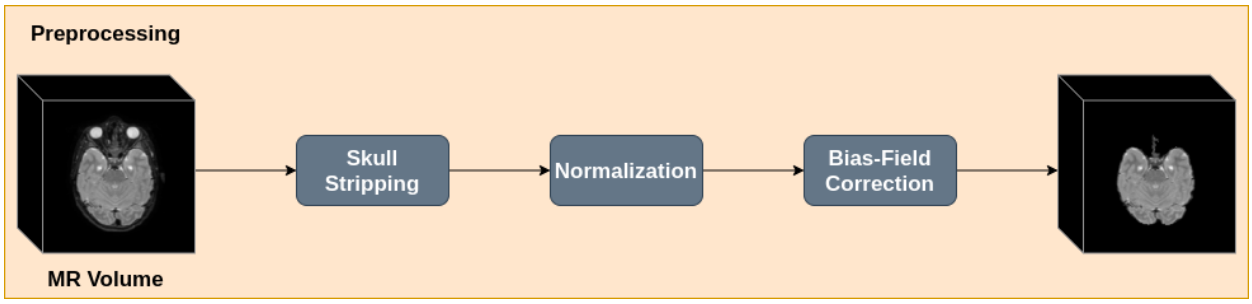



Figure 3.6: The prepocessing pipeline. There three methods applied to the original volume for the removal of non-brain structures, normalization and correction of inhomogeneities in the volume.

Figure 3.6 show the operations involved in the preprocessing of the MRI, which are common when dealing with such kind of data. Also, they are the same applied by [11]. The skull stripping is performed in order to remove the skull and other non-brain tissues present on the images. To achieve it, the Brain Extraction Tool (BET) [26] from Nipype [18] package was used. It is an automatic method, based on thresholding and an iterative surface estimation. In figure 3.7 a result obtained by applying this function is shown. It is possible to notice how the eyes, skull and most of nasal cavity are no longer present in the image. Even though some non brain structure are left in the top of the image, the BET operation is beneficial for simplifying the volume analysis by discarding the extraneous regions.

Then, the output is normalized according to the normalization method in Equation 3.1. That guarantees that each volume will have values in the range $[0, 1]$. This operation is important so that we have all volumes values in the same range, which facilitates the generalization of some methods, such as neural networks for this kind of data.

Finally, the Bias-Field Correction (BFC) [54] is applied to reduce signal inhomogeneity and illumination non-uniformity, that may occur in the images during the exam acquisition and image formation, due to inhomogeneities on the magnetic field. The bias-field is a low-pass noise that affects the image. Figure 3.8 shows the result of the application of the function in the same volume as shown in the previous figure (Fig.3.7). It is not possible to observe visual differences, but we can see some details if we make the difference between both images. Lighter and darker grey regions are the same where there was more noise correction. Both the normalization and BFC help to reduce the intensity and contrast bias.

$$Normalize(V) = \frac{V - V_{min}}{V_{max} - V_{min}} \tag{3.1}$$

where $V, V_{min}, V_{max}$ are, respectively, the volume and the volume's maximum and minimum values.

### 3.2.2 LiviaNet

One of the main challenges regarding the application of 3D CNNs is the memory and computational requirements demanded by such architectures. For this reason, many 2D

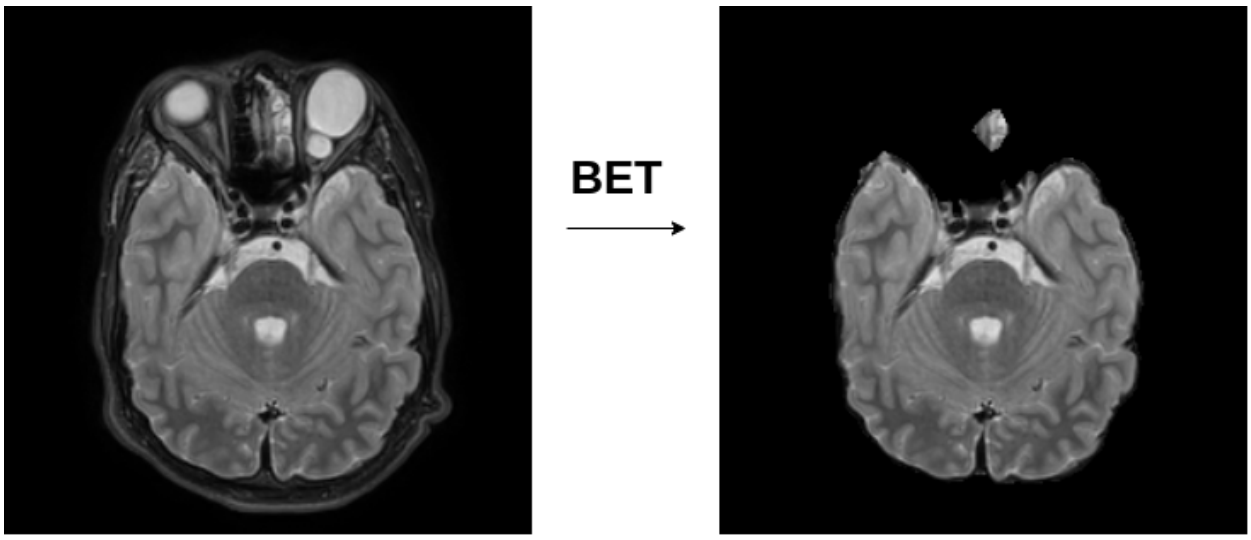



Figure 3.7: Example of the application of the brain extraction operation. On the left, we have a original T2-weighted slice of a volume and on the right, the result of the brain extraction. It is possible to verify that the skull structures and other non-brain structures, such as the eyes, are removed from the image.

methods were developed to 3D data. Such methods can reduce the cited costs by treating each slice of the volume individually as an input to the model. However, one important drawback of such approach is the loss of 3D spatial context.

LiviaNet proposes a 3D patch-based approach in order to deal with such limitations. The input of the network are subvolumes of the MRI, with the size $27 \times 27 \times 27$. Though this technique loses some spatial context, it still preserves more information than the 2D ones, and is less computationally expensive to train than fully 3D methods.

Figure 3.9 shows the network's architecture. The subvolume is fed into the convolutional block. It is composed by 9 layers that use $3 \times 3 \times 3$ convolution kernels, followed by a batch normalization and then, by a PReLU activation functions. Such sequence of operations are responsible for extracting features from the images. The first layers identify simple structures as edges and blobs, while the deeper ones extract more complex information, such as shapes, by combining the features from the previous layers.

Then, the output of layers nine, six and three are concatenated and fed to the fully-connected block. These skip-connections help to propagate the information from earlier layers to the deeper ones. The feature maps from layers three and six have $9 \times 9 \times 9$ cubes cropped around their centers, so that they have the same size as layer 9 output. Such operation also help reduce the computational requirements of the network by reducing the feature maps volume.

Also, in order to maintain the fully convolutional structure of the network, the fully-connected layers are encoded by doing $1 \times 1 \times 1$ convolutions [34, 27]. The fully-connected block is responsible to extract semantic information from the images, by combining the values of all the pixels from the feature maps [34].

Finally, the last layer (or "classification" layer) is the output from the fully-connected block. It is composed by $C$ feature maps $9 \times 9$, where $C$ is the number of classes in the

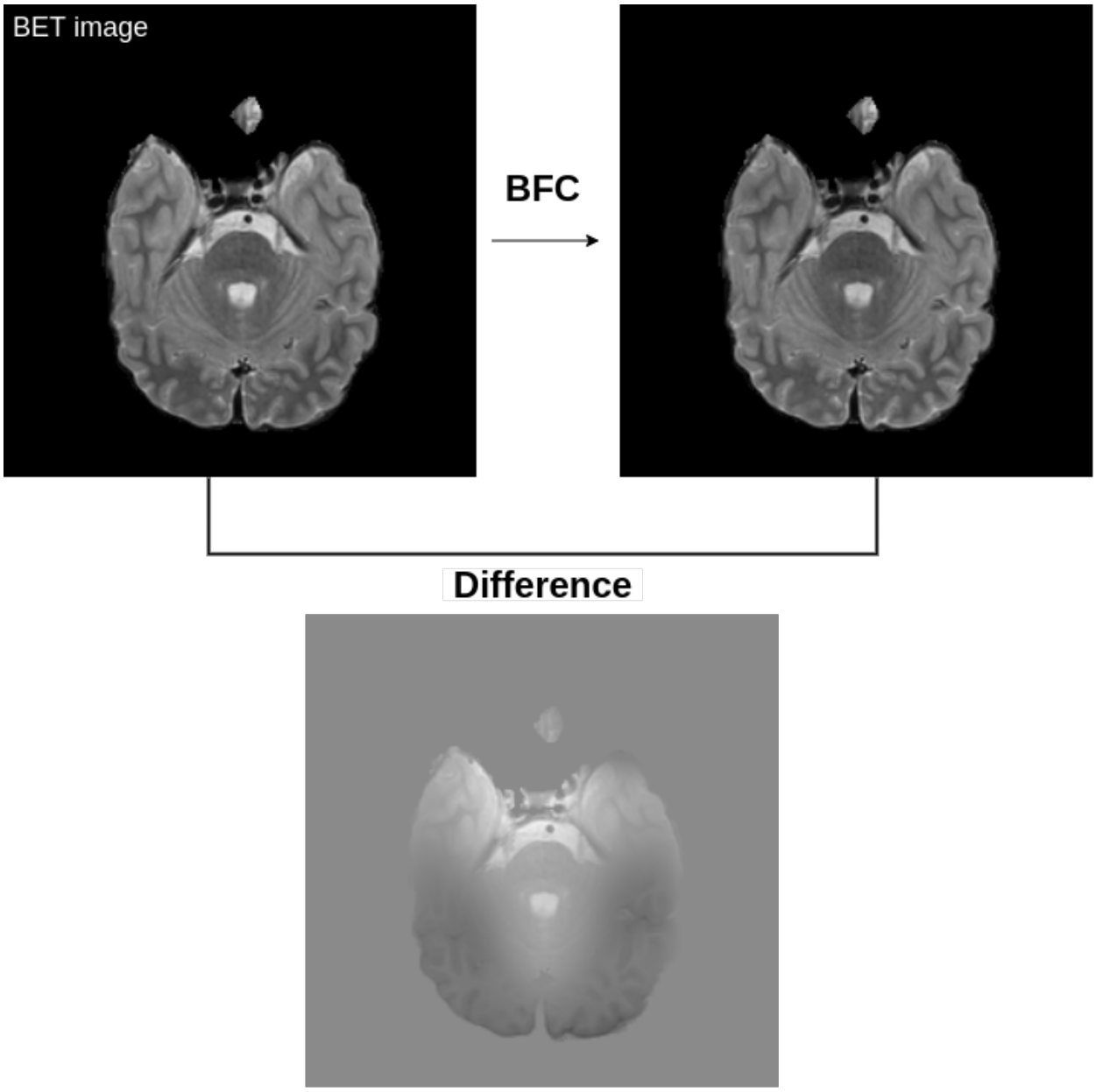



Figure 3.8: Example of the application of the bias-field correction algorithm. On the left, we have a slice that has already undergone the skull stripping operation and, on the right we have the bias-filed corrected image. Below, the difference between the two images is displayed in order to facilitate the visualization of what the BFC operation does. The lighter and the darker shades of grey indicates where the image changed the most. The gradient-like appearance of this image corresponds to the field inhomogeneities.

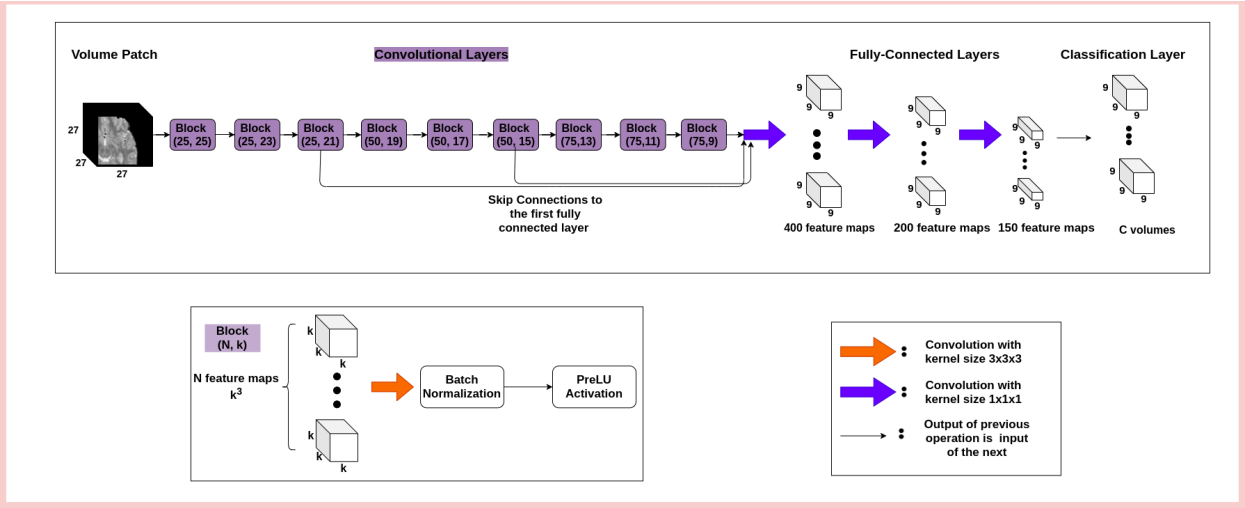



Figure 3.9: LiviaNet's architecture.

semantic segmentation task (with the background included). These maps are normalized by applying the softmax operation (shown in the previous section in Equation 2.6), which produces voxel-wise probability maps for each class.

### Training and Validation strategies

Given how we received the data (as detailed in section 3.1.1), the datasets hereafter detailed were constructed. We used the same sets for training and validation for A1, because the dataset was small, since we had only four samples available. Then, in A2, we used the 11 volumes (the four in A1, and additionally 7 volumes that were previously unseen) as training and the four from A1 as validation, in order to compare if the addition of data would improve the network performance. Finally, A3 was composed by the same 11 first volumes, but had 8 volumes for the training set and 3 for the validation one, which were chosen randomly. With the newest sagittal data, we constructed set A4. Table 3.1 shows the number of volumes in each division into training, validation and test for each of the four mentioned dataset. Therefore, the experiments conducted used the dataset division showed on Table 3.1.

| Name | Training Data | Validation Data | Test Data |
|---|---|---|---|
| A1 | Patients 1-4 | Patients 1-4 | - |
| A2 | Patients 1-11 | Patients 1-4 | - |
| A3 | 8 patients | 3 patients | - |
| A4 | 26 patients | 3 patients | 3 patients |

Table 3.1: The datasets used for training the LiviaNet, and its split into training, validation and test sets. The patients in each set of A3 and A4 were chosen randomly.

For each dataset listed above, we trained three models, each using a different loss (cross-entropy, Dice and Lovász). Therefore, in total we have 12 models ({A1, A2, A3, A4} $\times$ {CE, Dice, Lovász}). By doing so, we wanted to compare the impact of the loss function

for the datasets and the classes in the final performance of the model.

For the update of the network's weights, we used the ADAM (from ADAptive Moment estimation) optimizer and set its hyperparameters $\alpha = 0.9$ and $\beta = 0.999$. Combined to this, the learning rate decay strategy was adopted, so every 30 epochs the learning rate was divided by two. In order to save the best model during training, inference was performed every 10 epochs in the validation set. Then, the best performing model was applied to the test set volumes.

### 3.2.3 Postprocessing

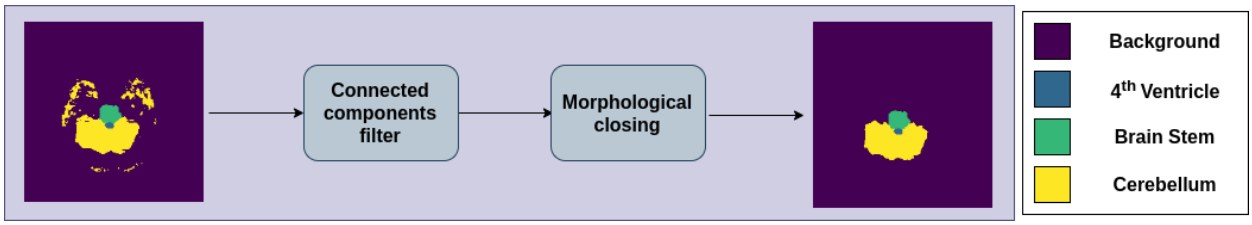



Figure 3.10: The pipeline of the post processing strategy adopted.

Figure 3.10 shows the output of the model, and then, the result of the postprocessing. The output image has various small regions, which appear due to the patches approach we adopted in the prediction model. In order to remove them, a filtering based on the analysis of the connected components (CCs) by slice of the volume is performed. For each class, except the background, we obtain the binary image of the class for each slice of the volume, an example can be seen in Figure 3.11. Then, the CCs are computed for every binary image and only the biggest one of each class is kept on the final slice as shown in Figure 3.12. We have a slightly different rule for the cerebellum, because sometimes the cerebellar region is disconnected in the same slice, so we keep the two biggest components, if they have similar size. In order to obtain the CCs, the *measure.label* function from SCIKIT-IMAGE python library was used. This method is based on the union-find data structure. Two neighbor pixels are considered to belong to the same CC if they have the same value in the input image.

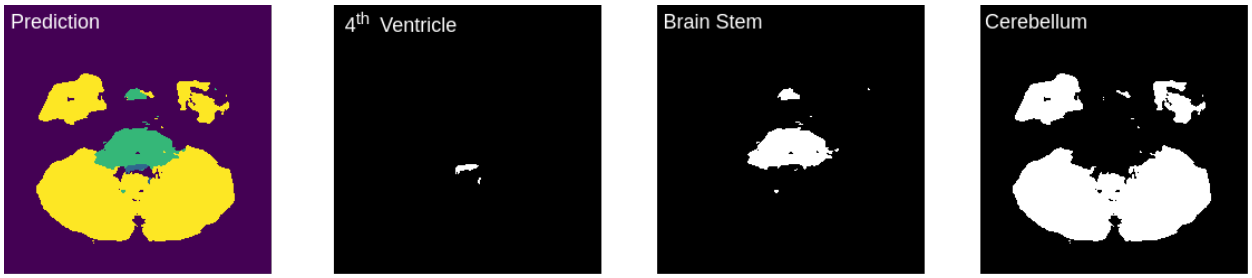



Figure 3.11: The binary images for each of the classes – fourth ventricle, brain stem and cerebellum.

Then, a 3D closing morphological operator is applied to the image in order to eliminate some missegmentations present in the final result that resemble holes inside the structures of interest. We used a structuring element $3 \times 3 \times 3$. The final result can be seen in Figure 3.13.

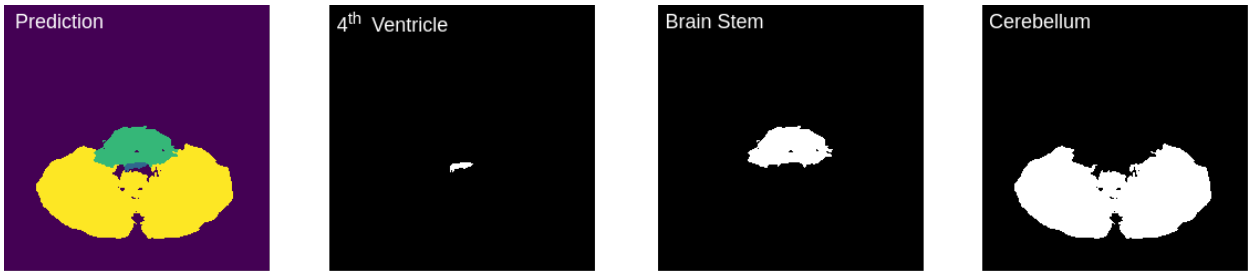



Figure 3.12: The filtered images for each class, where only the biggest connected component was kept – fourth ventricle, brain stem and cerebellum.

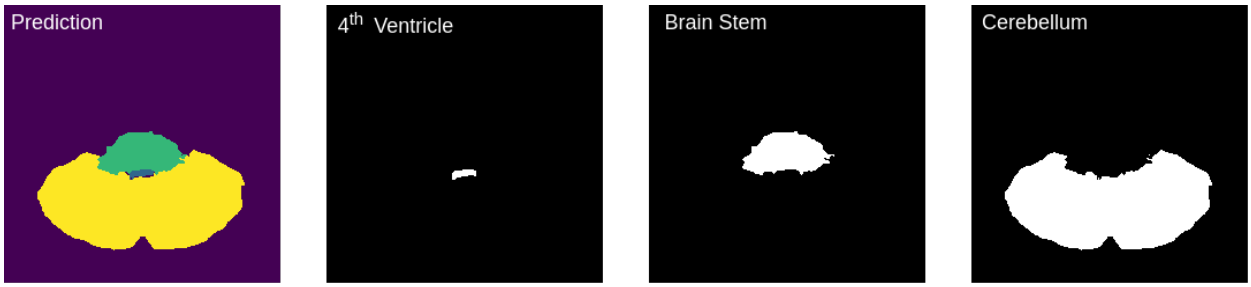



Figure 3.13: The final images for each class, after the connected components and closing operations – fourth ventricle, brain stem and cerebellum.

# Chapter 4

# Experiments

## 4.1 Preliminary results

### 4.1.1 Supersegmentation Approaches

Besides the proposed approach, the work also includes a preliminary research on using supersegmentation algorithm for automatic segmentation of pediatric images. Although this research has not yet lead to conclusive findings, the work is described here to help future initiatives. The proposed pipeline is shown in Figure 4.1. After the supersegmentation step, the regions obtained would be vertexes in a graph, where neighboring superpixels would be connected by an edge. The edges and vertex would have attributes that would describe them in discriminatively enough. So, with this set of graphs from the supersegmentation a Graph Neural Network will be trained to perform node classification and, therefore, label the corresponding regions of the image.

The experiments presented below were conducted using just a subset (the first 3 patients data we had) of the dataset used in this project.

#### Watershed with Gaussian Smoothing

In order to reduce the noise of the images, a Gaussian smoothing filter (Equation 4.1) was applied to each slice of the dataset and then, the local minima were found and used as initial markers for the watershed algorithm. The parameter for the Gaussian tested were $\sigma \in \{0.05, 0.1, 0.5, 1, 3, 5, 7\}$. The ones that presented best results for each patient are reported in Table 4.1.

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.1}$$

where $(x, y)$ is the pixel location on the image.

| Patient | $\sigma$ | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 1.0 | 0.913 | 122726 | 65890 | 1.377 |
| 2 | 1.0 | 0.880 | 110642 | 48133 | 1.129 |
| 3 | 0.5 | 0.924 | 173970 | 63628 | 1.475 |

Table 4.1: Results of the watershed test with Gaussian smoothing.

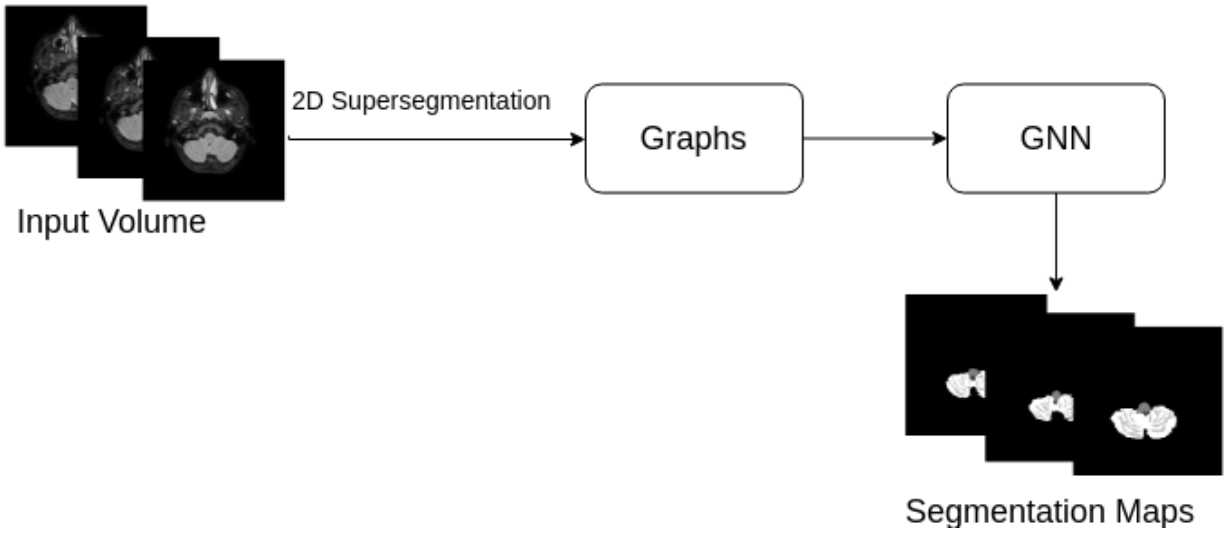



Figure 4.1: Pipeline based on the use of supersegmentation methods and Graph Neural Networks to perform semantic segmentation on MRI volumes.

