

VOCAL TRACT SEGMENTATION

GROUP 1

Locatelli Lorenzo, Masotti Aurora,
Naclerio Andrea, Poli Giuseppe and Rosato Elena

A.Y. 2023/2024

INDEX

1. CLINICAL CONTEST AND AIM OF THE PROJECT	3
2. PREPROCESSING: GAUSSIAN NOISE REMOVAL.....	3
2.1. TEST 1.....	4
2.1.1. GAUSSIAN FILTER: GAUSSIANBLUR.....	4
2.1.2. HIGH-PASS FILTER	5
2.2. TEST 2.....	6
2.2.1. GAUSSIAN FILTER: RUDIN, OSHER AND FATEMI ALGORITHM	6
2.2.2. ENHANCEMENT OF HIGH FREQUENCIES	7
2.3. TEST 3.....	8
2.3.1. GAUSSIAN FILTER: FASTNLMEANSDENOISING.....	8
2.4. PREPROCESSING FINAL CONSIDERATIONS	8
2.5. SPLITTING TRAIN, VALIDATION AND TEST SET	9
3. MODEL ARCHITECUTURE.....	10
3.1. RUTHVEN U-NET	10
3.1.1 WEIGHTED LOSS FUNCTION	11
3.1.2. DATASET AUGMENTATION	12
3.2. U-NET WITH RESIDUAL BLOCKS.....	13
3.2.1 HYPERPARAMETER TUNING	14
4. POST PROCESSING	15
4.1. ARGMAX	15
4.2. MEDIAN FILTER	15
4.3. MODEL EVALUATION	16
5. CONCLUSION	17
LIST OF FIGURES	18
LIST OF TABLES.....	18

1. CLINICAL CONTEST AND AIM OF THE PROJECT

Studies about speech are performed using Real Time Magnetic Resonances (rtMR) which is a non-invasive technique, based on the fast acquisition of MR images that allows clinicians to perform analysis regarding the shape, size, motion and position of vocal tract articulators. To extract these type of information, segmentation methods have been investigated in the literature. The one proposed is based on the use of a fully convolutional network with an architecture similar to an original U-net, which can perform the segmentation of six regions: Upper lip, Hard palate, Soft palate, Tongue, Lower lip and Head. Between them, tongue and soft palate have a central role: their motion and anatomical features, indeed, are representative of the reasons why some people are more predisposed to speech problems. In particular, the soft palate enclosure with the pharyngeal wall is required to produce most of the sounds while speaking. Therefore, the computation of the distance between these two structures, after performing vocal tract segmentation through the U-net, is of clinical interest.

The aim of the project was to implement a Ruthven U-net architecture in order to create a model able to perform the vocal tract segmentation with the highest accuracy, especially for these last regions cited.

2. PREPROCESSING: GAUSSIAN NOISE REMOVAL

The Gaussian noise is a statistical term that describes a certain type of random variation. In the context of images, it refers to the presence of random variations in pixel intensity values that follow a Gaussian distribution. In simpler terms, the intensity values of pixels are perturbed randomly, with most of the perturbations centred around the original intensity value and becoming less likely as the perturbations deviate from the centre. This distribution of noise results in a smooth and continuous spread of intensity variations. It can arise from various factors such as physical variables like temperature, brightness and intrinsic sensor characteristics, or can be due to, the quantization process during digitization of images, errors in electronic circuits or due to patient characteristics like movements during image acquisition and presence of complex anatomical structures.

In order to remove this kind of noise from the images in the dataset, 3 tests have been performed using different kind of filters:

- TEST 1: Gaussian filter (*GaussianBlur*) with a high pass filter;
- TEST 2: Gaussian filter (*Rudin, Osher and Fatemi algorithm*) with an enhancement of high frequencies;
- TEST 3: Gaussian filter only (*fastNIMeansDenoising*).

2.1. TEST 1

2.1.1. GAUSSIAN FILTER: GAUSSIANKBLUR

Starting with a random image of the dataset, as a first test a Gaussian filter is applied using the `cv2.GaussianBlur()` function of the OpenCV library. This function blurs the image using a Gaussian filter and, in particular, it convolves the source image with the specified Gaussian kernel.



Figure 1: Original image

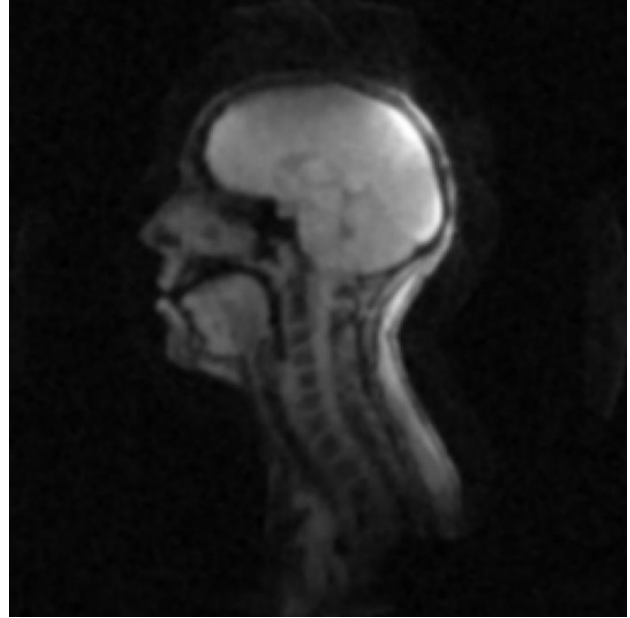


Figure 2: Gaussian noise removed

The original image (*Figure 1*) on the left is characterised by visible noise mainly in the background, while the filtered one (*Figure 2*) has a more uniform background but with more blurred anatomical parts, making it difficult to distinguish between the various classes.

2.1.2. HIGH-PASS FILTER

By applying an additional high-pass filter using OpenCV's `cv2.filter2D()` function, the previously filtered image can be improved. This function in fact creates a convolution between the image at pixel level and the given kernel (a 2D matrix).

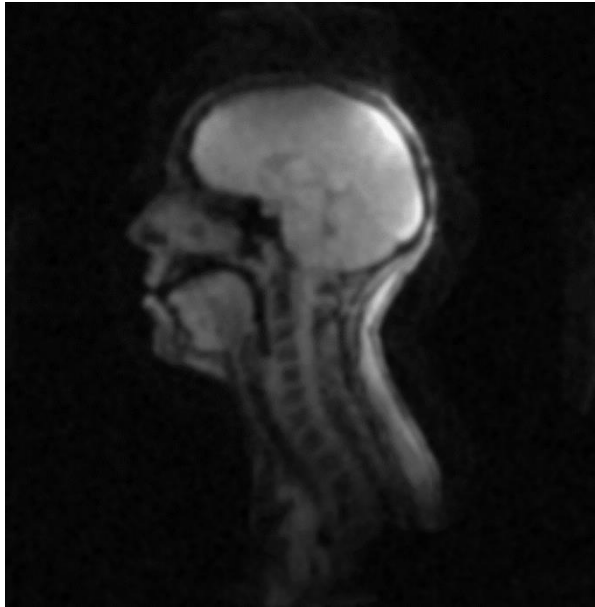


Figure 3: Gaussian noise removed

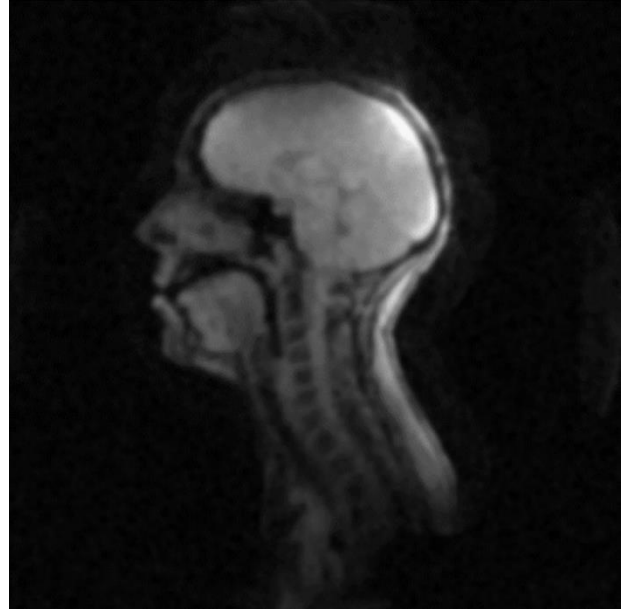


Figure 4: Gaussian noise removed + high-pass filter

The new filtered image (*Figure 4*) has in this way an improvement in the overall sharpness, even if, by looking closer to the mouth region, it's possible to notice that the improvement is not so significant in the areas of interest for segmentation.