Then, with these results, different thresholds for the region merging algorithm were also tested. The values tested were $\{5, 10, 20, 30, 40, 50\}$. The results that yielded best mean dice values while reducing the number of final regions are presented in Table 4.2.

| Patient | Threshold | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 10 | 0.909 | 68215 | 36451 | 58.228 |
| 2 | 10 | 0.878 | 45420 | 22135 | 69.657 |
| 3 | 10 | 0.921 | 65937 | 27504 | 102.603 |

Table 4.2: Results of the watershed test with Gaussian smoothing and region merging post-processing.

It was possible to notice that small values for the $\sigma$ parameter produced a watershed result with a greater number of regions, which are small in size. Figure 4.2 shows the watershed segmentation for slices with little structures. Larger values of $\sigma$ result in a final supersegmentation with less segments, however, the little structures as the ones in Figure 4.2 are oversegmented, which is not good for the pipeline proposed.

Figure 4.3 shows how it is possible to use different values for both $\sigma$ and the threshold parameters for slices containing larger structures. However, such parameters does not perform well for the little structures, yielding a zero-valued dice coefficient between the optimal labelmap and the ground truth.

#### Watershed with Anisotropic Diffusion Smoothing

In this case, the smoothing applied was the Anisotropic Diffusion filter (discrete version in Equation 4.2 [7]), which is commonly used for MRI images. Then, markers were determined from local minima and watershed transformation was applied as in previous section.

$$I_s^{t+1} = I_s^t + \frac{\lambda}{|\eta_s|} \sum_{p \in \eta_s} g(\nabla I_{s,p}) \nabla I_{s,p} \tag{4.2}$$

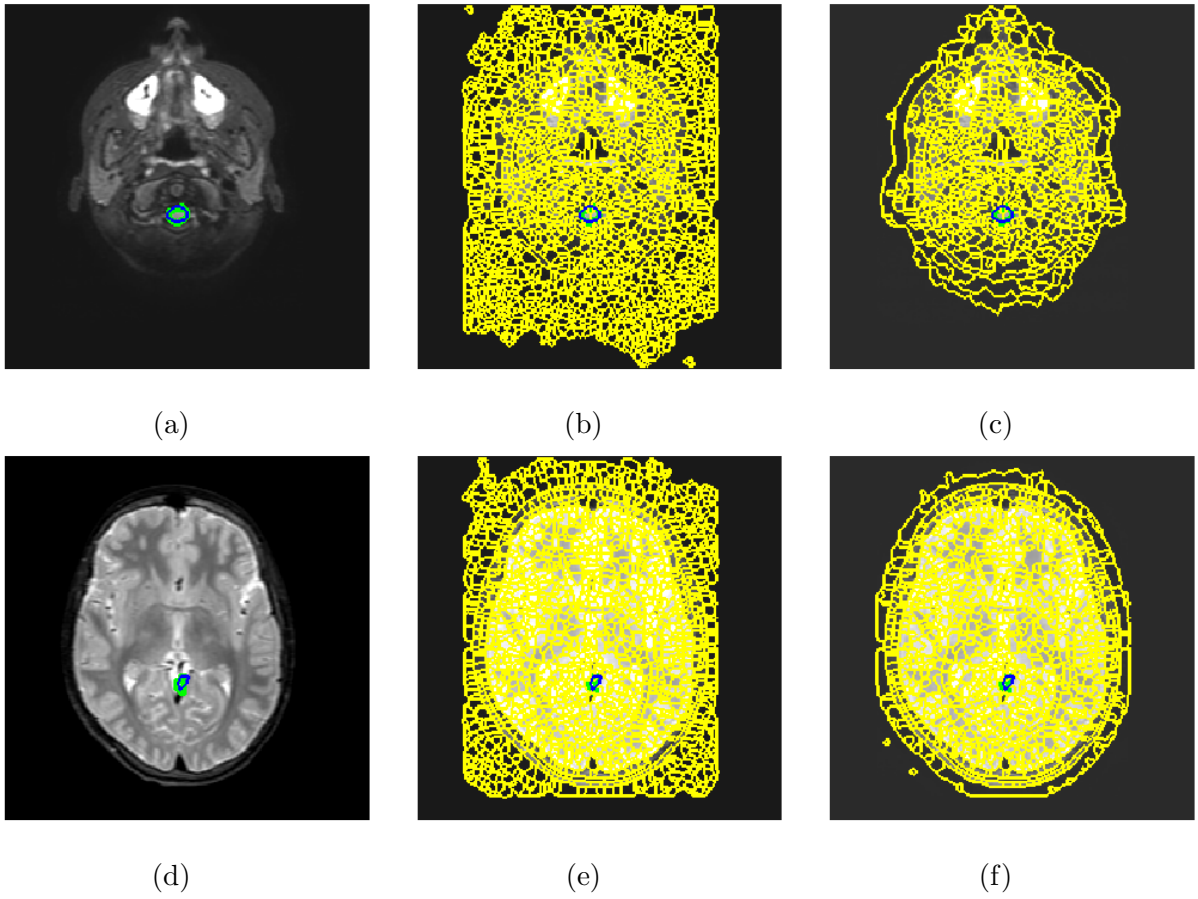


Figure 4.2: Slices with little structures from Patient 1 - watershed with Gaussian smoothing, using parameters from Tables 4.1 and 4.2. (a) Optimal label (blue) and ground truth (green), identifying a brain stem region; (b) The same as (a), but with the supersegmentation overlay (yellow) (dice 0.87, 1865 regions) ; (c) Same as (b) after region merging (dice 0.87, 758 regions); (d) Optimal label (blue) and ground truth (green), identifying a cerebellum region; (e) The same as (d), but with the supersegmentation overlay (yellow) (dice 0.67, 1737 regions); (f) same as (e) after region merging (dice 0.67, 1145 regions).

where $I_s^t$ is an image in time step $t$ (iteration) and $s$ is a pixel location, $\lambda$ denotes the diffusion rate, $\eta_s$ is the spatial neighborhood of $s$. $\nabla I_{s,p} = I_p - I_s^t$ is the gradient of the image in direction $p$. And $g(.)$ is an edge-stopping function.

Different values were tested for the number of iterations of the anisotropic diffusion ($N \in \{1, 5, 10, 20, 30, 50\}$). Table 4.3 shows those that presented a best final watershed results. The threshold values tested for the region merging method were the same as for Gaussian smoothing pre-processing ($\{5, 10, 20, 30, 40, 50\}$.). In Table 4.4, the results of region merging are displayed.

When using anisotropic diffusion smoothing, there is also a relation between how much the pre-processed image is smoothed and the number of regions produced by the watershed transformation. The fewer the number of iterations($N$), more and smaller regions are produced. Because of slices with little structures, as those in Figure 4.4, smaller values of $N$ result in higher mean dice values.

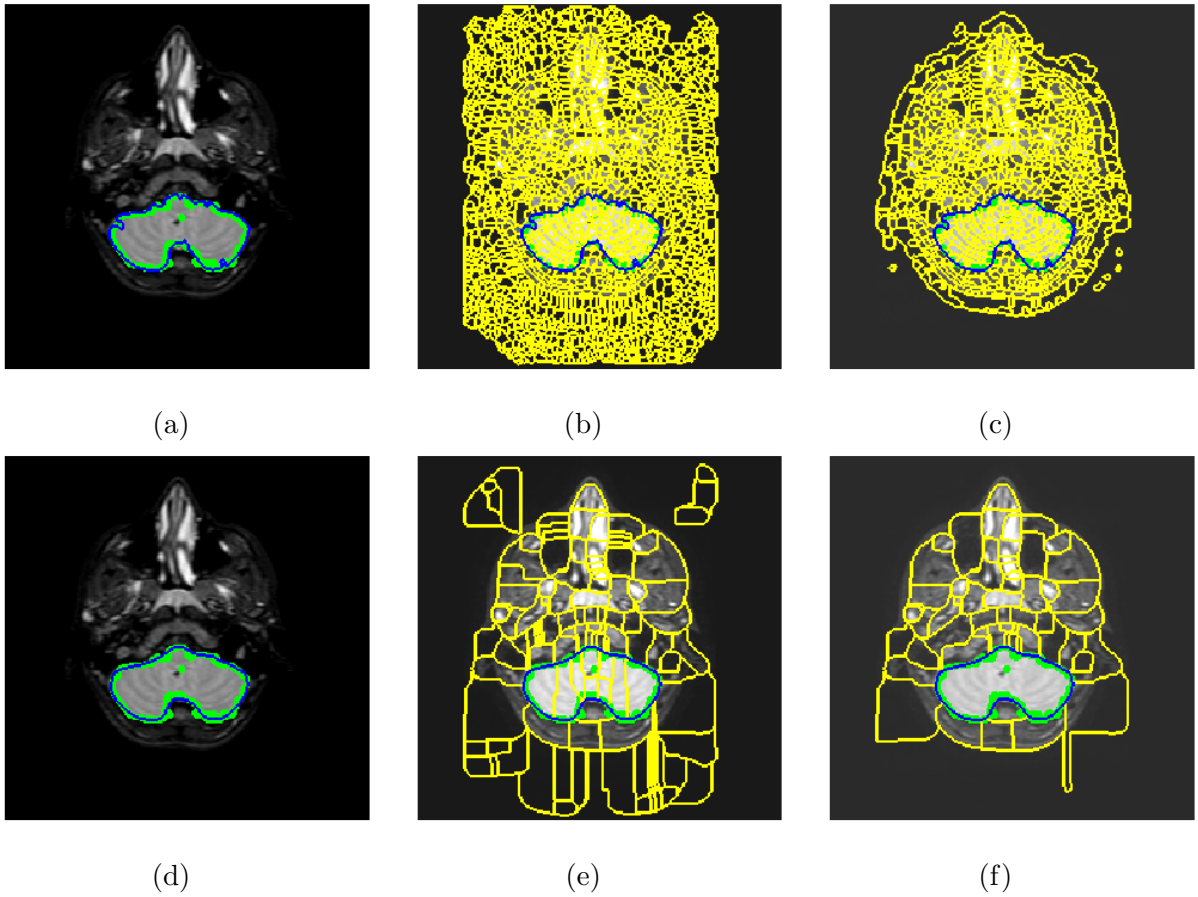


Figure 4.3: Images from slice 13 of Patient 1. (a) optimal labelmap (blue) obtained using $\sigma = 1$ for Gaussian smoothing and ground truth (green); (b) The same as (a), but with the supersegmentation overlay (yellow) (dice 0.94, 1967 regions); (c) Same as (b) after region merging with threshold 10 (dice 0.94, 928 regions); (d) Same as (a), but using $\sigma = 5$; (e) Same as (d), with the supersegmentation overlay (yellow) (dice 0.95, 118 regions); (f) Same as (e) after region merging with threshold 50 (dice 0.95, 59 regions)

.

**SLIC with Gaussian Smoothing**

Gaussian smoothing was applied as a pre-process step, similar to what is done in Section 4.1.1. Then, the SLIC transform is applied to the filtered image in order to find superpixels. Parameters for the Gaussian filter (Equation 4.1) parameter $\sigma$ were tested in the set $\{0.5, 1, 1.3, 1.5, 1.7, 2\}$, because smaller or bigger $\sigma$ than these values produced degraded results.

For the SLIC compactness parameter ($m$) it was first tested the values on log scale {0.001, 0.01, 0.1, 1, 10, 100}, as it is commonly done, before refining the chosen value. Then, $m$ in $\{0.001, 0.01, 0.030.05\}$ was tested, because of results yielded by the previous test. The parameters that obtained best dice coefficient scores are showed in Table 4.3.

Then, the region merging was applied and the best results can be seen in Table 4.6. The number of both regions of the entire volume and the regions only of the ROI are reduced by more than one-third for all the three patients data.

In Figure 4.5 it is possible to notice that the little structures from some slices also

| Patient | $N$ | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 10 | 0.913 | 112919 | 59866 | 1.409 |
| 2 | 10 | 0.880 | 94367 | 42586 | 1.329 |
| 3 | 10 | 0.925 | 105467 | 40226 | 1.428 |

Table 4.3: Results of the watershed test with anisotropic diffusion smoothing.

| Patient | Threshold | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 10 | 0.910 | 71920 | 38195 | 53.239 |
| 2 | 10 | 0.875 | 46860 | 23279 | 54.289 |
| 3 | 10 | 0.921 | 51653 | 22198 | 58.458 |

Table 4.4: Results of the watershed test with anisotropic diffusion smoothing and region merging post-processing.

influence the choice of parameters similar to what happens with watershed. If $\sigma, m$ are bigger than the ones in Table 4.5 these structures are not present in the optimal labelmap, due to oversegmentation.

#### SLIC with Anisotropic Diffusion Smoothing

Similarly, for the anisotropic diffusion pre-processing, the number of iterations was tested for $N \in \{100, 150, 200\}$, because smaller values yielded worst superpixel results. Then, the log-scale test was performed for the compactness parameter $m$, as in previous section. The results that presented best mean dice score are presented in Table 4.7.

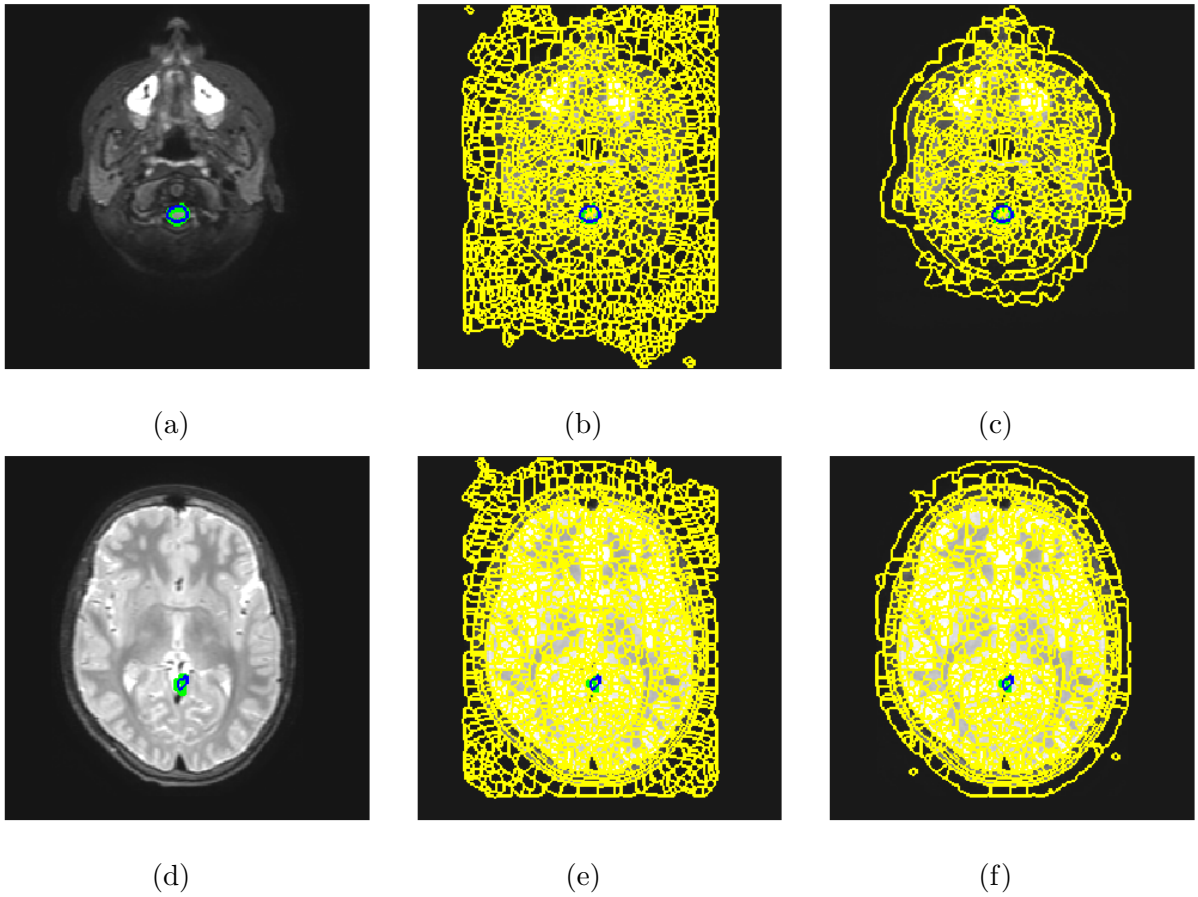
The region merging post-processing is exhibited in Table 4.8. It is possible to notice how little values of threshold produced a significant reduction on the final aggregated number of regions of the 70 slices of each volume.

Figure 4.6 shows the SLIC segmentation results of slices from Patient 1, where the ROIs are small. Because of these regions the parameters that yielded a best result are the ones that generate small superpixels. Thus, the final segmentation is composed of a lot of regions. For other slices, where the ROIs are bigger, it is possible to use different parameters that generate a final result with similar dice score, but less regions.

#### Discussion

The small structures present on some slices of the ROI determine the parameters chosen for the watershed method and both pre- and post-processing techniques used. Therefore, the final result is the one with smaller regions, that are capable of containing these structures more tightly, that is, with less oversegmentation. On the other hand, this result also present many regions, which is not good in terms of efficiency for the rest of the A2 pipeline.

These structures were also determining for the choice of parameters of the SLIC method and the pre-processing stepping. However, differently from the watershed results, the region merging technique was able to reduce significantly the number of regions for the final volume segmentation using a small threshold. Thus, in terms of adherence to boundary both watershed and SLIC perform similarly, but the latter produces less regions after post-processing



(a) (b) (c)

(d) (e) (f)

Figure 4.4: Slices with little structures from Patient 1 - watershed with anisotropic diffusion smoothing, using parameters from Tables 4.3 and 4.4. (a) Optimal label (blue) and ground truth (green), identifying a brain stem region; (b) The same as (a), but with the supersegmentation overlay (yellow) (dice 0.86, 1519 regions); (c) Same as (b) after region merging (dice 0.86, 752 regions); (d) Optimal label (blue) and ground truth (green), identifying a cerebellum region; (e) The same as (d), but with the supersegmentation overlay (yellow) (dice 0.68, 1667 regions); (f) same as (e) after region merging (dice 0.68, 1219 regions).

than the former.

## 4.2 Dataset

As cited in Section 3.1, the dataset used in this work was gradually formed and is heterogeneous, containing both volumes in the axial and sagittal views, as well as patients with ages varying from zero to 18 years. In Figure 4.7, different volumes that were originally taken in the axial plane are shown. Each line contains the exam from a patient, and each column slices at different heights from the volume. From the bottom to the top of the head, we take slices 15, 30, 45 and 60 out of 70 in total. Rows ones and four shows very similar slices, differing mainly in the size of the brain. Something similar happens between rows two and three. In the last row, however we can notice some similarity to patients in rows one and four, however the brain is much smaller and the contrast is inverted, because the patient is 0 years old and their brain is in the myelination process phase.

Figure 4.8 shows different volumes that were originally captured in the sagittal plane.

| Patient | $\sigma$ | $m$ | $R$ | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|---|---|
| Patient 1 | 2 | 0.001 | 2000 | 0.910 | 157283 | 80806 | 2.942 |
| Patient 2 | 2 | 0.001 | 2000 | 0.879 | 158039 | 67654 | 2.448 |
| Patient 3 | 1 | 0.01 | 2000 | 0.914 | 161199 | 59876 | 2.220 |

Table 4.5: Parameters that presented the best dice scores for the SLIC segmentation with Gaussian pre-processing. The $\sigma$ is the Gaussian smoothing parameter, $m$ is the compactness (see Section 2.5.2),and $R$ is the approximate number of regions to be generated, defined by the user.

| Patient | Threshold | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 5 | 0.906 | 58630 | 30555 | 101.912 |
| 2 | 5 | 0.875 | 36488 | 17682 | 189.466 |
| 3 | 5 | 0.909 | 41554 | 17389 | 135.227 |

Table 4.6: Results of the SLIC test with Gaussian smoothing and region merging post-processing.

Each row contains a patient and each column a slice at different sections from the left to the right ear. The slices are the 30, 50, 70 and 90 from the first to the fourth column. patients in rows one, three and four have similar parts of the head being displayed on the slices, with some differences in size. Something similar can be seen for rows two and five. Also, for the younger patients in rows two and four (who are 0 year old), it is possible to see that some of their necks and spines are visible in the slices as well.

Such variability seen among the slices in both figures can be justified by the patient's age, their head inclination during the exam and relative position on the table of the MRI machine. Such factors contribute for such different slices at the same volume "height" when comparing distinct volumes. For example, patients from rows three and four in Figure 4.7 and rows two and four in Figure 4.8 are the same age, but the slices in the same height are different, which is probably justified by the other two factors cited.

Figure 4.9 shows intermediate slices (slice 30 out of 70, from the base head to the top of the ) for patients in different age ranges per row. The volumes are all in the axial view, for better comparison (the ones that were originally in the sagittal view have been transposed and resampled to match the axial exams). The first line of the figure that presents the patients from zero to two years is the one that shows more variability when compared to the others. Even though all the slices are at the same height, we can notice a lot of variability among the patients. This could be solved if some registration method was used. However, these algorithms usually present some drawbacks such as elevated time and memory consumption. Also, they tend to fail for large age range, as is the case of our volumes. Moreover, since the segmentation method used in this project is patch-based, such variability should not affect too much the performance of the algorithm.