2.2. TEST 2

2.2.1. GAUSSIAN FILTER: RUDIN, OSHER AND FATEMI ALGORITHM

As a second test, an algorithm developed by Rudin, Osher and Fatemi, found through an online research (on the GitHub platform), is used. It consists of a series of operations carried out with the aim of performing total variation denoising on a grayscale image.

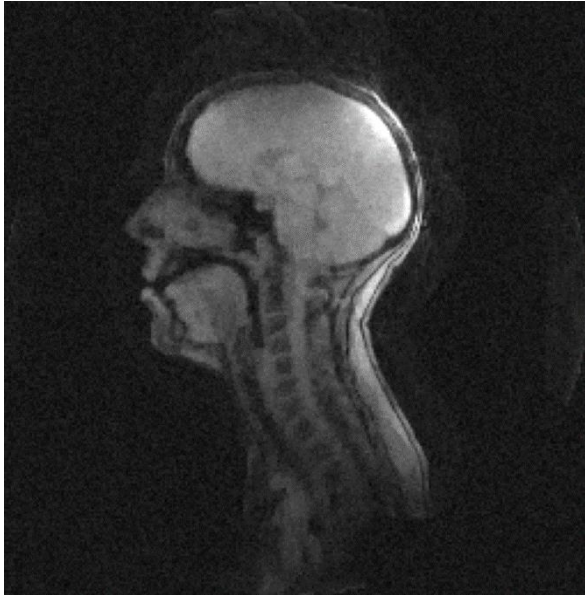


Figure 5: Original image



Figure 6: Gaussian noise removed

Compared to the original image (*Figure 5*), after the filter application, a greater uniformity of the larger areas and a better distinction of the different anatomical parts can be observed (*Figure 6*). These distinctions are even more visible by looking at the area around the mouth and nose.

2.2.2. ENHANCEMENT OF HIGH FREQUENCIES

Following this process, a further high-frequency enhancement operation is applied in order to better refine certain details, such as the contours of the different areas of the vocal tract that are of interest for this analysis.

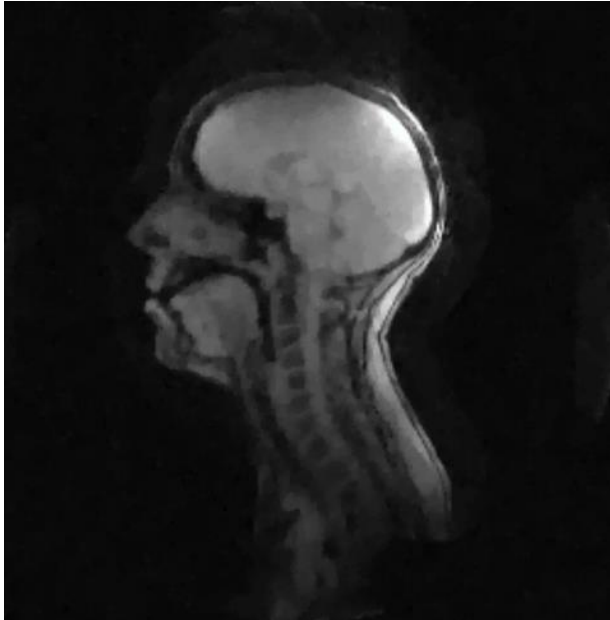


Figure 7: Gaussian noise removed

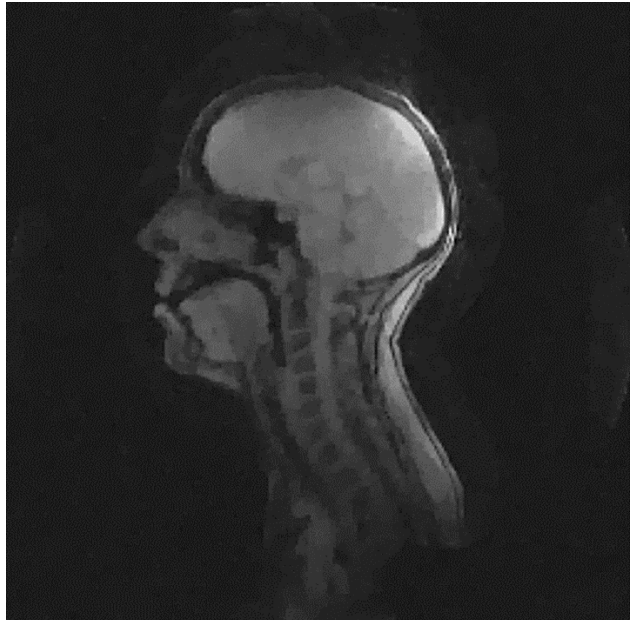


Figure 8: Gaussian noise removed + enhancement of high frequencies

As it can be seen in *Figure 8* the images are better in term of contour definition with respect to the ones in output of 2.1. TEST 1 but still have some noise regions, like the ones around the head, and a flattening of the grey scale.

2.3. TEST 3

2.3.1. GAUSSIAN FILTER: FASTNLMEANSDENOISING

As a final approach, it has been developed a code that directly acts on the Gaussian noise without relying on a pre-developed algorithm or even a high-pass filter. To do this, the function `cv2.fastNlMeansDenoising()`, from the OpenCV library, was used to perform a non-local denoising (by using a *Non-local Means Denoising algorithm*), which means that the noise removal process takes into account the global characteristics of the image instead of considering only a small local portion. The method is based on replacing the colour of a pixel with an average of the colours of similar pixels that have no reason to be close at all. It is therefore licit to scan a vast portion of the image in search of all the pixels that really resemble the pixel one wants to denoise. This function is optimal for the case in analysis because it is expected to be applied to grayscale images like the ones in the dataset.



Figure 9: Original image



Figure 10: Gaussian noise removed

After its application, good noise reduction can be seen in the output image (*Figure 10*), characterised by a uniform background, clearly distinguished from the patient's head, and by defined contours among the various anatomical classes.

2.4. PREPROCESSING FINAL CONSIDERATIONS

As a conclusion to this preprocessing analysis, it can be said that the three tests carried out in general to good results with particular emphasis on 2.2. TEST 2 and 2.3. TEST 3. For the next analysis and realisation of the neural architecture, it has been decided to use the filtered images coming out of 2.3. TEST 3 because the one characterised by images with better sharpness and reduction of background noise.

2.5. SPLITTING TRAIN, VALIDATION AND TEST SET

In this phase indexes of the dataset have been plotted, and it has been discovered that the data represent MRI from four different subjects. Being images from the same subject highly correlated, it has been decided to use two of them for the train, one for the validation and the remaining one for the test instead of using a combination of them. The selection of the first two subjects for the training was based on the fact that they had more images, 280 and 240 respectively with respect to 150 of the other two subjects, so they could bring more variability to the model. Moreover, the choice of testing and validate the model on different subjects from the ones used for the training reflects the aim to assess the ability of the model to generalize on data that it had never seen before.

3. MODEL ARCHITECTURE

3.1. RUTHVEN U-NET

The base of the architecture developed for this analysis is the so-called, Ruthven U-net (*Figure 11*), taken from the literature and it is constituted by two main paths: encoder and decoder.

Specifically, the encoder path extracts high-level features while going deeper. The standard approach consists of reducing the spatial dimension, attaching a MaxPooling layer to the base-block, while increasing the number of feature maps from 64 to 1024. In this way, the so-called latent representation space is reached (where high-level features have been extracted) but the spatial resolution is lost. So, the information regarding the location in the initial images is missing.

At this point the decoder path is applied. Conversely from the previous one, the aim of this path is to restore the initial image spatial dimension. A transpose convolutional layer has been adopted in combination with the standard block to increase gradually the spatial dimension.