## 4.3 Segmentation Results

### 4.3.1 Preprocessing

Figure 4.10 shows fours slices (each column) of five different patients (per row), after the skull stripping operation was performed as explained in Section 3.2.1. It is possible to

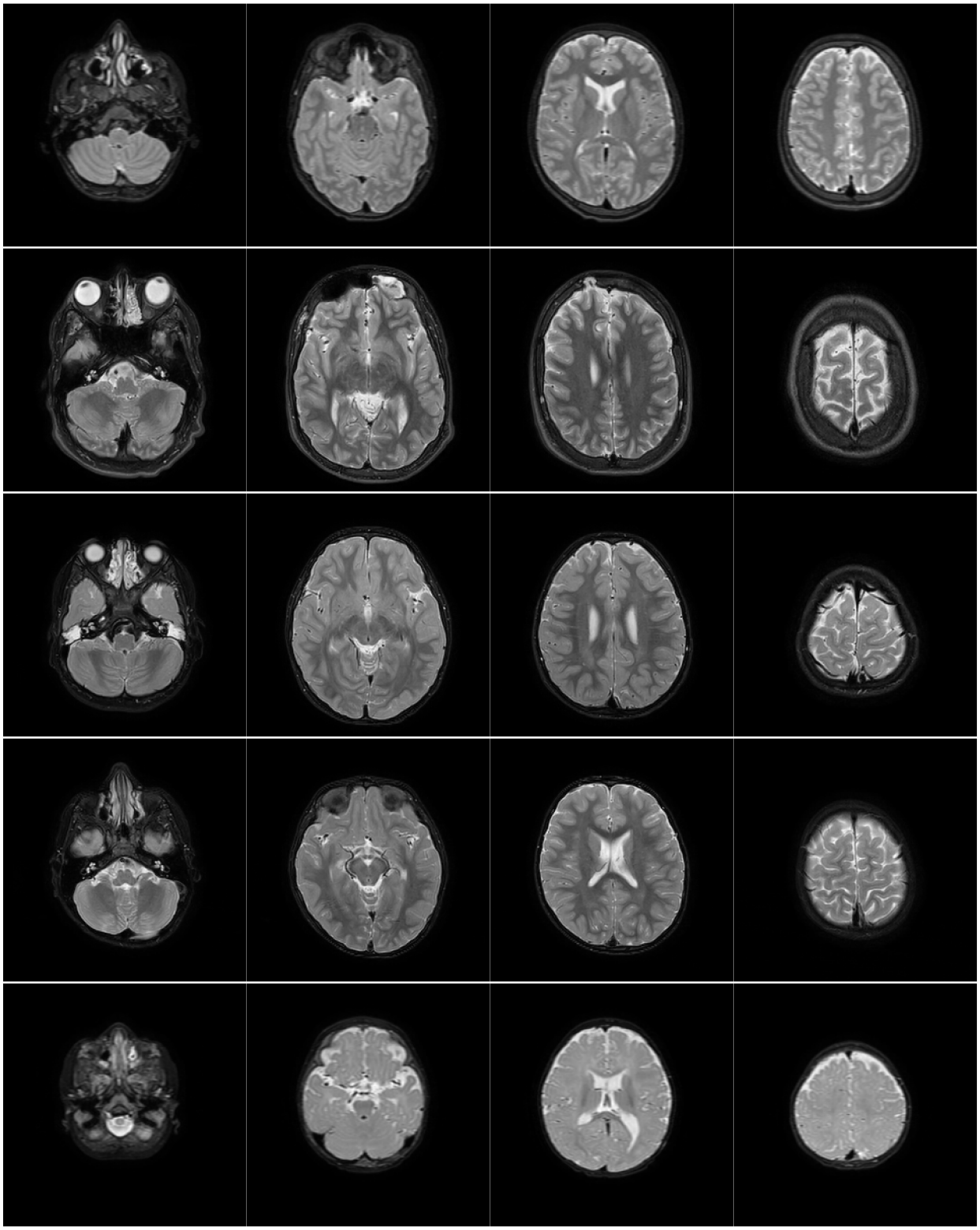


Figure 4.7: Mosaic for the visualization of different volumes that were originally in the axial plane. Each line shows slices at different heights (slices 15, 30, 45, and 60, respectively) for one volume, from the base to the top of the head. From the first row to the last, the patients are 13, 18, 4, 4 and 0 years old.

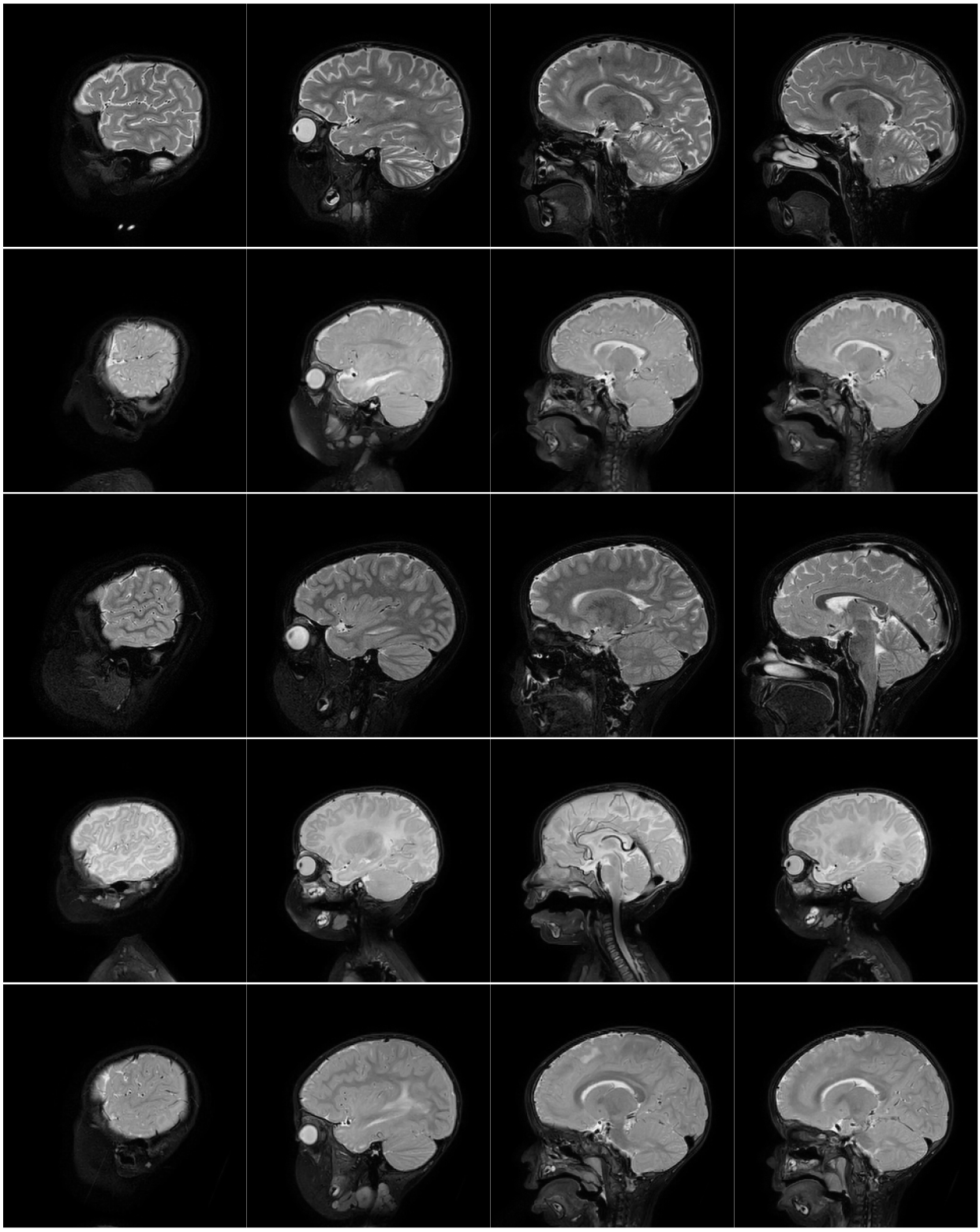


Figure 4.8: Mosaic for the visualization of different volumes that were originally in the sagittal plane. Each line shows four slices at different heights (slices 30, 50, 70 and 90, respectively) for one volume. From the first row to the last, the patients are 6, 0, 7, 0 and 1 years old.

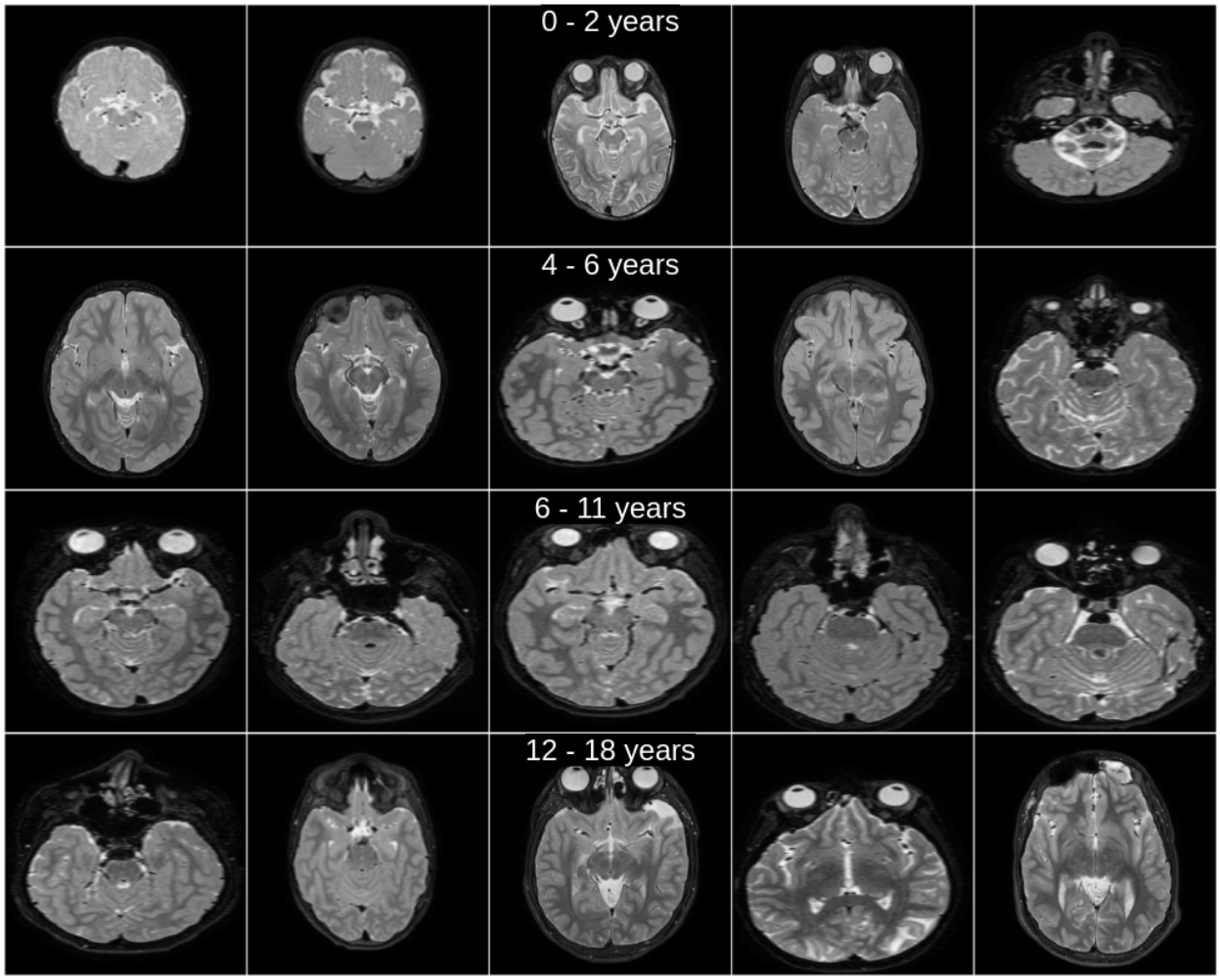
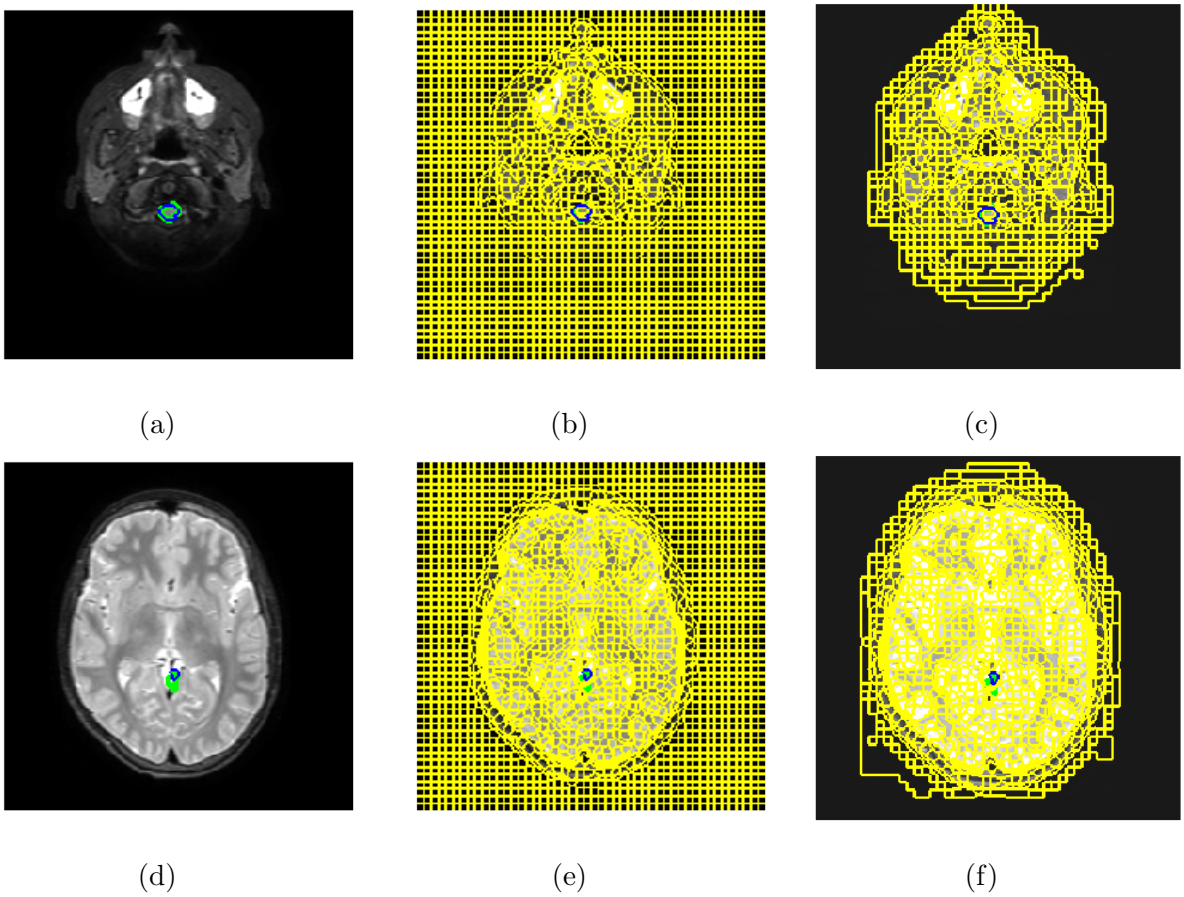



Figure 4.9: Visualization of a slice from different volumes at different age ranges, from the youngest to the oldest.



(a) (b) (c)

(d) (e) (f)

Figure 4.5: Slices with little structures from Patient 1 - SLIC with Gaussian smoothing parameters from Tables 4.5 and 4.6. (a) Optimal label (blue) and ground truth (green), identifying a brain stem region; (b) The same as (a), but with the supersegmentation overlay (yellow) (dice 0.89, 2271 regions) ; (c) Same as (b) after region merging (dice 0.89, 672 regions); (d) Optimal label (blue) and ground truth (green), identifying a cerebellum region; (e) The same as (d), but with the supersegmentation overlay (yellow) (dice 0.39, 2254 regions); (f) same as (e) after region merging (dice 0.39, 1002 regions).

notice that after the BET some regions nearer the top of the head were removed from the volumes (the fourth column of the Figure 4.11). We could easily modify the threshold used in order to avoid such effect. However we verified that the brain extraction operation did not removed parts from the ROI. Therefore, we decided to keep the default threshold of the method.

Similarly, Figure 4.11 displays the same five patients after the normalization and Bias Field Correction filter were applied. As explained in Section 3.2.1, it is hard to perceive the operation result visually when comparing to the original input. However we tried to train the model only with the BET result and the predictions of the network were worse than those obtained with the data also filtered by the BFC. Therefore, we concluded that BFC had a positive impact on the model's convergence.

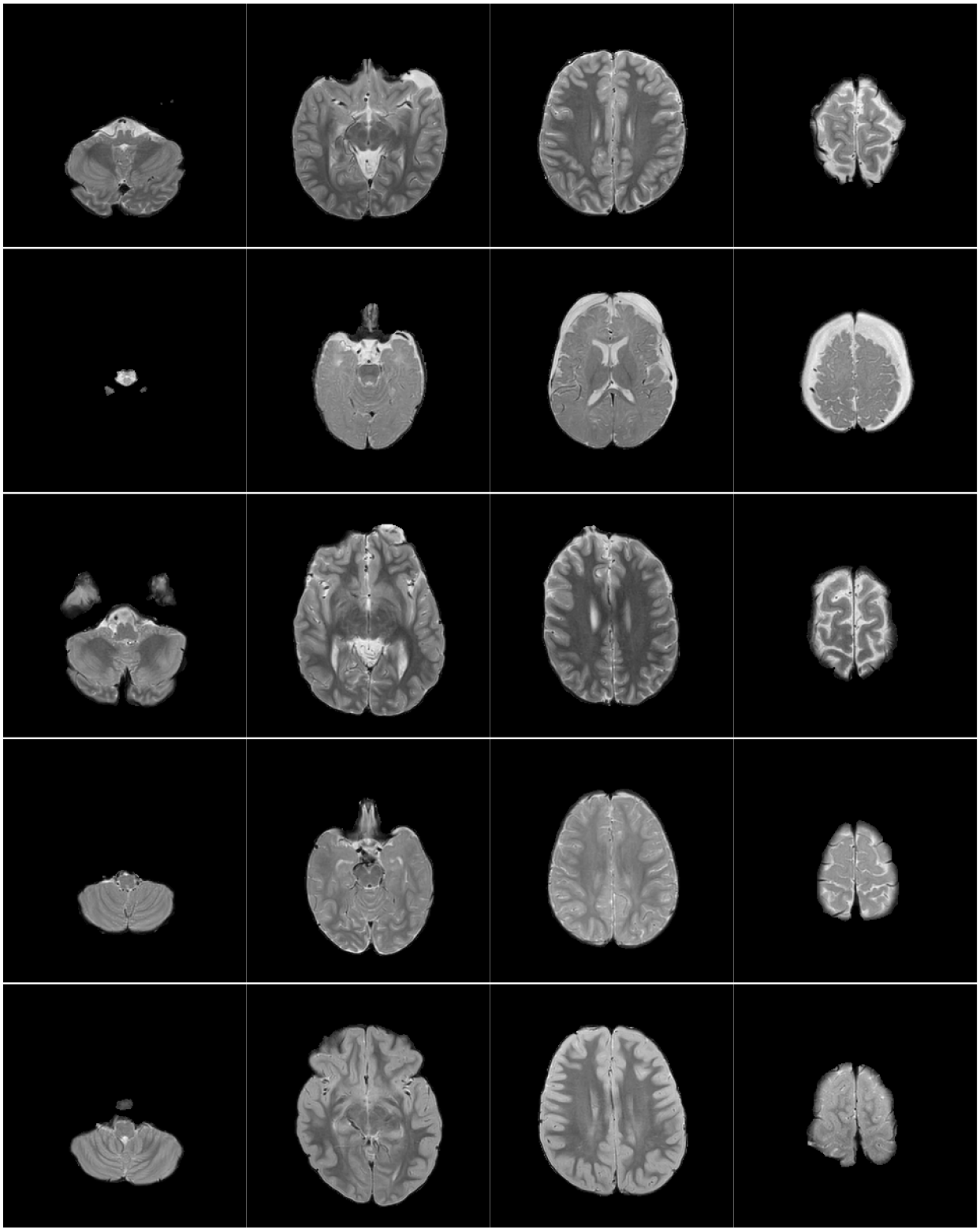


Figure 4.10: Axial view of five volumes after the skull stripping performed with BET. Each column shows four patients slices from the bottom to the top of the head.

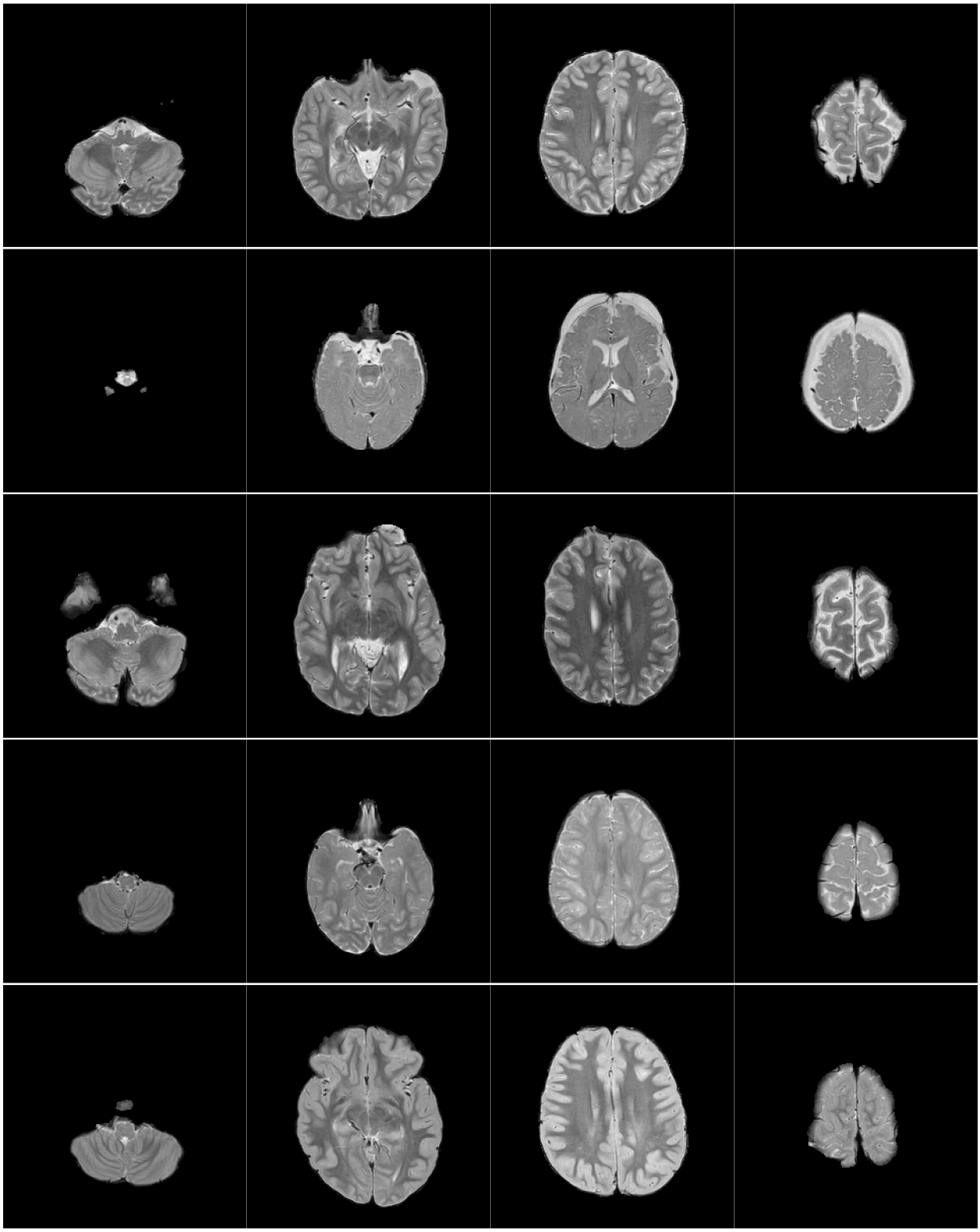


Figure 4.11: Axial view of five volumes after the skull stripping performed with BET

| Patient | $N$ | $m$ | $R$ | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|---|---|
| Patient 1 | 100 | 100 | 2000 | 0.903 | 153748 | 78885 | 2.234 |
| Patient 2 | 200 | 300 | 2000 | 0.871 | 159222 | 68111 | 2.243 |
| Patient 3 | 150 | 300 | 2000 | 0.918 | 158786 | 58761 | 2.129 |

Table 4.7: Parameters that presented the best dice scores for the SLIC segmentation with anisotropic diffusion pre-processing for the whole volume applying the methods for each slice. $N$ is the number of iterations performed by anisotropic diffusion, $m$ is the compactness (see Section 2.5.2),and $R$ is the approximate number of regions to be generated, defined by the user.

| Patient | Threshold | Mean Dice | No Regions | No Regions ROI | Time (s) |
|---|---|---|---|---|---|
| 1 | 5 | 0.902 | 55148 | 28704 | 101.310 |
| 2 | 5 | 0.865 | 36833 | 17823 | 171.812 |
| 3 | 5 | 0.912 | 39413 | 16392 | 133.651 |

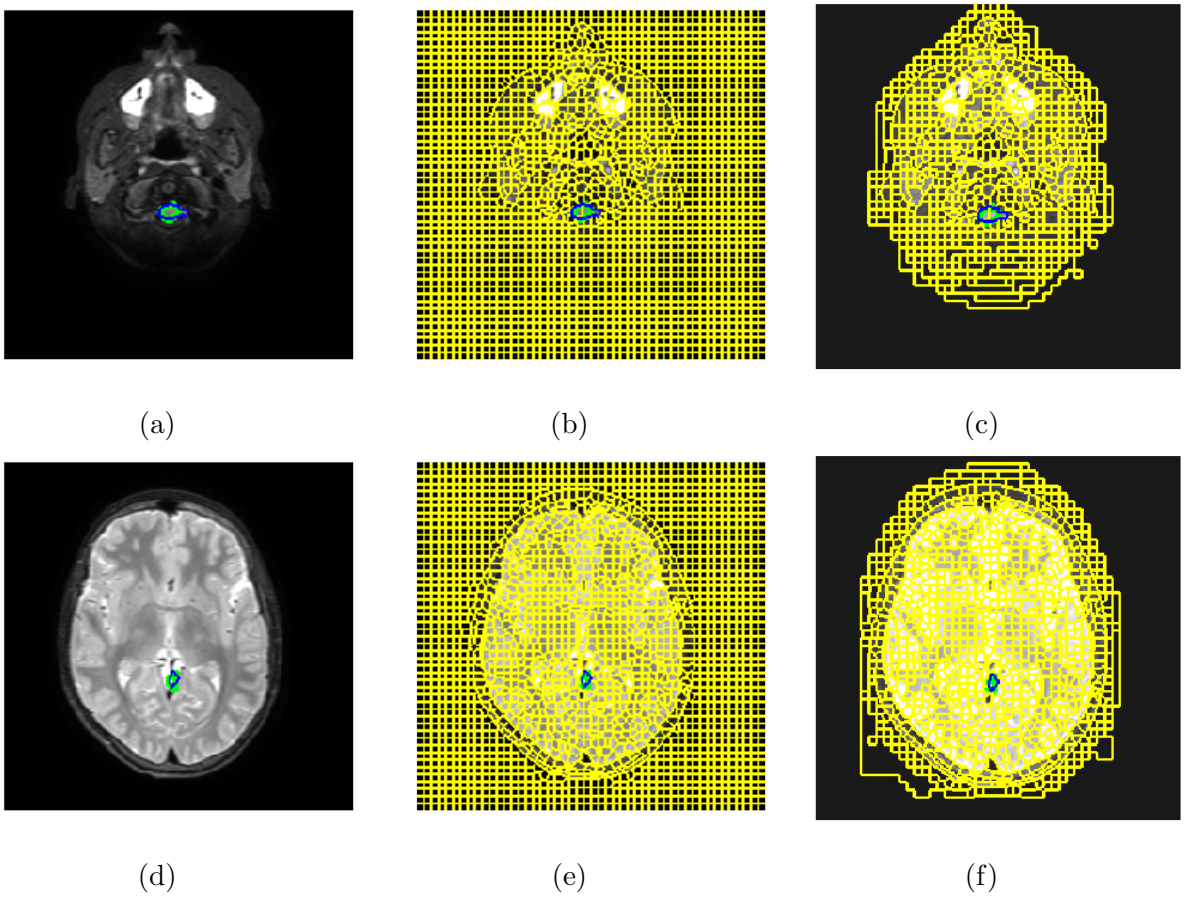
Table 4.8: Results of the SLIC test with anisotropic diffusion smoothing and region merging post-processing for the whole volume applying the methods for each slice.