To help the reconstruction phase, at each step a skip connection is inserted to pass some information that has been extracted with the encoder where the spatial information is partially preserved.

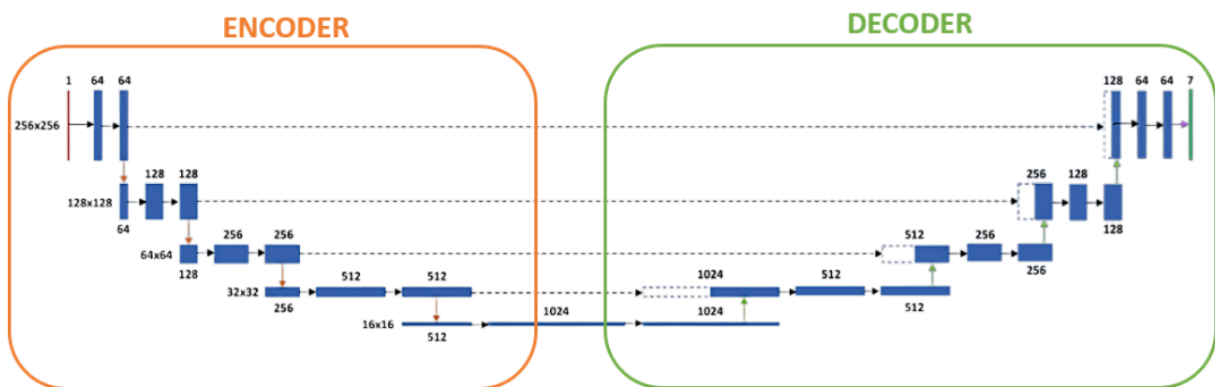


Figure 11: Literature U-Net architecture

The network is based on a common block structure composed by 3 layers (Figure 12): a 2D convolutional, batch normalization and an activation layer (with ReLu as activation function).

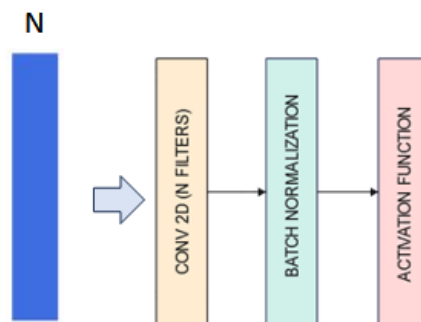


Figure 12: Common block structure

3.1.1 WEIGHTED LOSS FUNCTION

The final output of the model is a volume with the same spatial dimension as the input images, but with 7 different channels. In fact, by looking at *Figure 13*, binary images (after argmax) can be recognized: only regions of a particular anatomical part are identified with values equal to 1, while the rest of the image is characterized by the values 0.

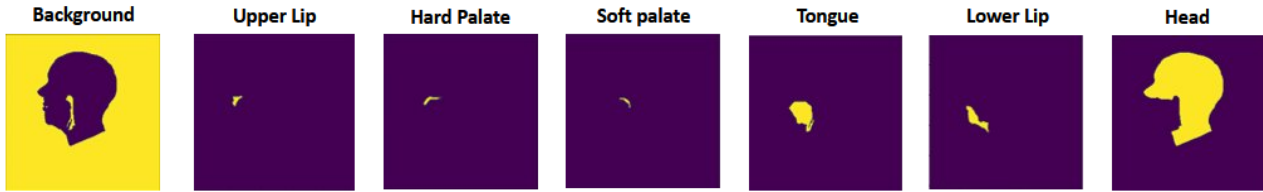


Figure 13: Output channels content

Since the area covered by different anatomical parts is quite different, the model might present a good performance, in terms of mean DICE, even though the smaller areas are not correctly classified. This could be problematic since, as mentioned before, those are the regions of clinical interest, such as the 'Soft palate'.