### 4.3.2 LiviaNet

LiviaNet was trained for each dataset presented in 3.1 and with each one of the three losses explained in the Section 3.2.2. Therefore, we have 12 models in total. Table 4.9 shows the training performance for the model trained with each dataset and loss. It is possible to conclude that the model somewhat overfitted for the A3 and A4 datasets due to the difference in the mean dice obtained in the training data and the validation one. Also, observe that on these datasets both sets are disjoint. A1 and A2 are not, because there was not enough data to split A1, and A2 was designed to have the same 'validation' set as A1 for comparison. Furthermore, the results obtained with dataset A4 were better than those of the models trained using A3, showing that there was a benefit in including more volumes to the training set.

Moreover, we tested different initial learning rate for each model and the chosen one is also displayed in Table 4.9 in the column named *lr*. During training, the learning rate decay strategy is applied, because using a fixed rate may cause the network to never converge [3]. Every 30 epochs, the learning rate was divided by two and updated in the model training. By doing so we guarantee that the optimizer takes smaller steps during the function minimization, which facilitates the convergence of the model.

For the first three datasets, we trained the models for 100 epochs. Figure 4.12 shows that the network converged for all of them with the cross-entropy loss. For the dice loss, however the model was not able to converge in only 100 epochs, the learning curve stabilizes after the first epochs. Lovász loss has an expected learning curve, that is, the model is converging, but the result may be better if trained for more epochs. For A4, we have something similar, however the network was trained for 350 epochs, instead. We decided on doing so, because there were more volumes in the training set of this dataset.



(a) (b) (c)

(d) (e) (f)

Figure 4.6: Slices with little structures from Patient 1 - SLIC with Anisotropic Diffusion smoothing parameters from Tables 4.7 and 4.8. (a) Optimal label (blue) and ground truth (green), identifying a brain stem region; (b) The same as (a), but with the supersegmentation overlay (yellow) (dice 0.75, 2239 regions) ; (c) Same as (b) after region merging (dice 0.75, 633 regions); (d) Optimal label (blue) and ground truth (green), identifying a cerebellum region; (e) The same as (d), but with the supersegmentation overlay (yellow) (dice 0.72, 2182 regions); (f) same as (e) after region merging (dice 0.72, 918 regions).

### 4.3.3 Postprocessing

Figures 4.13, 4.14, 4.15, 4.16 show some outputs from the model trained with each one of the three losses on the respective validation sets, before and after post-processing. It is possible to see how all smaller areas are removed from the image, thus improving the segmentation result. Similarly, Figure 4.17 shows the prediction and post-processing results of the models for the test set of A4. Figures 4.15, 4.16, 4.17 shows that the morphological closing was efficient to fully or partially close the holes present in the network prediction. Some holes were not fully closed because of the size of the structuring element (a $3 \times 3 \times 3$ cube) used to perform the closing. We chose this size, because it was the one that produced better results, without extrapolating the segmented regions. In the next Section 4.4, more details for the postprocessing will be given.

| Dataset | Loss Function | lr | Mean training dice | Mean validation dice |
|---|---|---|---|---|
| A1 | CE | 0.001 | 0.763 | 0.763 |
| | Dice | 0.0001 | 0.576 | 0.576 |
| | Lovász | 0.001 | 0.797 | **0.797** |
| A2 | CE | 0.001 | 0.745 | 0.744 |
| | Dice | 0.0001 | 0.558 | 0.558 |
| | Lovász | 0.001 | 0.751 | **0.746** |
| A3 | CE | 0.001 | 0.706 | **0.619** |
| | Dice | 0.0001 | 0.570 | 0.501 |
| | Lovász | 0.0001 | 0.730 | 0.614 |
| A4 | CE | 0.001 | 0.778 | **0.662** |
| | Dice | 0.0001 | 0.553 | 0.518 |
| | Lovász | 0.0001 | 0.709 | 0.636 |

Table 4.9: Training metrics obtained for the LiviaNet model trained with each dataset (Table 3.1) and each loss function.

## 4.4 Validation

In order to evaluate the results obtained, the dice score was calculated for each class in the validation subsets. All the results improved after the post-processing was applied (Table 4.10). For datasets A1, A2 and A3, the model trained with cross-entropy loss obtained better mean dice results. The one trained with Lovász obtained a comparable result, however with similar dices for the three ROI classes. Since it is important to identify all the three classes, the results suggest that the model trained with the Lovász loss would be more adequate for the current segmentation task in such datasets.

For the A4 dataset, the model trained with cross-entropy also presented better results, both before and after the post-processing for all the classes. The results obtained with Lovász loss are similar, however, differently than what happened with the previous datasets, it does not produce more class balanced results. Therefore, for this dataset we can conclude that the cross-entropy is more suitable.

Overall the models trained with Dice-score loss produced the worst results. It is possible to see in Table 4.10 that the networks optimized using such loss could not learn how to segment the fourth ventricle class. One of the reasons why it happened is because of class imbalance. The other regions have a bigger volume than the fourth ventricle. Therefore, we have a class imbalance between the fourth ventricle and the other three classes. Moreover, the dice loss works well for binary class imbalance. For the multi-class case, class weighting schemes tend to be necessary, and we believe it is the case for this task. So we suggest that such technique should be investigated as future work.

### 4.4.1 Details on results from models trained in dataset A4

Since dataset A4 had more volumes, we are going to discuss the results from the models trained with such dataset with more detail. As explained in Section 3.1, the data that was added to the initial 11 volumes (numbered from 0 to 10) were in the sagittal view (numbered from 11 to 31) and had different shapes. Therefore, to make the data less heterogeneous we

| Dataset | Loss | Background | $4^{th}$ Ventricle | Brain Stem | Cerebellum | Total |
|---|---|---|---|---|---|---|
| A1 - Val | CE | 0.991 | 0.847 | 0.526 | 0.687 | 0.763 |
| | CE + **post-proc** | 0.997 | 0.876 | 0.773 | 0.842 | **0.872** |
| | Dice | 0.989 | 0.000 | 0.739 | 0.574 | 0.576 |
| | Dice + **post-proc** | 0.996 | 0.000 | 0.863 | 0.744 | **0.650** |
| | Lovász | 0.985 | 0.740 | 0.693 | 0.563 | 0.746 |
| | Lovász + **post-proc** | 0.996 | 0.851 | 0.873 | 0.766 | **0.872** |
| A2 - Val | CE | 0.992 | 0.687 | 0.636 | 0.665 | 0.745 |
| | CE + **post-proc** | 0.997 | 0.765 | 0.784 | 0.814 | **0.840** |
| | Dice | 0.987 | 0.000 | 0.735 | 0.510 | 0.558 |
| | Dice + **post-proc** | 0.995 | 0.000 | 0.818 | 0.690 | **0.626** |
| | Lovász | 0.989 | 0.770 | 0.693 | 0.563 | 0.746 |
| | Lovász + **post-proc** | 0.996 | 0.775 | 0.775 | 0.761 | **0.827** |
| A3 - Val | CE | 0.989 | 0.397 | 0.464 | 0.624 | 0.619 |
| | CE + **post-proc** | 0.996 | 0.518 | 0.604 | 0.818 | **0.740** |
| | Dice | 0.981 | 0.000 | 0.572 | 0.451 | 0.501 |
| | Dice + **post-proc** | 0.990 | 0.000 | 0.610 | 0.614 | **0.553** |
| | Lovász | 0.976 | 0.577 | 0.534 | 0.368 | 0.614 |
| | Lovász + **post-proc** | 0.989 | 0.624 | 0.611 | 0.548 | **0.687** |
| A4 - Val | CE | 0.979 | 0.593 | 0.644 | 0.432 | 0.662 |
| | CE + **post-proc** | 0.989 | 0.627 | 0.780 | 0.564 | **0.740** |
| | Dice | 0.976 | 0.000 | 0.697 | 0.399 | 0.518 |
| | Dice + **post-proc** | 0.986 | 0.000 | 0.743 | 0.499 | **0.557** |
| | Lovász | 0.975 | 0.592 | 0.560 | 0.416 | 0.636 |
| | Lovász + **post-proc** | 0.987 | 0.591 | 0.723 | 0.547 | **0.712** |

Table 4.10: Comparison between the mean dice scores per class and total, before and after the post-processing as describe in section 3.2.3 in the validation sets.

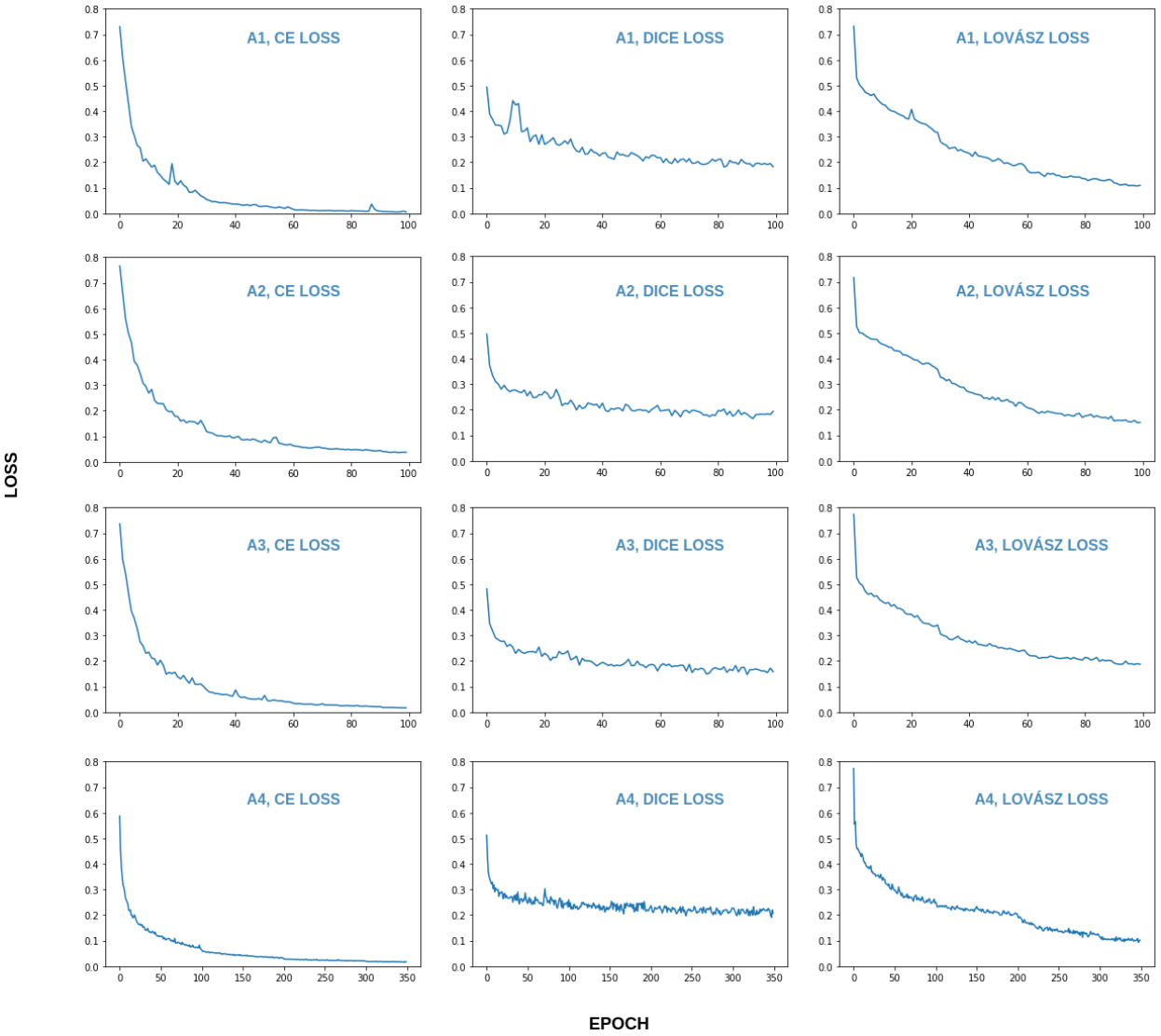



Figure 4.12: The loss function value for each one of the three losses and four datasets.

performed a series of preprocessing operations in order to put them in axial view and shapes equal to the first 11 volumes as explained and showed in Sections 3.1 and 4.3.1.

Figures 4.18, 4.19 and 4.20 shows the results for the network trained with cross entropy (CE), dice and Lovász losses, respectively. Each row in the figures present one volume, and each column shows the ground truths and corresponding predictions for a certain slice of such volume. In the prediction slices (second and fourth column of each figure), it is possible to see some smaller regions that are mis-segmentations. Most of them were given the cerebellum label (yellow), but there are some belonging to the brain stem as well (green).

Generally, when comparing to the ground truth the predictions seem to be approximating the desired shapes. In Figure 4.18, the first and third volumes predictions are the ones that are more similar to the respective ground truths. For the second volume, there seem to be more evident differences, for example, the fourth ventricle is not well predict (see row two, column four of the figure) and there are more holes present in the cerebellum region. These characteristics can be seen in Figure 4.19 as well, with the main difference that there is no fourth ventricle prediction, as it has already been discussed. For Figure 4.20, it is possible

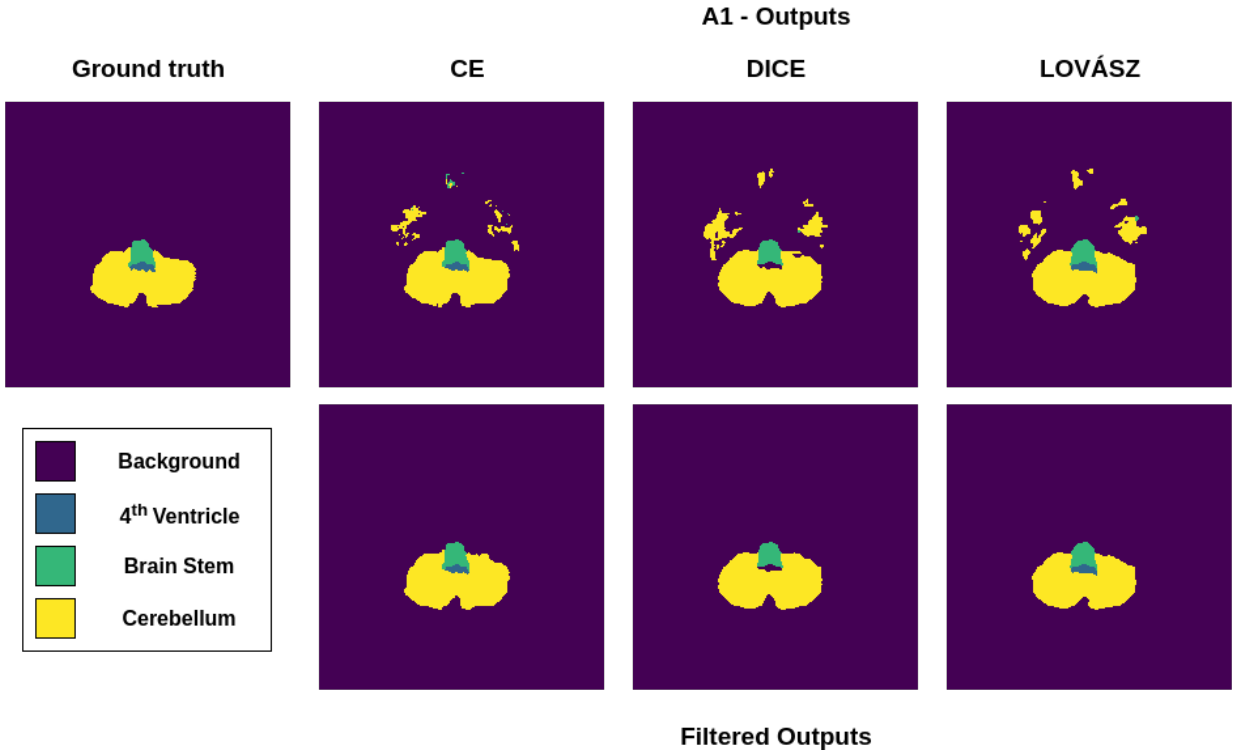



Figure 4.13: Output from model trained with each one of the three losses without (upper row) and with post-processing (bottom row) for images from the validation set of dataset A1.

to verify that it displays more holes in the second and third volumes predictions (see column four for rows two and three of the figure). Moreover, the brain stem predictions for the second volume were not satisfactory (see row two and column two).

In order to try to mitigate some of these issues, namely the smaller mis-segmentations and holes, we applied the largest connected components (CC) and closing filters to the predictions obtained, as explained in Section 3.2.3. In Figures 4.21, 4.22 and 4.23, we see the results after the CC filter was applied to the predictions on the validation set of A4 of the models trained with CE, dice and Lovász losses, respectively. It is possible to check that after this operation, all the smaller regions that were mis-segmentation are removed from the predictions (see specially column four for the three figures).

Following the CC filter, we applied the closing operation with a cubic structuring element with size $3 \times 3 \times 3$. Figures 4.24, 4.25 and 4.26 show the results after the application of the filter for the predictions obtained with models trained with CE, dice and Lovász losses, respectively. As we can notice, all the holes of the predictions were closed in the first two figures (Figures 4.24 and 4.25). In the third one (Figure 4.26), there are still some holes left in the second and third volumes. This happened because the holes were bigger than the size of the structuring element. However, we noticed that if we used bigger elements, the dice-score of the predictions were decreasing when comparing to the previous step, because the closing operation was starting to excessively extrapolate the predictions. Therefore, we decided to use the $3 \times 3 \times 3$ structuring element, even tough some holes still remained.

For the A4 dataset, we were also able to isolate three volumes to serve as test set. Table 4.12 shows the results of the models trained with the A4 dataset in the test set per

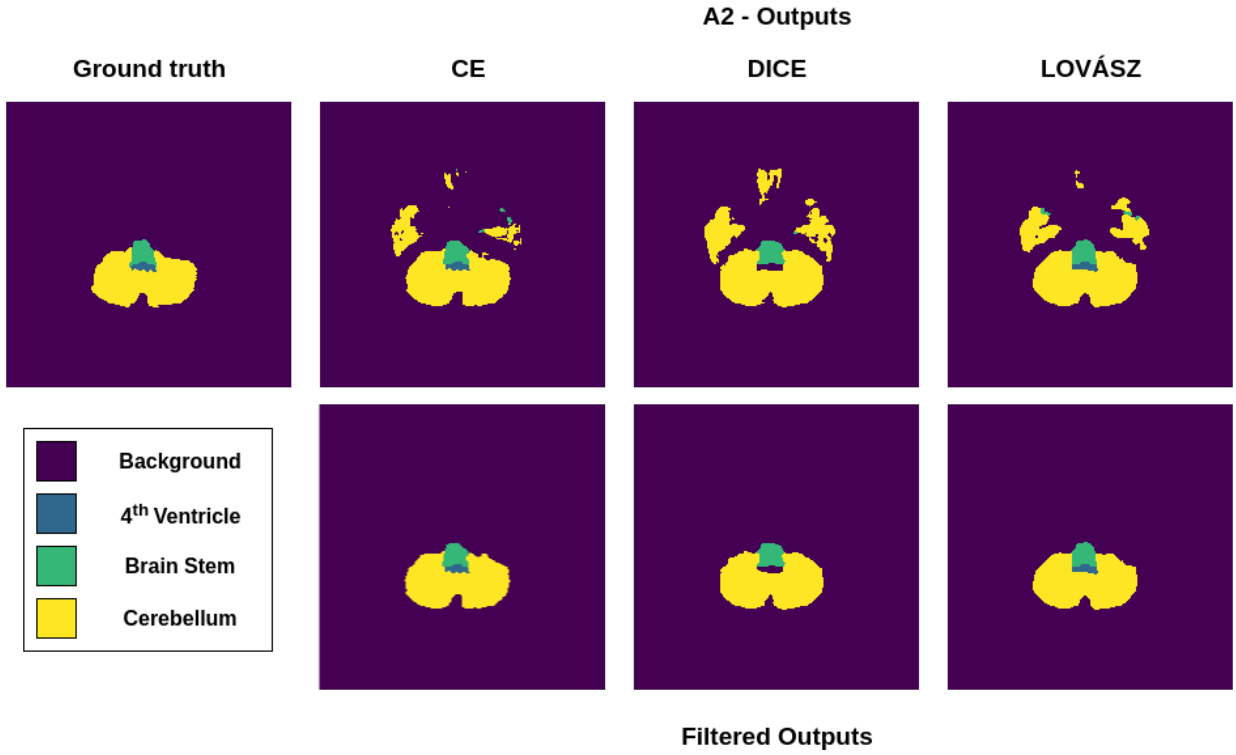



Figure 4.14: Output from model trained with each one of the three losses without (upper row) and with post-processing (bottom row) for images from the validation set of dataset A2.

volume. We can notice how badly the models were when identifying the cerebellum for the volume 30 –what does not happen for the number 5 and 7 images. Also, the results for the other structures is overall worse for this volume than the other two.