To avoid the problem that the number of pixels in those regions is lower, it's been increased the importance of the error corresponding to a misclassification in those anatomical parts, multiplying each by a weight: w_k . To perform this, two different algorithms have been applied:

- The first method adopted has been taken from the literature and was based on the following formula: $w_k = \sum_k N_k / N_k$ where N_k is the number of pixels associated to the k^{th} class. In this case the weight of the small anatomical part is quite high, reaching a value of 200, while those for larger regions are close to 1.
- The second algorithm is based on the first one through the following formula: $w_k = 1 - N_k / \sum_k N_k$. In this way the weights are bounded between 0 and 1 to avoid losing precision on small values due to how MAC is managed with float representation.

As we can see from the table represented in *Table 1*, the model trained with the second algorithms achieved better performance with a mean DICE of 0.83 in the validation with respect to 0.79 obtained using the first one.

Then, the customized loss function has been adopted in the architecture described above.

MEAN DICE	Literature weights	Customized weights
Val	0.793	0.839
Test	0.791	0.829

Table 1: U-Net plus weighted loss function performance

3.1.2. DATASET AUGMENTATION

Successively, efforts have been made to enhance the model's performance while keeping the architecture previously described, specifically the Ruthven U-net with customized weights. Additionally, the training set has been expanded through augmentation to improve the model generalization capabilities. To preserve the image content, 3 transformations have been applied, limiting their quantities: maximum rotation of 20 degrees, translation of 0.0625 cm and scaling of 0.50.

These augmented images were added to the initial training doubling the number of images, from 512 to 1024. To see the efficacy of the augmentation, the performance of our customized Ruthven U-net, trained with only the initial training dataset and the augmented one have been compared. Validation and test set remained equal.

Looking at the performance reported in *Table 2*, we can see that the model trained on the augmented set achieved a higher Mean DICE, 0.88, respect 0.86.

MEAN DICE	CUSTOMIZED WEIGHTS + AUGMENTATION	CUSTOMIZED WEIGHTS + NO AUGMENTATION
Val	0.880	0.864
Test	0.881	0.862

Table 2: Model performance with augmented dataset

3.2. U-NET WITH RESIDUAL BLOCKS

To enhance the performance of the U-net neural network, residual blocks were incorporated into our architecture (*Figure 14*). This enhancement involves a simple but impactful modification to the traditional U-net structure, where the basic block is transformed into a residual block by concatenating the input of the first block with the output of the second one in the same layer. This operation is repeated for each layer, both in the decoder and the encoder part.

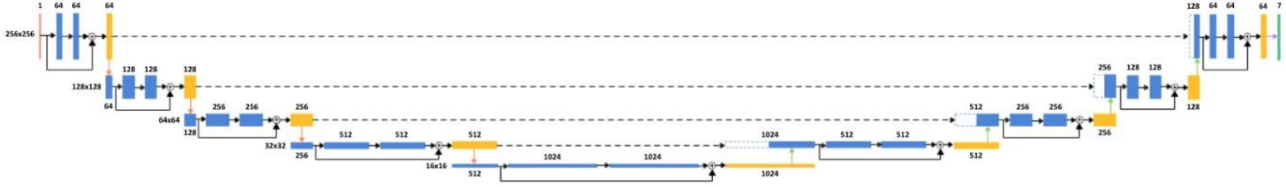


Figure 14: U-net with residual blocks

The incorporation of residual blocks allows the network to capture more intricate features, proving particularly advantageous when performing segmentation tasks that involve subtle details. The improved architecture was evaluated using all the different parameters already explored with the previous U-net: this included trying different weights for the loss function, employing the three Gaussian filters mentioned above in the preprocessing phase and evaluating the impact of data augmentation. The best results were achieved under the same conditions that had proven most effective for the classic U-net. Overall, by comparing the segmentations obtained with both architectures, we can see a clear improvement given by the introduction of residual blocks, especially in the segmentation of smaller and more challenging classes, such as the soft palate (*Figure 15*).

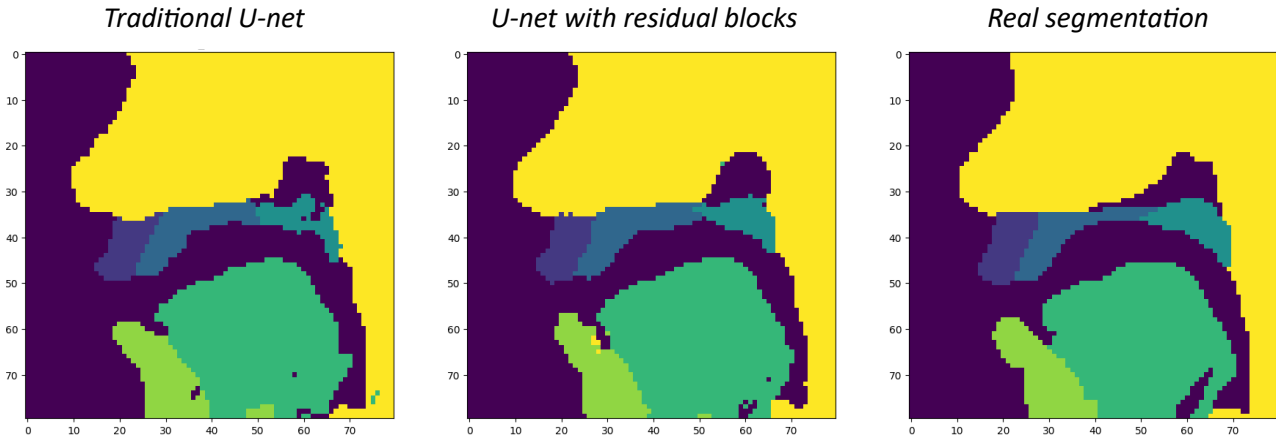


Figure 15: Comparison of the segmentation performed with traditional U-net and U-net with residual blocks

This improvement is reflected also in the main metrics that has been evaluated in the study: the mean DICE, the precision, and the recall. As can be seen in the table below (*Table 3*), the implementation of residual blocks showed an improvement in each of these metrics.

	Traditional U-net	U-net with residual blocks
Precision	0,911	0,915
Recall	0,883	0,887
Mean DICE	0,896	0,899

Table 3: Performance of the U-net and U-net with residual blocks

3.2.1 HYPERPARAMETER TUNING

The model obtained at this point was trained with a learning rate of 0.0001, a batch size of 15 and starting with 64 filters for the first layer. To enhance performance, various models were experimented with by adjusting hyperparameters, and the outcomes were evaluated based on the mean DICE metric. Conducting a systematic grid search was impractical due to the extensive training time required for each model, so a manual hyperparameter tuning was performed. Regarding the learning rate and the number of filters, no improvement was recorded after changing them. For example, it is possible to see in the table below (*Table 4*) that by reducing the starting filters to 32 the performance gets worse.

	FILTERS = 32	FILTERS = 64
Mean DICE	0,876	0,899

Table 4: Mean DICE with different number of filters

The batch size was the only hyperparameter that showed improvement upon modification. Specifically, setting the batch size to 10 resulted in a marginal increase in mean DICE (*Table 5*). This improvement could be attributed to the fact that a smaller batch size introduces more stochasticity and noise into the training process creating a regularizing effect and promoting better generalization. However, this positive effect comes at the expense of a further increase in training time.

	BATCH SIZE = 10	BATCH SIZE = 15
Mean DICE	0,903	0,899

Table 5: Mean DICE with different batch size

4. POST PROCESSING

4.1. ARGMAX

Being the final layer of the network a Softmax, the outputs consisted in seven probability maps, one for each class. So, in order to binarize the final images, argmax has been applied. This function consists in the assignment of a class for which a specific pixel has the maximum probability to belong to, and led to an increasing in the mean DICE, as expected (see *Figure 16*).