In Figures 4.27, 4.28 and 4.29, we show the results of the models trained with dataset A4 and loss functions cross-entropy, dice and Lovász, respectively on A4 test set. Each row contains different slices from a certain volume. As it is possible to check the third row of the pictures is very different from the first two rows. Such line corresponds to the predictions of the volume 30. For example, in the first two column there is no prediction or ground truth for this patient's exam. Both the prediction and ground truth of the first two volumes in the figures are very similar to the ones in the validation dataset. This indicates that volume 30 is possibly an outlier (which means it is very different from the other 31 volumes that compose the dataset).

Figures 4.36, 4.37, 4.38 and 4.39 show the comparison between slices from volume 30 and another patient with similar age, both from the axial and sagittal views. It is possible to check that patient 30 was in an offset position in the table when compared to other volumes in the dataset. Also, there are some hyperintense (white) regions in the cerebellum area of volume 30, that are different from the cerebellum of the other patients, which could explain the worse performance of the model for this volume.

Moreover, Figures 4.30, 4.31 and 4.32 display the results of the predictions (A4 test set) obtained by the models trained with dataset A4 and cross-entropy, dice and Lovász losses, respectively. These predictions were obtained after the first step of the post-processing pipeline, namely the connected components filter. It is possibly to see how it was successful

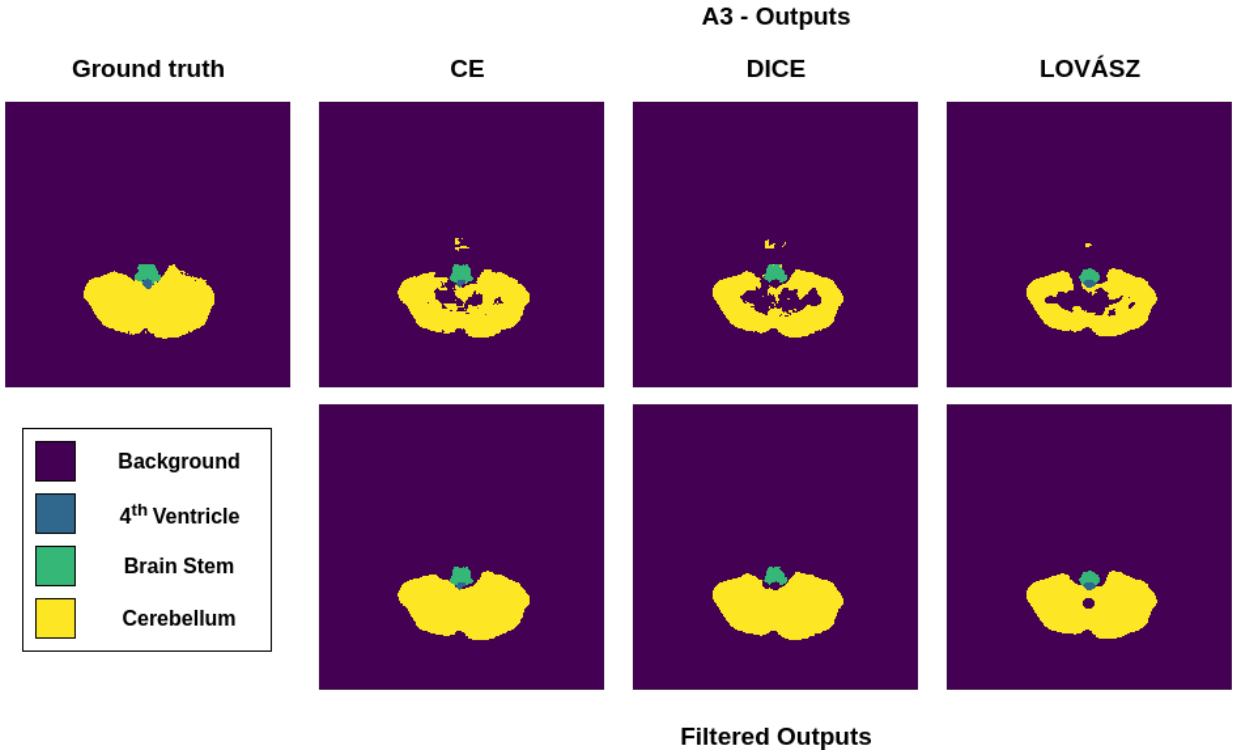



Figure 4.15: Output from model trained with each one of the three losses without (upper row) and with post-processing (bottom row) for images from the validation set of dataset A3.

in removing the smaller region predictions for the CE and Lovász losses, specially. For the results regarding the model trained with dice loss, in the first row and fourth column, the wrong region of the cerebellum was kept. Fortunately this situation is very uncommon, and the post-processing was able to increase the average dice score for all classes and volumes. However, the filter should be improved in order to avoid it and improve even more the prediction. One idea would be to check the cerebellum components centroids and keep the ones that are the biggest and have the centroid closer to the lower part of the slice. This can be done because usually the cerebellum is localized in the lower half of the image. We suggest such investigation as future work.

Finally, in Figures 4.33, 4.34 and 4.35, we display the closing step of the post-processing on the predictions on the volumes from A4 test set, of the models trained with cross-entropy, dice and Lovász losses, respectively. Overall the filter managed to close the holes from the predictions from rows one and two of Figure 4.33. Similarly, it worked well for the volumes obtained from the model trained with dice loss. Although, likewise, what happened in the validation dataset, some holes remain for the results from the model trained with Lovász loss, specially in row one.

Therefore, it is possible to conclude that the post-processing techniques used in this work were able to improve the predictions produced by the network independently of which loss was used to train it. For instance, Table 4.10 show that the mean dice score increased about 0.1 for the results of the validation dataset of A4, when comparing the raw prediction with the post-processed one. Also when analysing Table 4.11, we can notice how well the pipeline worked for the cerebellum and brain stem segmentation predictions of volumes 5 and 7 –

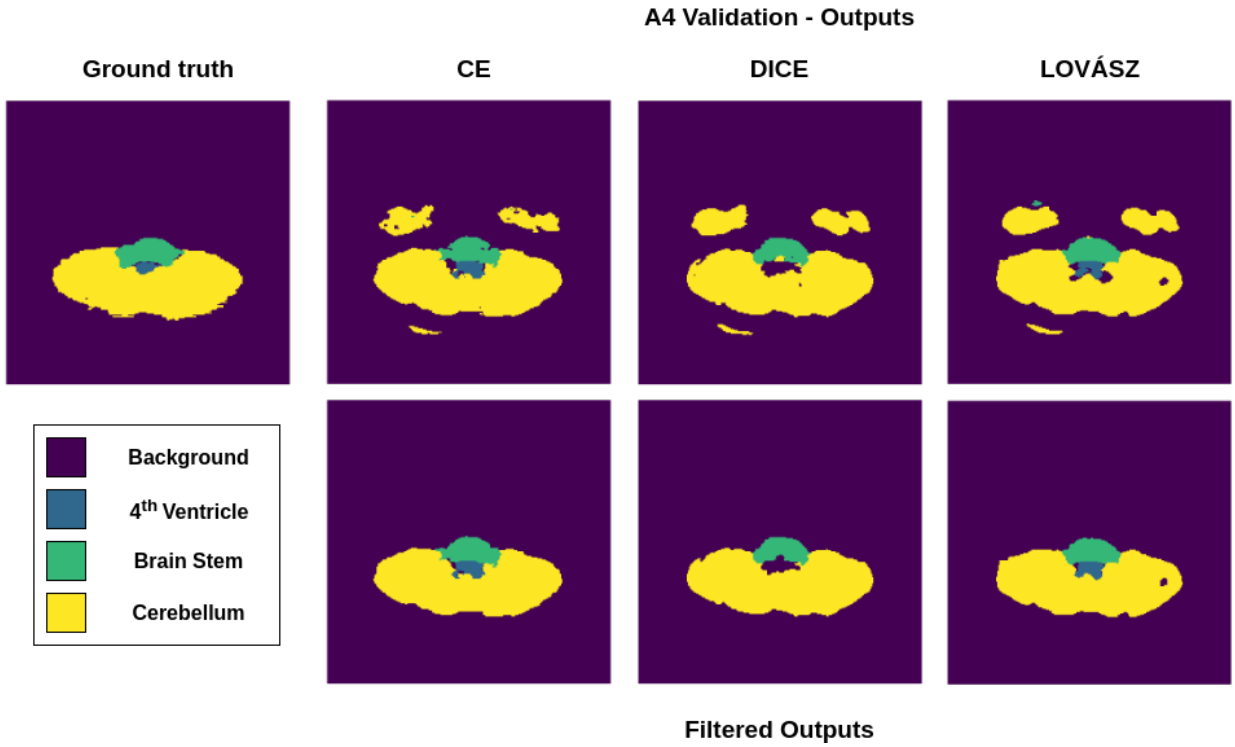



Figure 4.16: Output from model trained with each one of the three losses without (upper row) and with post-processing (bottom row) for images from the validation set of dataset A4.

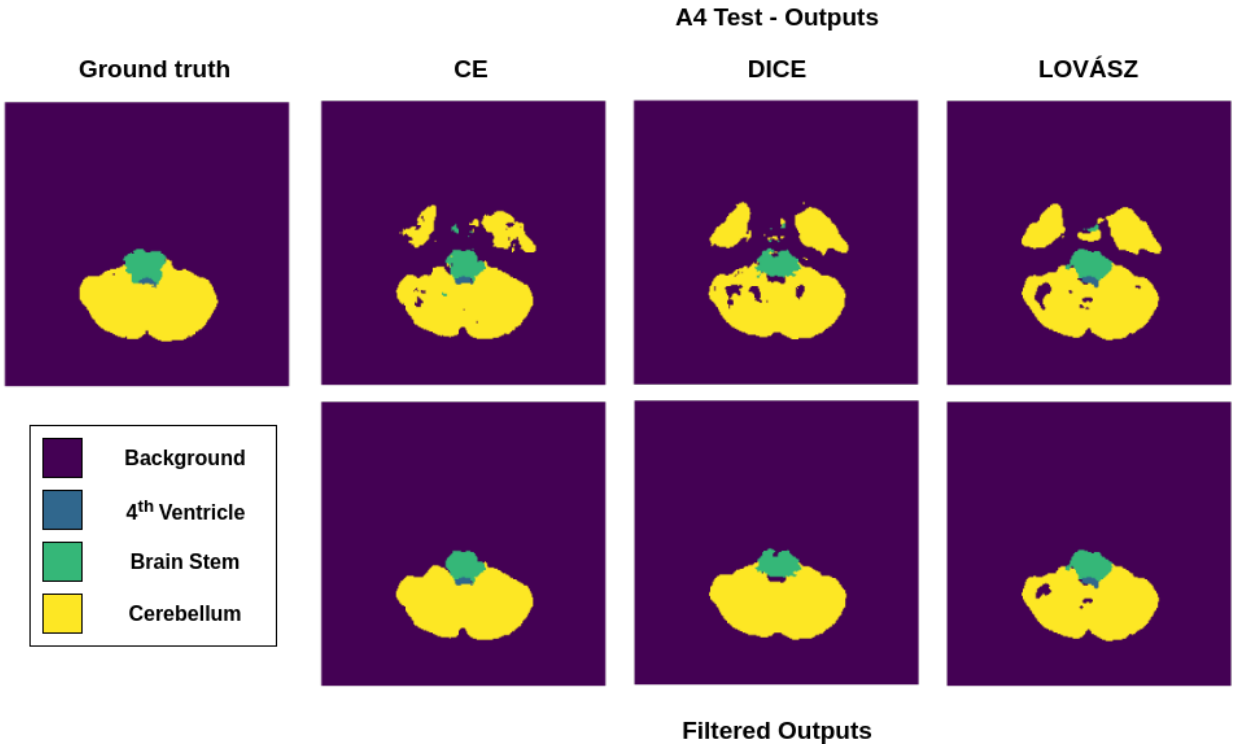



Figure 4.17: Output from model trained with each one of the three losses without (upper row) and with post-processing (bottom row) for images from the test set of dataset A4.

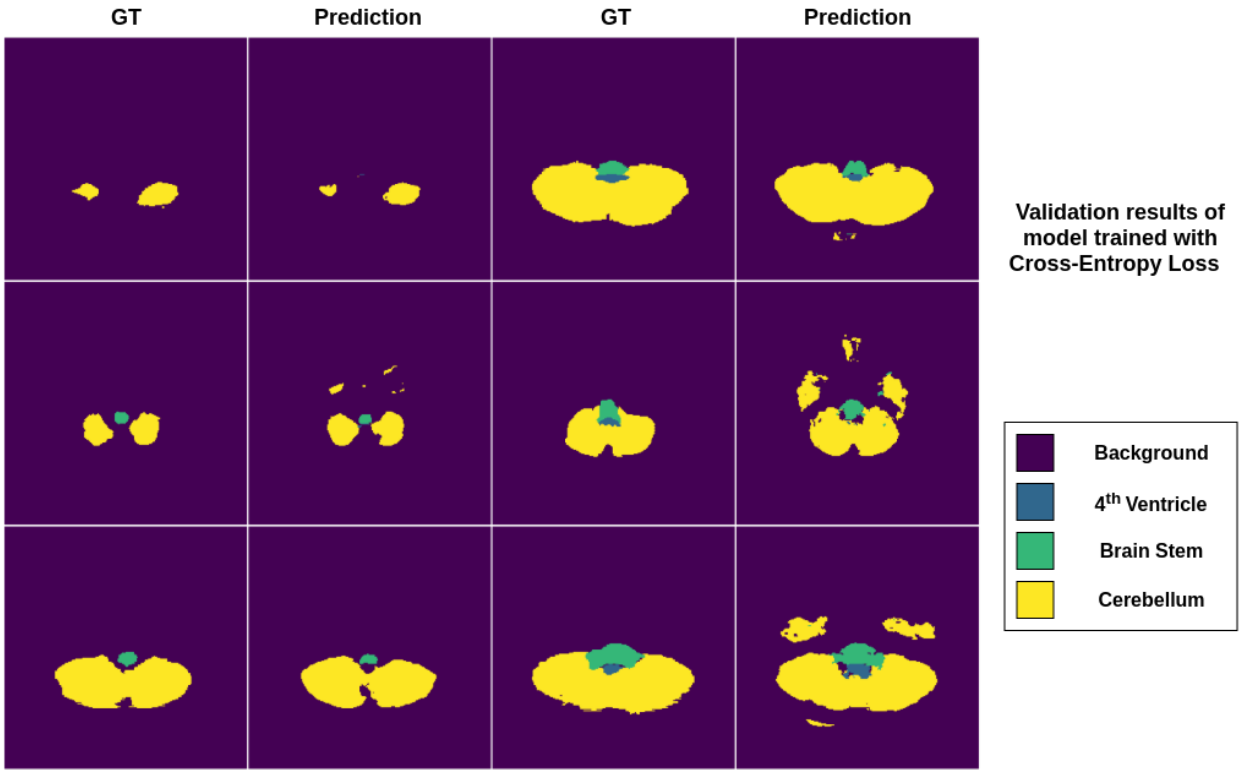



Figure 4.18: A4 Validation results of model trained with cross-entropy loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

with dice scores higher than 0.8 and 0.7 for each of the mentioned classes, respectively. This shows that the network performed well, considering the age and volume orientation variability present in the A4 dataset.

| Dataset | Loss | Background | $4^{th}$ Ventricle | Brain Stem | Cerebellum | Total |
|---|---|---|---|---|---|---|
| A4 - Test | CE | 0.982 | 0.534 | 0.639 | 0.359 | 0.628 |
| | CE + **post-proc** | 0.994 | 0.618 | 0.745 | 0.580 | **0.734** |
| | Dice | 0.976 | 0.000 | 0.654 | 0.309 | 0.485 |
| | Dice + **post-proc** | 0.989 | 0.000 | 0.735 | 0.481 | **0.551** |
| | Lovász | 0.975 | 0.607 | 0.480 | 0.302 | 0.591 |
| | Lovász + **post-proc** | 0.990 | 0.626 | 0.597 | 0.473 | **0.672** |

Table 4.11: Mean dice scores per class and total, before and after the post-processing as describe in section 3.2.3 in the test set of the A4 dataset.

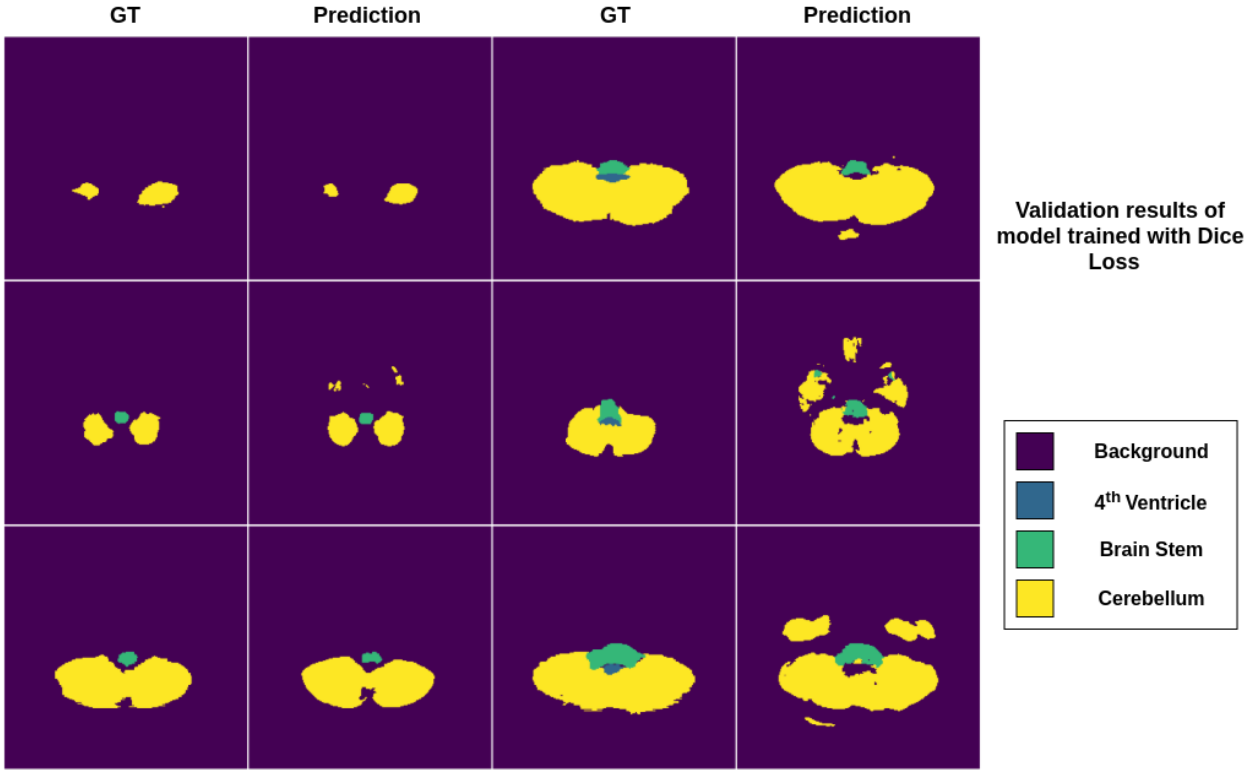



Figure 4.19: A4 Validation results of model trained with dice loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

| Volume | Loss | Background | $4^{th}$ Ventricle | Brain Stem | Cerebellum | Total |
|---|---|---|---|---|---|---|
| 5 | CE | 0.996 | 0.643 | 0.747 | 0.834 | 0.805 |
| | Dice | 0.991 | 0.000 | 0.707 | 0.667 | 0.591 |
| | Lovász | 0.990 | 0.686 | 0.415 | 0.673 | 0.691 |
| 7 | CE | 0.997 | 0.705 | 0.874 | 0.849 | 0.856 |
| | Dice | 0.994 | 0.000 | 0.851 | 0.718 | 0.640 |
| | Lovász | 0.993 | 0.751 | 0.740 | 0.695 | 0.795 |
| 30 | CE | 0.988 | 0.507 | 0.613 | 0.058 | 0.541 |
| | Dice | 0.981 | 0.000 | 0.646 | 0.059 | 0.422 |
| | Lovász | 0.985 | 0.441 | 0.635 | 0.052 | 0.528 |

Table 4.12: The performance of the models trained with A4 in each model of the test set after post-processing.

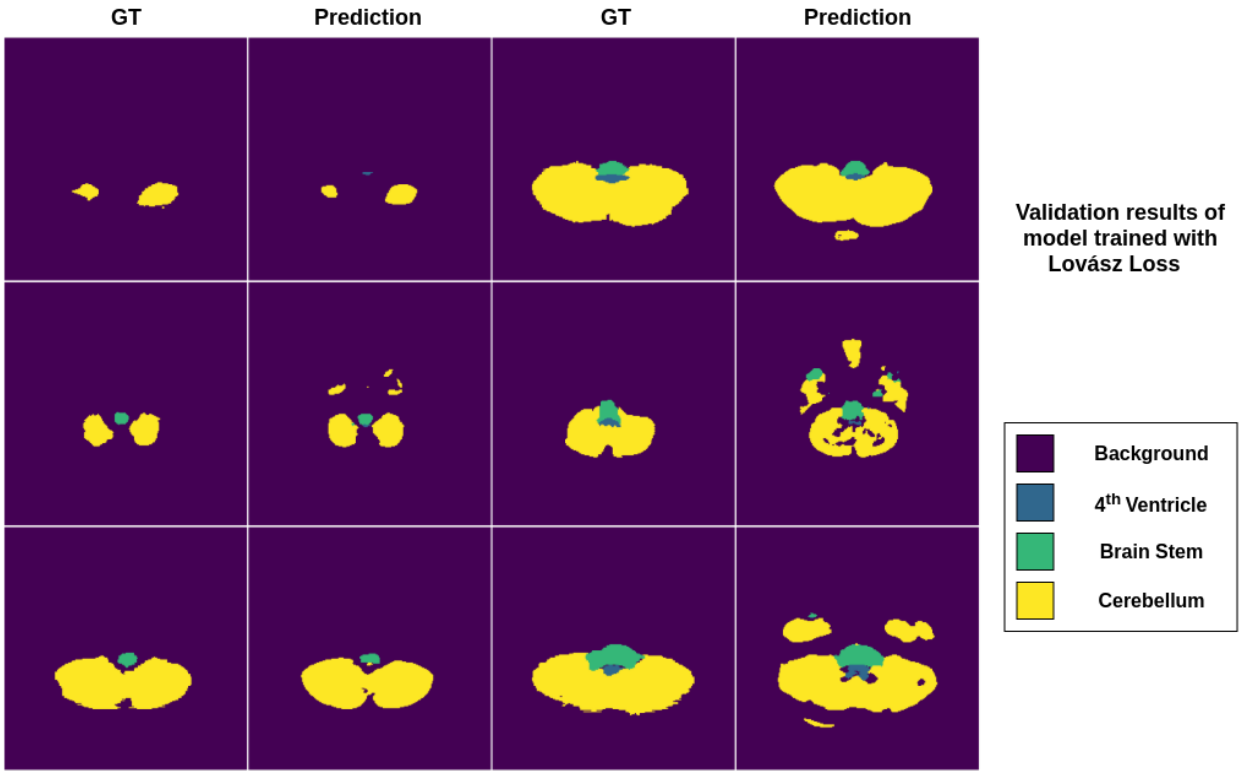



Figure 4.20: A4 Validation results of model trained with Lovász loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

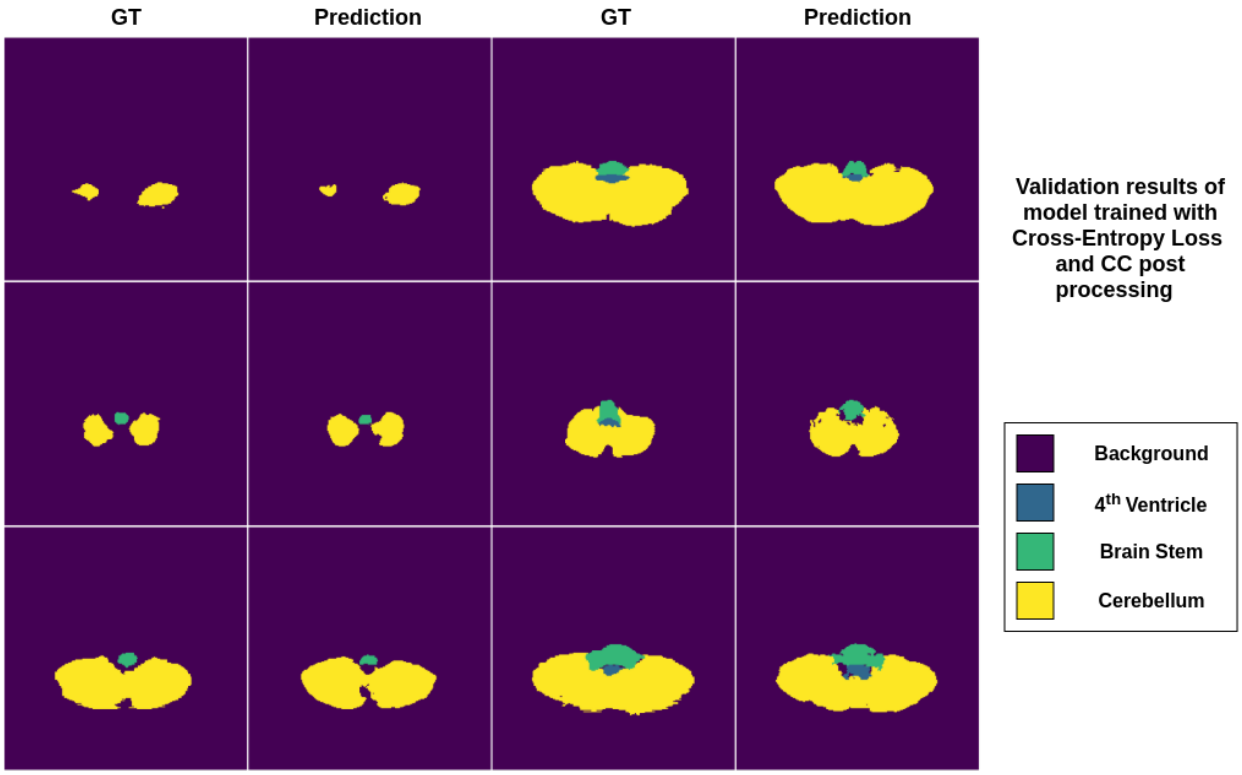



Figure 4.21: A4 Validation results of model trained with cross-entropy loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

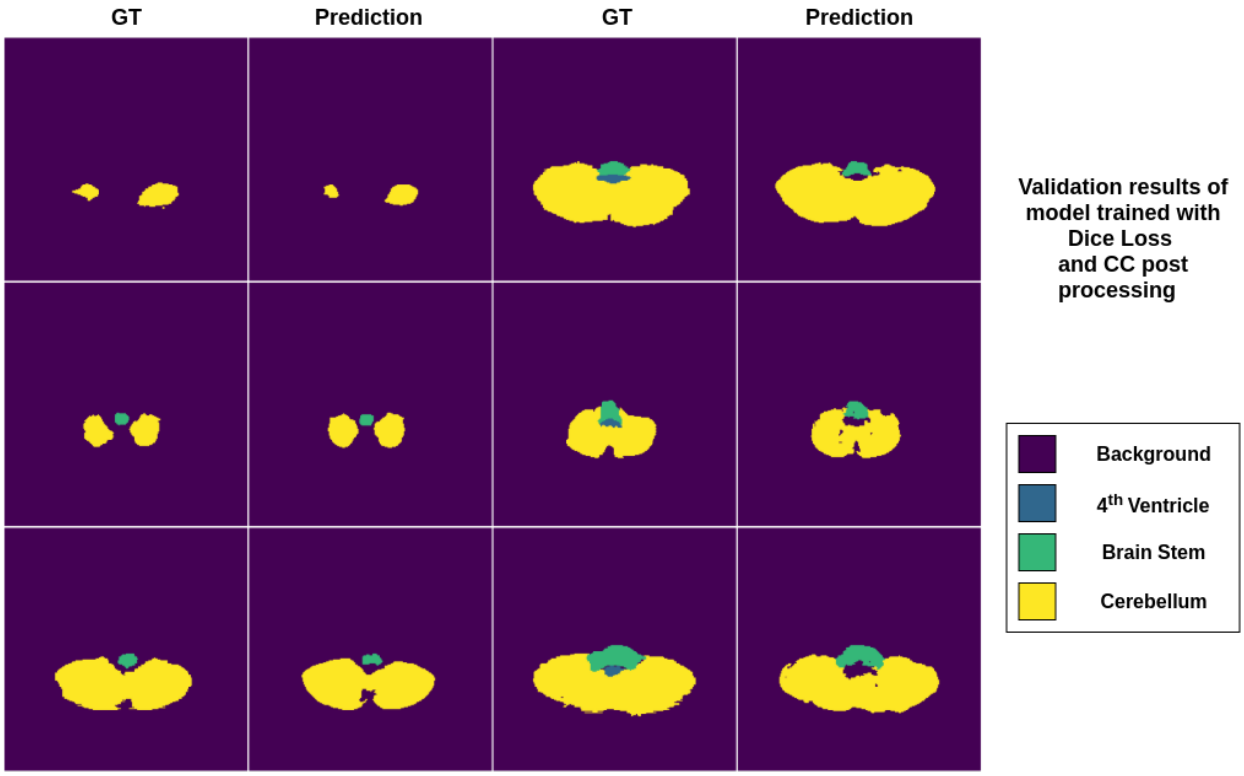



Figure 4.22: A4 Validation results of model trained with dice loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

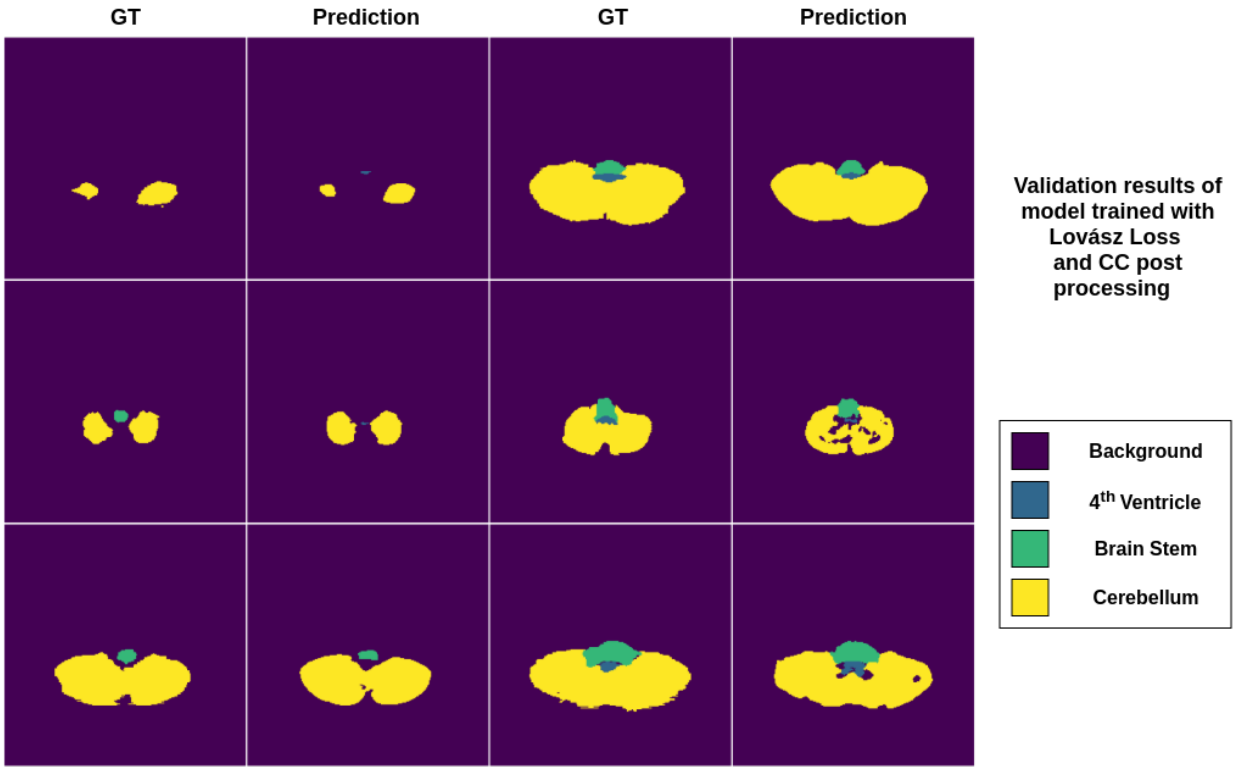



Figure 4.23: A4 Validation results of model trained with Lovász loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

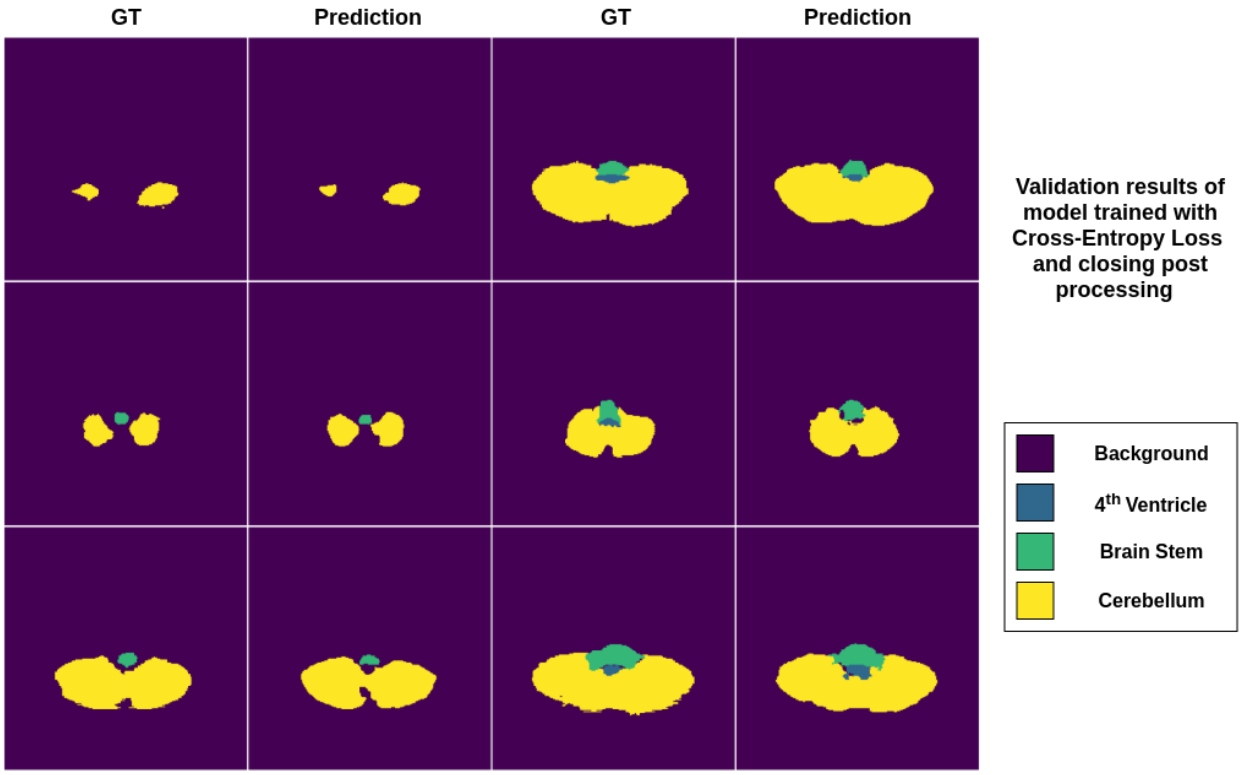



Figure 4.24: A4 Validation results of model trained with cross-entropy loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

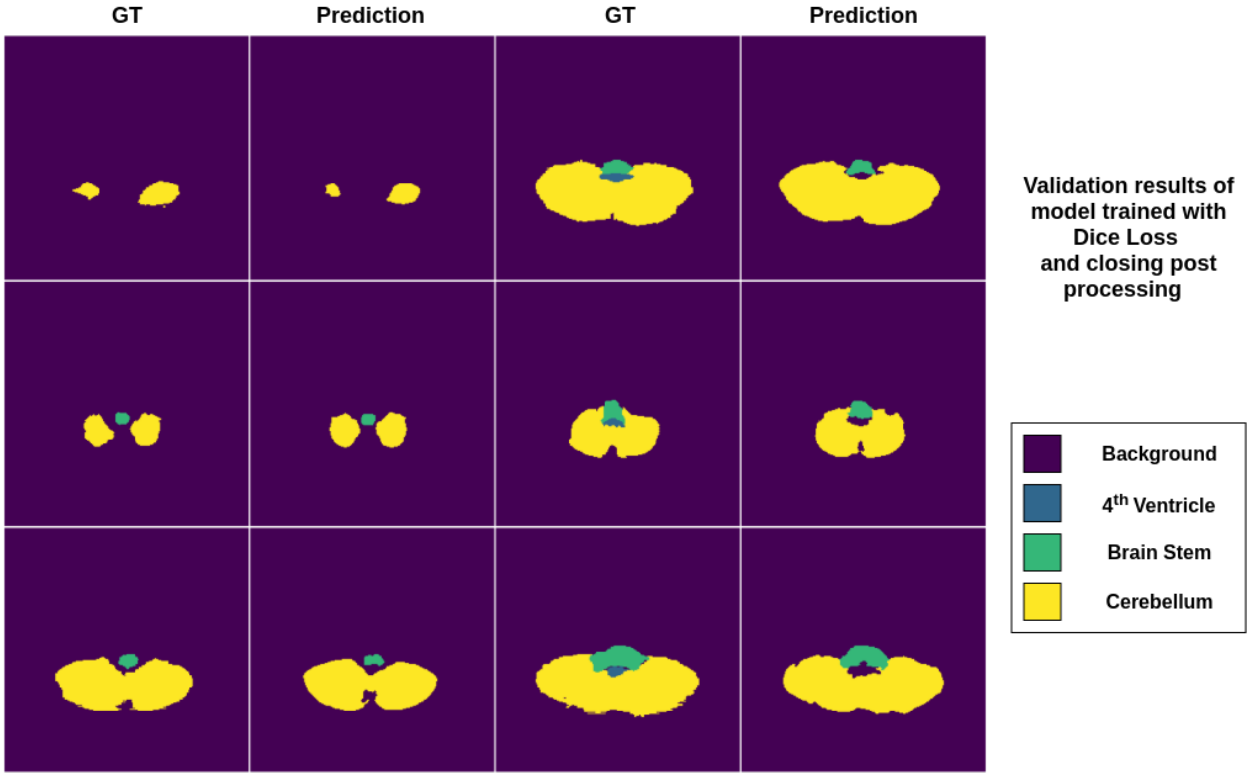



Figure 4.25: A4 Validation results of model trained with dice loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

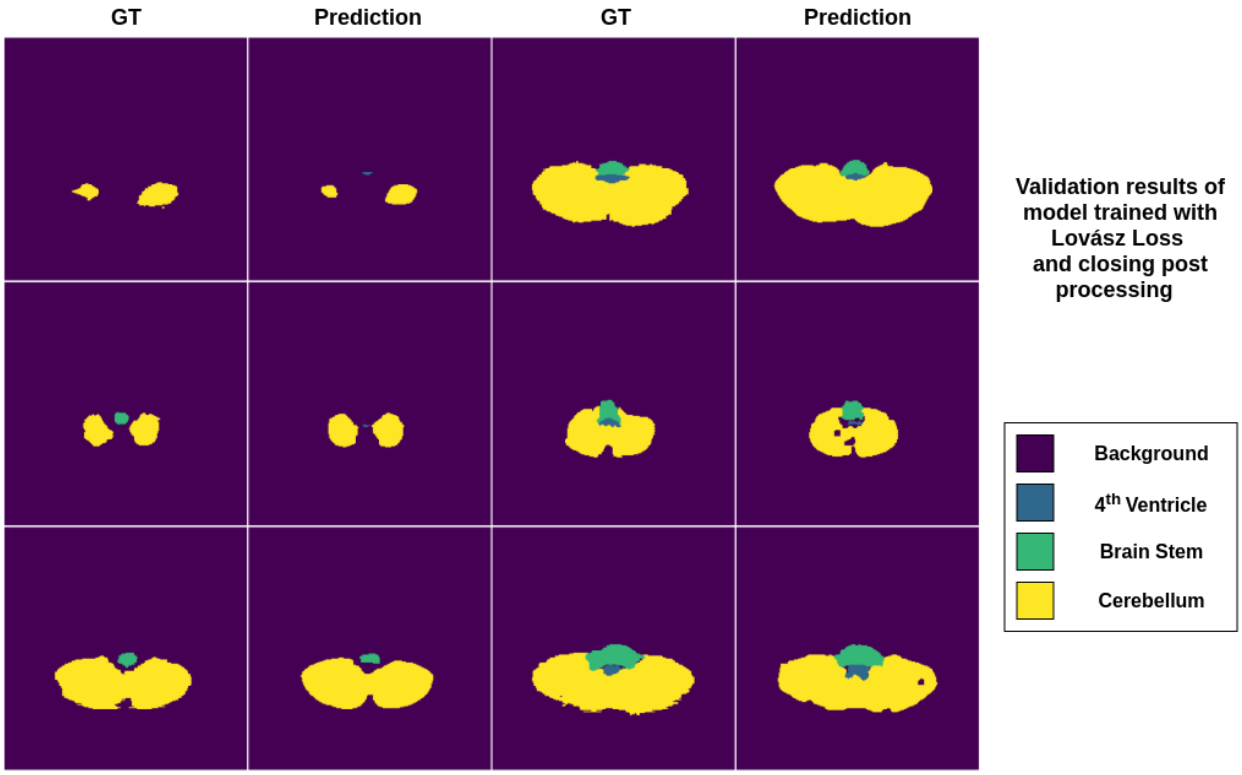



Figure 4.26: A4 Validation results of model trained with Lovász loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three validation volumes).

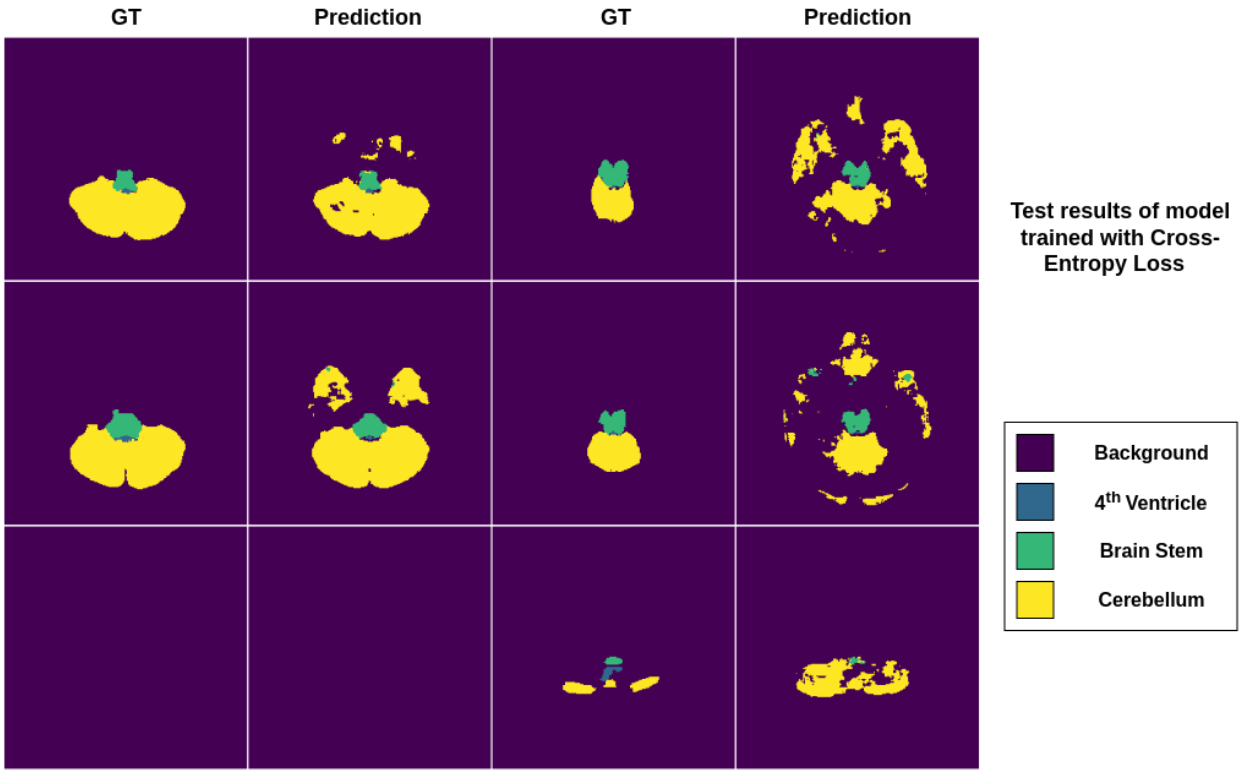



Figure 4.27: A4 Test results of model trained with cross-entropy loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

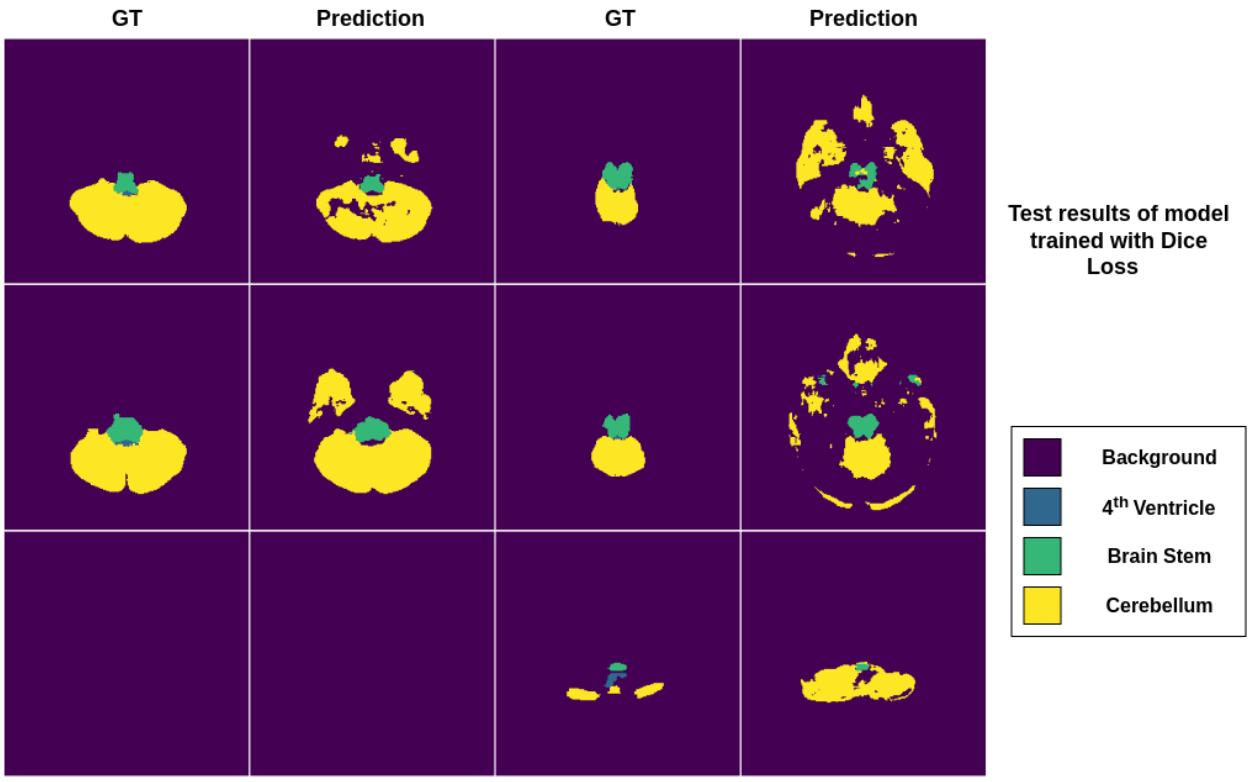



Figure 4.28: A4 Test results of model trained with dice loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

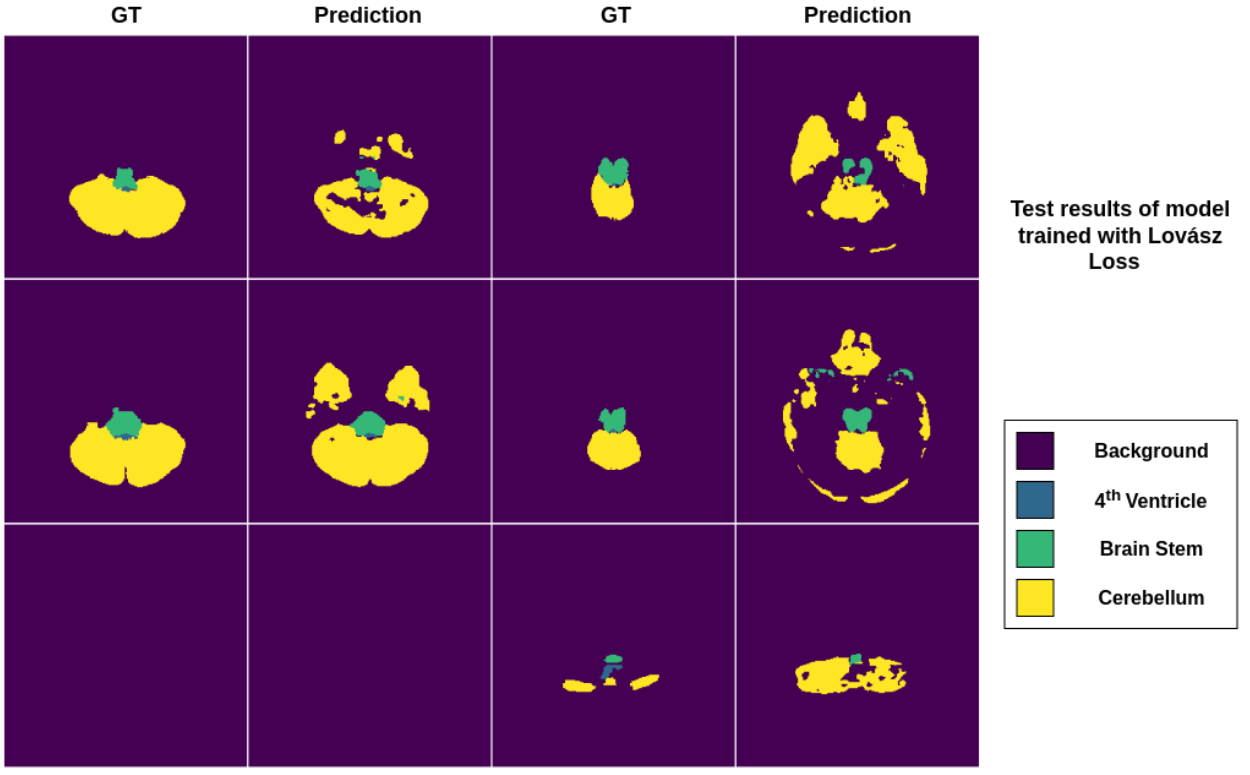



Figure 4.29: A4 Test results of model trained with Lovász loss. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

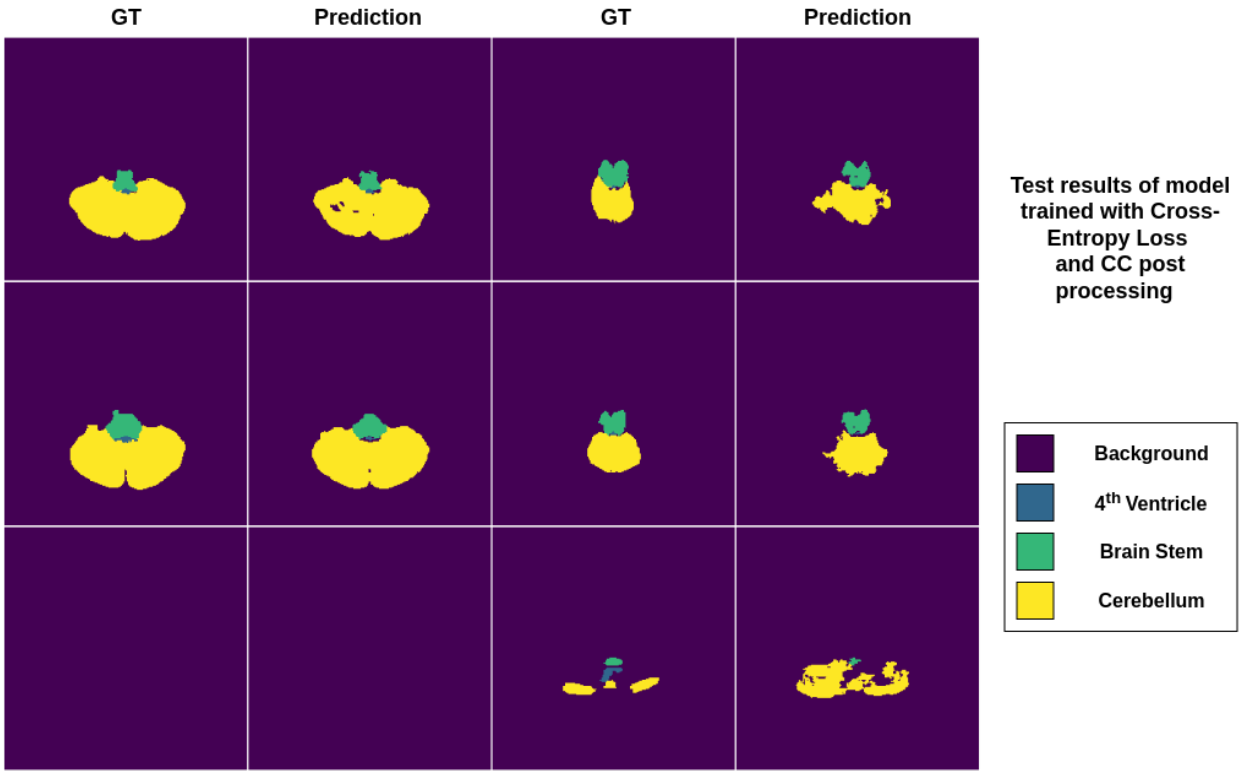



Figure 4.30: A4 Test results of model trained with cross-entropy loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

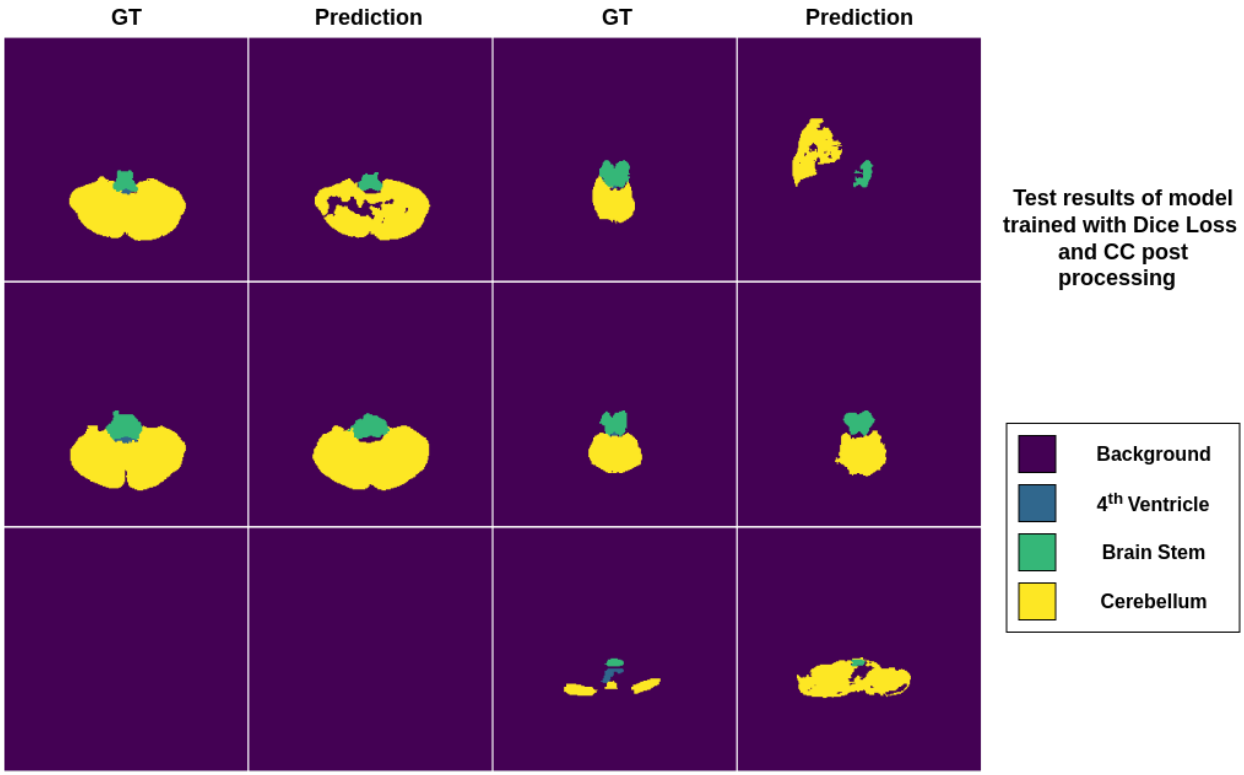



Figure 4.31: A4 Test results of model trained with dice loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

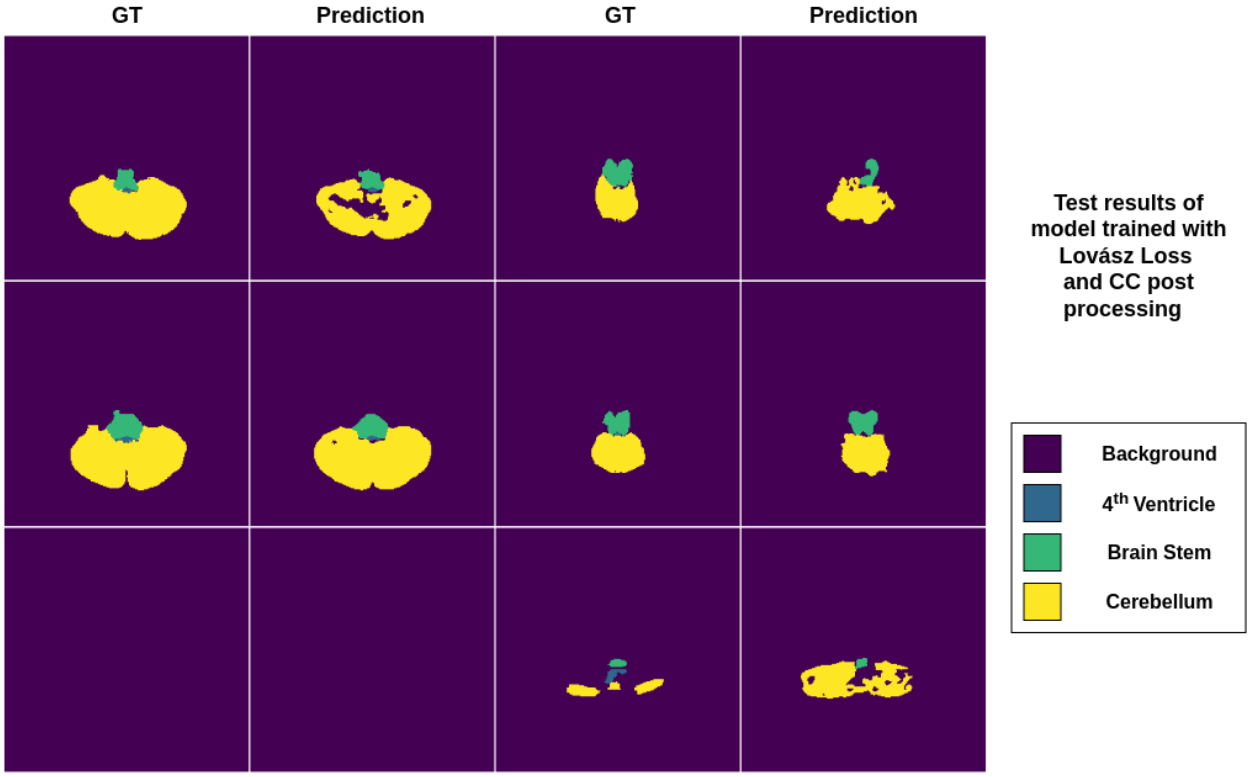



Figure 4.32: A4 Test results of model trained with Lovász loss after the connect-components filter. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

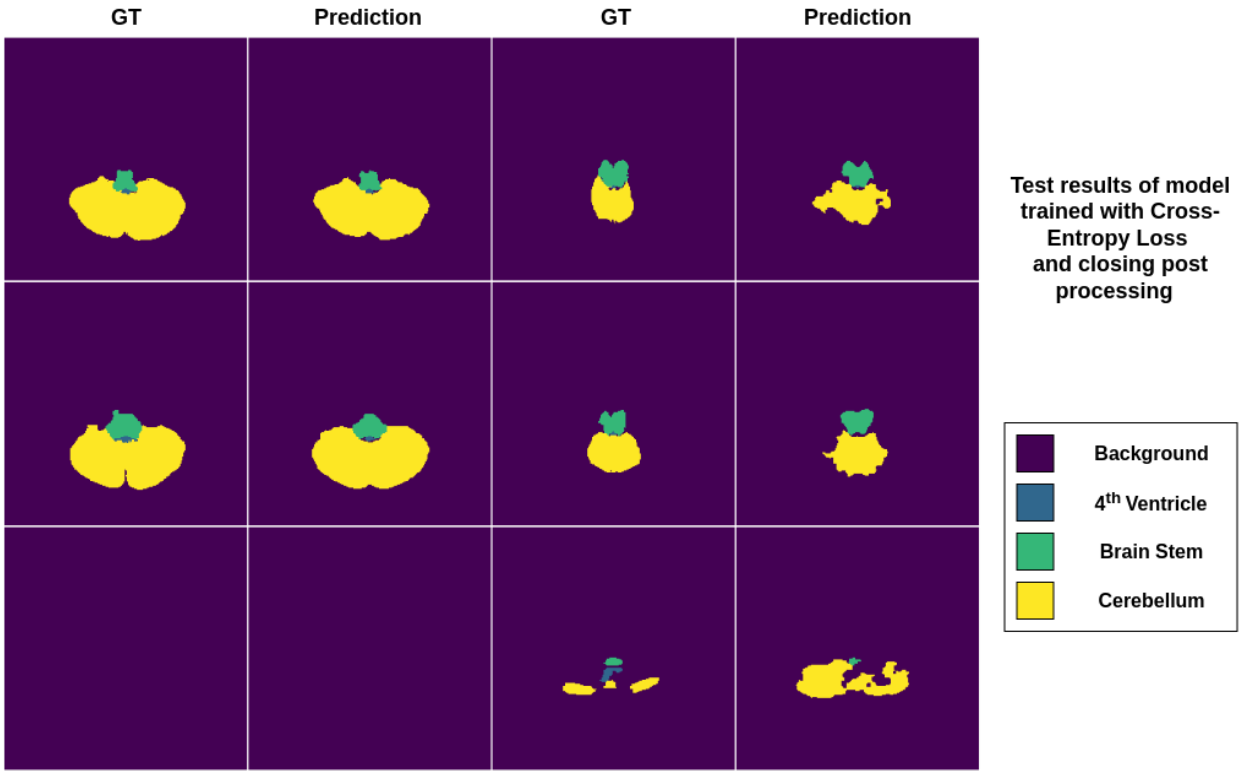



Figure 4.33: A4 Test results of model trained with cross-entropy loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

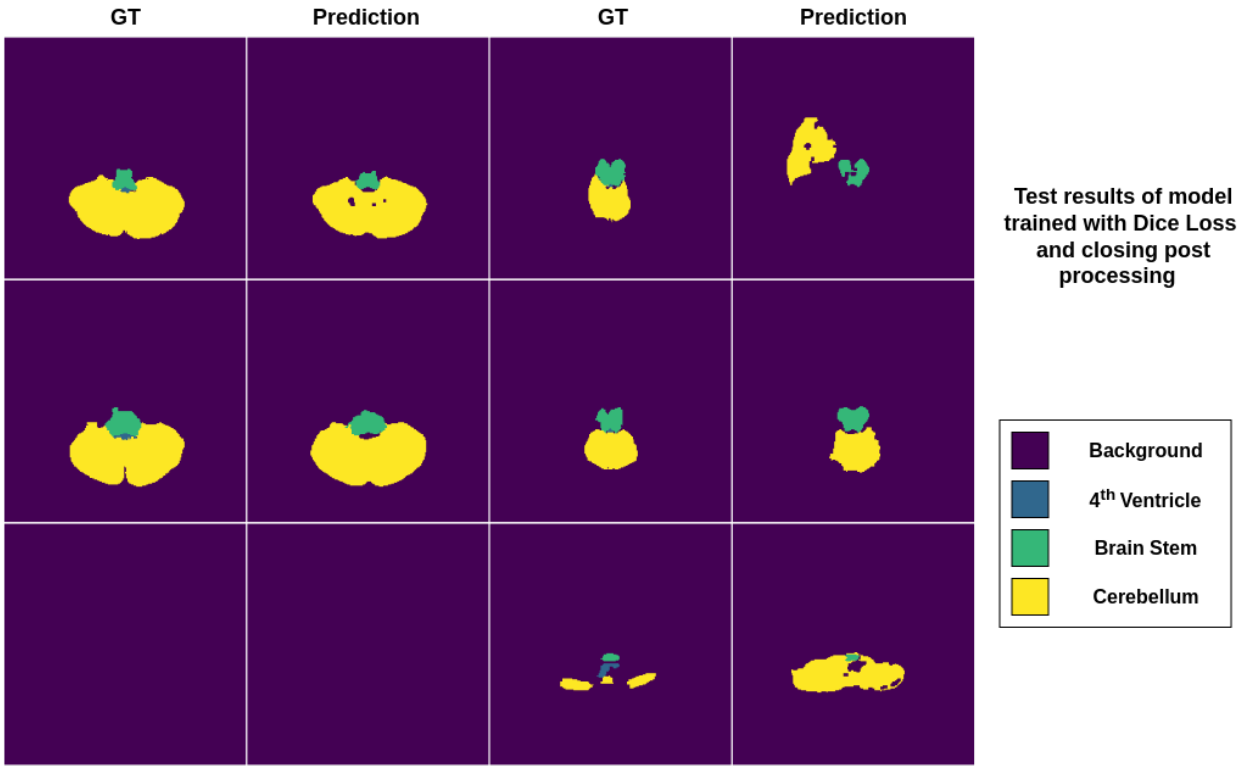



Figure 4.34: A4 Test results of model trained with dice loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

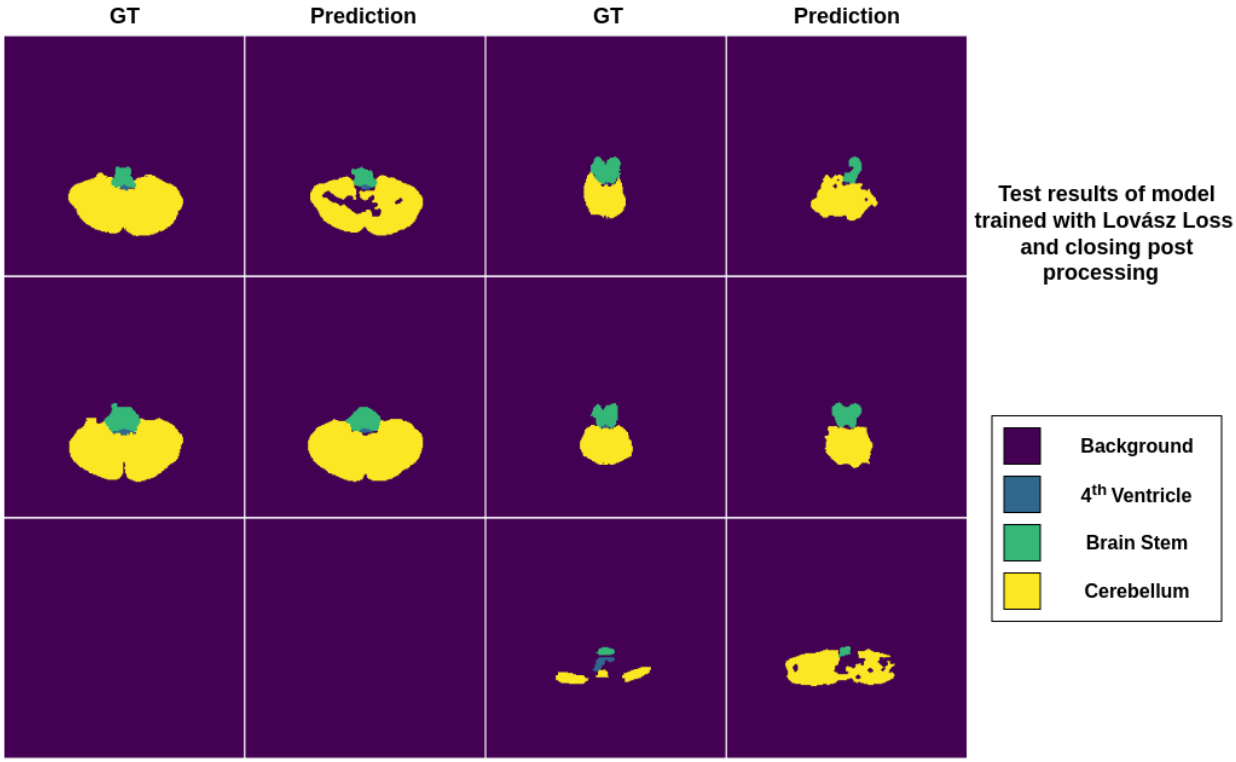



Figure 4.35: A4 Test results of model trained with Lovász loss after the connect-components and closing filters. In columns one and three we have the ground truth slices and in columns two and four, the respective predictions produced by the network. Each row corresponds to slices from a different patient (we have three test volumes).

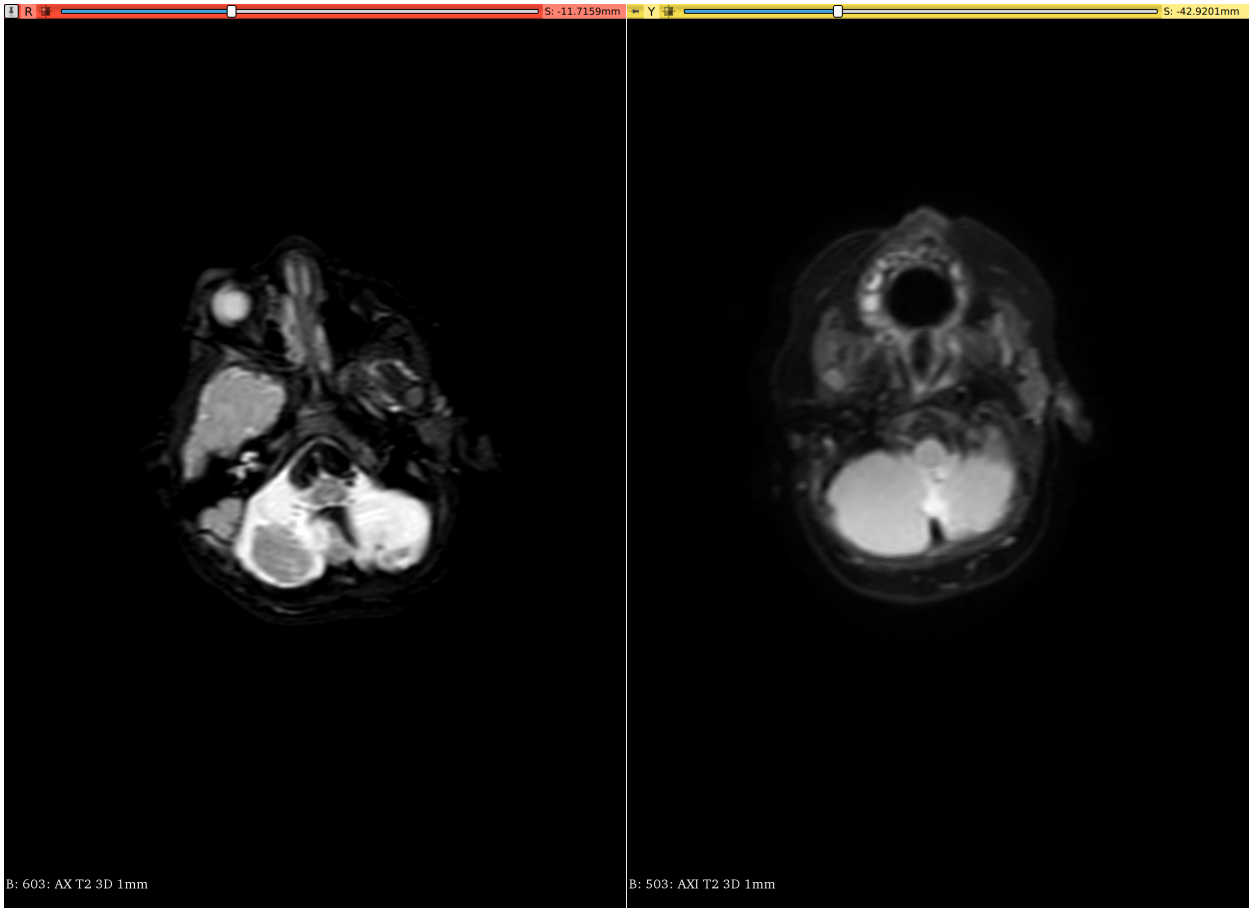



Figure 4.36: Comparison between volume 30 (left) and another volume (right) from the dataset with similar age and at the same 'height' in the axial view.


Figure 4.37: Comparison between volume 30 (left) and another volume (right) from the dataset with similar age in sagittal view. Notice the how the patient position is different from the one in the right. Also, there are hyperintense regions present in the cerebellum of patient 30.

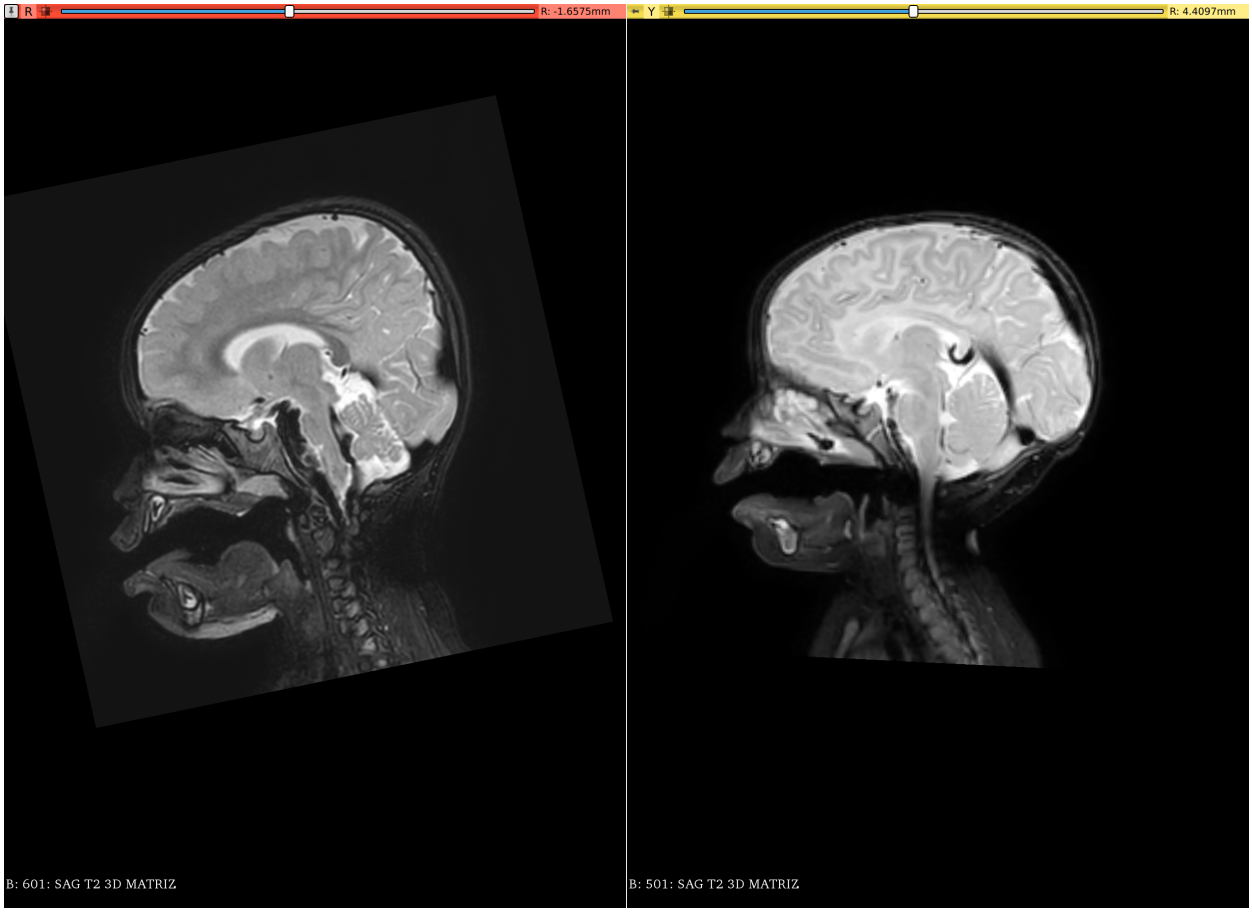



Figure 4.38: Comparison between volume 30 (left) and another volume (right) from the dataset with similar age in sagittal view. Notice the how the patient position is different from the one in the right. Also, there are hyperintense regions present in the cerebellum of patient 30.

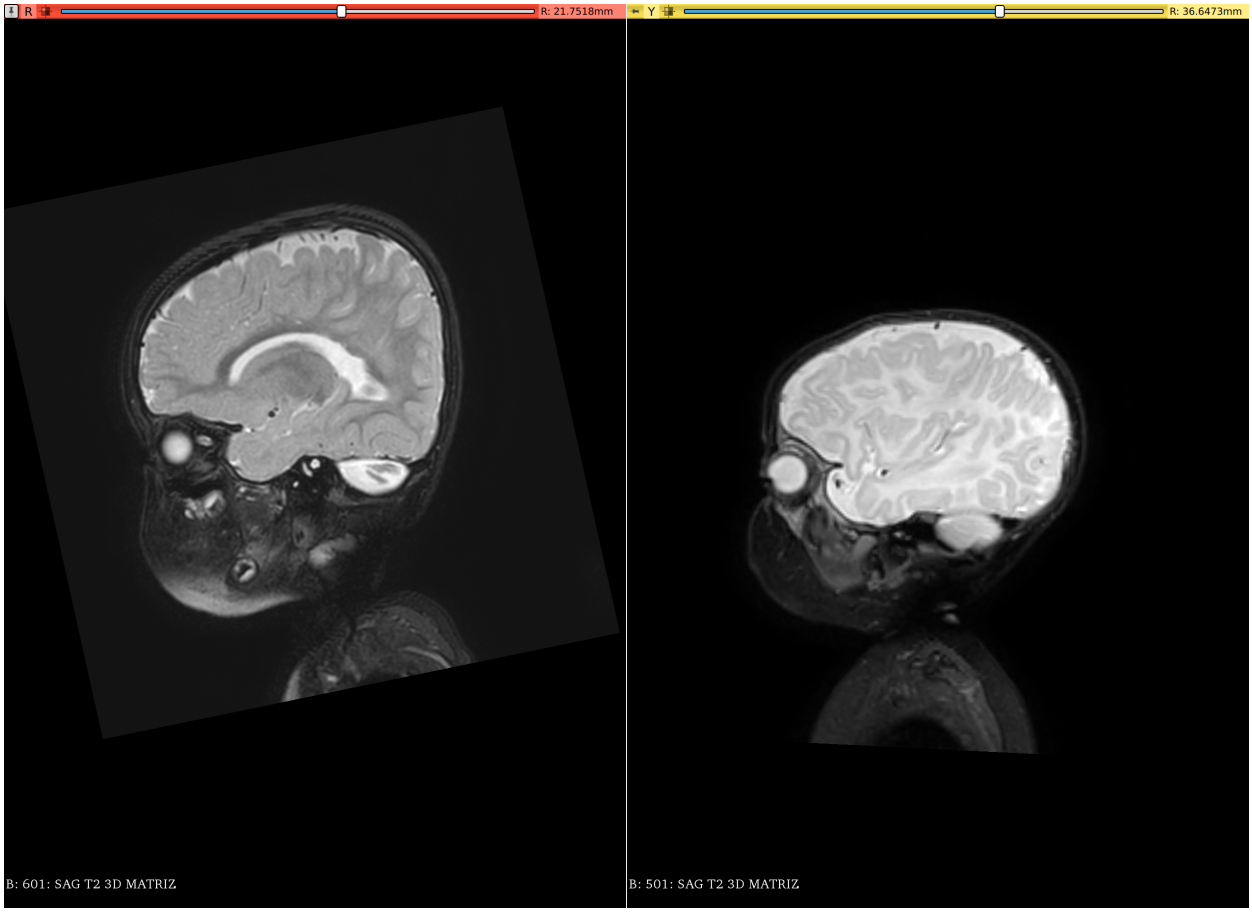



Figure 4.39: Comparison between volume 30 (left) and another volume (right) from the dataset with similar age in sagittal view. Notice the how the patient position is different from the one in the right. Also, there are hyperintense regions present in the cerebellum of patient 30.

# Chapter 5

# Conclusion

## 5.1 Summary

The analysis of volumetric image exams, such as MRIs, is important to the diagnostic, prognostic and treatment of various diseases. When radiologists perform the manual segmentation of such images, it is both time consuming and prone to analyzer-bias. In order to alleviate such issues, automatic and semi-automatic methods have been proposed during the past decades. Most of the developed algorithms for brain segmentation were designed to analyse the exams of adult and healthy patients or for adults with specific and predetermined diseases.

The analysis of MRI brain exams of fetal, neonates and pediatric patients presents some specific challenges as the smaller size of the brain structures, the inverted white-grey matter contrast and the high variability due to the development of the brain (which occur specially during the two first years of life). In order to deal with these issues, methods have been developed for such data, specifically. The usage of such techniques is not trivial, since usually the methods and datasets used are not publicly available.

In this work, we have developed a pipeline to perform the semantic segmentation of a region that includes the posterior fossa (composed of the brain stem and cerebellum) and the fourth ventricle. The importance of identifying these structures is related to the fact that it is the region of occurrence of the medulloblastoma – the most common cancer that affects mainly pediatric patients. There were no automatic methods to perform the automatic segmentation of the said region or tumor, until the conclusion of this project.

The pipeline presented in this work included the pre-processing of MRI data, the segmentation task performed by a neural network called LiviaNet, and finally, a post-processing step. After applying the pipeline, the method was able to achieve an average score dice of 0.74 for the validation set and 0.734 for the test set (and 0.8305 if we do not consider the outlier volume of patient 30). This shows that the method has potential, but there is still room for improvements, which will be listed in the Future Work section.

## 5.2 Future Work

### 5.2.1 3D Pediatric Segmentation

As future work we suggest investigating the different weighting techniques for the dice loss, in order to compensate for multi-class imbalance. The post-processing technique can also be improved. As cited in the previous chapter, it is possible to refine the closed components step for the cerebellum by considering the centroid of the regions being analysed. Since in the axial view, the cerebellum is usually localized in the lower half (vertically) of the slices. Also, using more powerful GPUs, it should be interesting training some 3D semantic segmentation neural networks (such as V-NET) and evaluate if the result is better, since we can preserve more spatial information with this kind of network, it could avoid the need of the post-processing techniques used in this work.

# Bibliography


[1] Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282.

[2] Anbeek, P., Išgum, I., van Kooij, B. J., Mol, C. P., Kersbergen, K. J., Groenendaal, F., Viergever, M. A., de Vries, L. S., and Benders, M. J. (2013). Automatic segmentation of eight tissue classes in neonatal brain mri. *PLoS One*, 8(12):e81895.

[3] Andrew Ng, Kian Katanforoosh, Y. B. M. (2017). Lecture learning rate decay. `https://www.coursera.org/lecture/deep-neural-network/learning-rate-decay-hjgIA`.

[4] Baytan, B., Evim, M. S., Güler, S., Güneş, A. M., and Okan, M. (2015). Acute central nervous system complications in pediatric acute lymphoblastic leukemia. *Pediatric neurology*, 53(4):312–318.

[5] Berman, M., Rannen Triki, A., and Blaschko, M. B. (2018). The lovász-softmax loss: a tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4413–4421.

[6] Beucher, S. and Meyer, F. (1993). The morphological approach to segmentation: the watershed transformation. *Mathematical morphology in image processing*, 34:433–481.

[7] Black, M. J., Sapiro, G., Marimont, D. H., and Heeger, D. (1998). Robust anisotropic diffusion. *IEEE Transactions on image processing*, 7(3):421–432.

[8] Carass, A., Cuzzocreo, J. L., Han, S., Hernandez-Castillo, C. R., Rasser, P. E., Ganz, M., Beliveau, V., Dolz, J., Ayed, I. B., Desrosiers, C., et al. (2018). Comparing fully automated state-of-the-art cerebellum parcellation from magnetic resonance images. *NeuroImage*, 183:150–172.

[9] Ceschin, R., Zahner, A., Reynolds, W., Gaesser, J., Zuccoli, G., Lo, C. W., Gopalakrishnan, V., and Panigrahy, A. (2018). A computational framework for the detection of subcortical brain dysmaturation in neonatal mri using 3d convolutional neural networks. *NeuroImage*, 178:183–197.

[10] Devi, C. N., Chandrasekharan, A., Sundararaman, V., and Alex, Z. C. (2015). Neonatal brain mri segmentation: A review. *Computers in biology and medicine*, 64:163–178.

[11] Dolz, J., Desrosiers, C., and Ayed, I. B. (2018). 3d fully convolutional networks for subcortical segmentation in mri: A large-scale study. *NeuroImage*, 170:456–470.

[12] Eran, A., Ozturk, A., Aygun, N., and Izbudak, I. (2010). Medulloblastoma: atypical ct and mri findings in children. *Pediatric radiology*, 40(7):1254–1262.

[13] Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181.

[14] Fletcher-Heath, L. M., Hall, L. O., Goldgof, D. B., and Murtagh, F. R. (2001). Automatic segmentation of non-enhancing brain tumors in magnetic resonance images. *Artificial intelligence in medicine*, 21(1-3):43–63.

[15] Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., and Garcia-Rodriguez, J. (2017). A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*.

[16] Gering, D. T., Nabavi, A., Kikinis, R., Hata, N., O'Donnell, L. J., Grimson, W. E. L., Jolesz, F. A., Black, P. M., and Wells III, W. M. (2001). An integrated visualization system for surgical planning and guidance using image fusion and an open mr. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 13(6):967–975.

[17] Goo, H. W. and Ra, Y.-S. (2017). Advanced mri for pediatric brain tumors with emphasis on clinical benefits. *Korean journal of radiology*, 18(1):194–207.

[18] Gorgolewski, K., Burns, C. D., Madison, C., Clark, D., Halchenko, Y. O., Waskom, M. L., and Ghosh, S. S. (2011). Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python. *Frontiers in neuroinformatics*, 5:13.

[19] Gousias, I. S., Edwards, A. D., Rutherford, M. A., Counsell, S. J., Hajnal, J. V., Rueckert, D., and Hammers, A. (2012). Magnetic resonance imaging of the newborn brain: manual segmentation of labelled atlases in term-born and preterm infants. *Neuroimage*, 62(3):1499–1509.

[20] Gousias, I. S., Hammers, A., Counsell, S. J., Srinivasan, L., Rutherford, M. A., Heckemann, R. A., Hajnal, J. V., Rueckert, D., and Edwards, A. D. (2013). Magnetic resonance imaging of the newborn brain: automatic segmentation of brain images into 50 anatomical regions. *PloS one*, 8(4).

[21] Guinard, S. and Landrieu, L. (2017). Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds. In *ISPRS Workshop 2017*.

[22] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

[23] Huo, Y., Xu, Z., Xiong, Y., Aboud, K., Parvathaneni, P., Bao, S., Bermudez, C., Resnick, S. M., Cutting, L. E., and Landman, B. A. (2019). 3d whole brain segmentation using spatially localized atlas network tiles. *NeuroImage*, 194:105–119.

[24] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

[25] Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579.

[26] Jenkinson, M., Pechaud, M., Smith, S., et al. (2005). Bet2: Mr-based estimation of brain, skull and scalp surfaces. In *Eleventh annual meeting of the organization for human brain mapping*, volume 17, page 167. Toronto.

[27] Kamnitsas, K., Ledig, C., Newcombe, V. F., Simpson, J. P., Kane, A. D., Menon, D. K., Rueckert, D., and Glocker, B. (2017). Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical image analysis*, 36:61–78.

[28] Khalili, N., Lessmann, N., Turk, E., Claessens, N., de Heus, R., Kolk, T., Viergever, M. A., Benders, M. J., and Išgum, I. (2019). Automatic brain tissue segmentation in fetal mri using convolutional neural networks. *Magnetic resonance imaging*, 64:77–89.

[29] Lanczos, C. (1950). *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA.

[30] Landrieu, L. and Simonovsky, M. (2018). Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4558–4567.

[31] Levinshtein, A., Stere, A., Kutulakos, K. N., Fleet, D. J., Dickinson, S. J., and Siddiqi, K. (2009). Turbopixels: Fast superpixels using geometric flows. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2290–2297.

[32] Li, W., Wang, G., Fidon, L., Ourselin, S., Cardoso, M. J., and Vercauteren, T. (2017). On the compactness, efficiency, and representation of 3d convolutional networks: brain parcellation as a pretext task. In *International Conference on Information Processing in Medical Imaging*, pages 348–360. Springer.

[33] Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

[34] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

[35] Menze, B. H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., Burren, Y., Porz, N., Slotboom, J., Wiest, R., et al. (2014). The multimodal brain tumor image segmentation benchmark (brats). *IEEE transactions on medical imaging*, 34(10):1993–2024.

[36] MICCAI-2012 (2012). Promise12: Prostate mr image segmentation 2012. `https://promise12.grand-challenge.org/`.

[37] Milletari, F., Navab, N., and Ahmadi, S.-A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571. IEEE.

[38] Moeskops, P., Viergever, M. A., Mendrik, A. M., De Vries, L. S., Benders, M. J., and Išgum, I. (2016). Automatic segmentation of mr brain images with a convolutional neural network. *IEEE transactions on medical imaging*, 35(5):1252–1261.

[39] Morel, B., Antoni, G., Teglas, J., Bloch, I., and Adamsbaum, C. (2016). Neonatal brain mri: how reliable is the radiologist's eye? *Neuroradiology*, 58(2):189–193.

[40] Pham, C.-H., Tor-Díez, C., Meunier, H., Bednarek, N., Fablet, R., Passat, N., and Rousseau, F. (2019). Simultaneous super-resolution and segmentation using a generative adversarial network: Application to neonatal brain mri. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 991–994. IEEE.

[41] Poussaint, T. Y., Kocak, M., Vajapeyam, S., Packer, R. I., Robertson, R. L., Geyer, R., Haas-Kogan, D., Pollack, I. F., Vezina, G., Zimmerman, R., et al. (2011). Mri as a central component of clinical trials analysis in brainstem glioma: a report from the pediatric brain tumor consortium (pbtc). *Neuro-oncology*, 13(4):417–427.

[42] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660.

[43] Robert, C. (2014). Machine learning, a probabilistic perspective.

[44] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

[45] Rosati, S., Toselli, B., Fato, M. M., Tortora, D., Severino, M., Rossi, A., and Balestra, G. (2019). Pediatric brain tissue segmentation from mri using clustering: a preliminary study. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 6557–6560. IEEE.

[46] Roth, H. R., Lu, L., Seff, A., Cherry, K. M., Hoffman, J., Wang, S., Liu, J., Turkbey, E., and Summers, R. M. (2014). A new 2.5 d representation for lymph node detection using random sets of deep convolutional neural network observations. In *International*

*conference on medical image computing and computer-assisted intervention*, pages 520–527. Springer.

[47] Shan, Z. Y., Ji, Q., Gajjar, A., and Reddick, W. E. (2005). A knowledge-guided active contour method of segmentation of cerebella on mr images of pediatric patients with medulloblastoma. *Journal of Magnetic Resonance Imaging: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 21(1):1–11.

[48] Sharma, S. (2017). Activation functions in neural networks. *towards data science*, 6.

[49] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *Departmental Papers (CIS)*, page 107.

[50] Stacke, K. (2016). Automatic brain segmentation into substructures using quantitative mri.

[51] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

[52] Tor-Díez, C., Passat, N., Bloch, I., Faisan, S., Bednarek, N., and Rousseau, F. (2018). An iterative multi-atlas patch-based approach for cortex segmentation from neonatal mri. *Computerized Medical Imaging and Graphics*, 70:73–82.

[53] Tortori-Donati, P., Fondelli, M., Rossi, A., Cama, A., Caputo, L., Andreussi, L., and Garre, M. (1996). Medulloblastoma in children: Ct and mri findings. *Neuroradiology*, 38(4):352–359.

[54] Tustison, N. J., Avants, B. B., Cook, P. A., Zheng, Y., Egan, A., Yushkevich, P. A., and Gee, J. C. (2010). N4itk: improved n3 bias correction. *IEEE transactions on medical imaging*, 29(6):1310–1320.

[55] Van Leemput, K., Maes, F., Vandermeulen, D., and Suetens, P. (1999). Automated model-based tissue classification of mr images of the brain. *IEEE transactions on medical imaging*, 18(10):897–908.

[56] Vincent, L. and Soille, P. (1991). Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(06):583–598.

[57] Wang, L., Nie, D., Li, G., Puybareau, É., Dolz, J., Zhang, Q., Wang, F., Xia, J., Wu, Z., Chen, J.-W., et al. (2019). Benchmark on automatic six-month-old infant brain segmentation algorithms: the iseg-2017 challenge. *IEEE transactions on medical imaging*, 38(9):2219–2230.

[58] Wang, S., Zhang, Y., Liu, G., Phillips, P., and Yuan, T.-F. (2016). Detection of alzheimer's disease by three-dimensional displacement field estimation in structural magnetic resonance imaging. *Journal of Alzheimer's Disease*, 50(1):233–248.

[59] Weisenfeld, N. I. and Warfield, S. K. (2009). Automatic segmentation of newborn brain mri. *Neuroimage*, 47(2):564–572.

[60] Xue, H., Srinivasan, L., Jiang, S., Rutherford, M., Edwards, A. D., Rueckert, D., and Hajnal, J. V. (2007). Automatic segmentation and reconstruction of the cortex from neonatal mri. *Neuroimage*, 38(3):461–477.

[61] Yeom, K. W., Mobley, B. C., Lober, R. M., Andre, J. B., Partap, S., Vogel, H., and Barnes, P. D. (2013). Distinctive mri features of pediatric medulloblastoma subtypes. *American Journal of Roentgenology*, 200(4):895–903.

[62] Yu, C.-P., Ruppert, G., Collins, R., Nguyen, D., Falcao, A., and Liu, Y. (2014). 3d blob based brain tumor detection and segmentation in mr images. In *Biomedical Imaging (ISBI), 2014 IEEE 11th International Symposium on*, pages 1192–1197. IEEE.

[63] Yu, J. and Blaschko, M. B. (2018). The lovász hinge: A novel convex surrogate for submodular losses. *IEEE transactions on pattern analysis and machine intelligence*.

[64] Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., and Sun, M. (2018). Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*.

[65] Zoghbi, J. M., Mamede, M. H., and Jackowski, M. P. (2010). Computer-assisted segmentation of brain tumor lesions from multi-sequence magnetic resonance imaging using the mumford-shah model. In *Image and Vision Computing New Zealand (IVCNZ), 2010 25th International Conference of*, pages 1–6. IEEE.