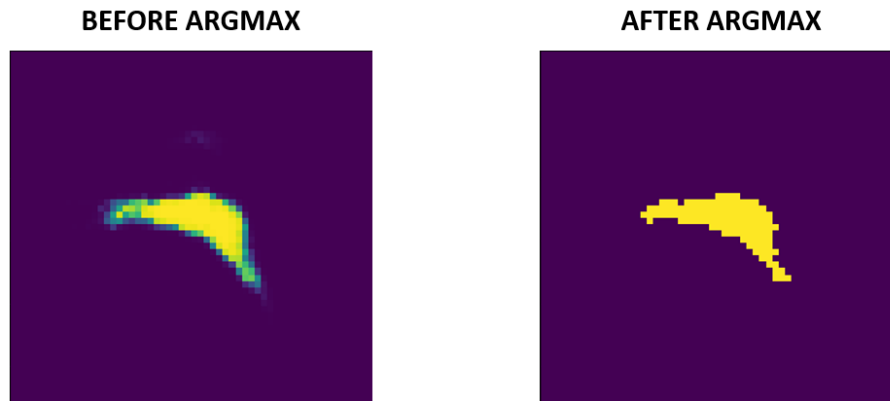


Figure 16: 'Soft palate' before and after argmax

4.2. MEDIAN FILTER

A median filter is used to smooth out irregularities of the images by assigning to a pixel the median value of the pixels around it which are considered by moving across the images a kernel with a specific size. We tried different values for it and the best performances were given by a 2x2 filter. This operation has been performed trying to remove those pixels belonging to a class different with respect to the surrounding one. However, applying the filter to all classes led to a reduction in performance in terms of mean DICE (*Figure 17*), except for class three, representing the 'Soft palate'. 'Soft palate' was the most critical one both from a clinical point of view, as already mentioned, but also in terms of practical reasons in training the model: indeed, this class represents the most involved region during the speech, so in rtMRI was the one with the most variability. Due to our results, we decided to filter only this class, which slightly improved the DICE for the 'Soft palate' to 0.90644.

















	Background	Upper Lip	Hard Palate	Soft palate	Tongue	Lower Lip	Head	MEAN DICE
DICE AFTER ARGMAX	0.9911	0.9031	0.8476	0.7910	0.9320	0.8858	0.9669	0.9025
MEDIAN FILTER TO ALL CLASSES	0.9913	↓0.8473	↓0.8164	↑0.8184	↓0.9279	↓0.8783	0.9657	0.8922↓

Figure 17: Median filter results when applied to all classes

4.3. MODEL EVALUATION

To evaluate the generalization capability of the network, we performed a model validation considering different combination of subjects for the training, validation and test set. In particular, we decided to maintain subject 1 in the training test since his larger number of images could introduce in the network higher variability.

Model performances are represented in the figure below (Figure 18) and are quite comparable. The second patient, when used for testing the model, gave worst results. This behaviour could be caused by: anatomical similarities between different subjects which could lead to a better segmentation when using a specific patient for training and another one for testing, or by the fact that the 2nd patient dataset was larger than the 3rd and the 4th, thus, since we have been dealing with a poor number of images, reducing the variability in input to the model.

TRAINING	VALIDATION	TEST	VALIDATION DICE	TEST DICE PRE ARGMAX	TEST DICE POST ARGMAX	TEST DICE MEDIAN FILTER
 280  240	 150	 150	0,8871	0,8916	0,9025	0,9064
 			0,8831	0,9129	0,9180	0,9186
 			0,9108	0,8527	0,8567	0,8579
 			0,9028	0,8473	0,8493	0,8499

— Patient 1
— Patient 2
— Patient 3
— Patient 4

Figure 18: Model evaluation

5. CONCLUSION

The evaluation of the best model has been performed on the mean DICE, however, to obtain other kinds of information, two additional metrics, precision and recall, have been computed.

Precision is defined as the portion of correct positive predictions of all cases classified as positive, so it is telling how often the model is right in predicting something as positive.

$$precision = \frac{TP}{TP + FP}$$

A low precision in our case means having a high number of FP and so is reflected in over-segmentation.

Recall, on the contrary, represents the percentage of a certain class correctly identified, by considering in the formula the FN instead of the FP.

$$recall = \frac{TP}{TP + FN}$$

Having a low recall means performing a misclassification error and so an under-segmentation.

Ideally, these two metrics should be as much high as possible but, when one increases the other one decreases, so a trade-off is needed.

Our final model shows a slightly higher precision (*Table 6*), which is reflected in an inclination to over-segment images. Over-segmentation could be a problem if, for example, the aim is to identify a speaking problem by looking at the gap between the ‘soft palate’ and the pharyngeal wall when the subject is saying specific words. In this situation, indeed, this gap might be invisible, even if the subject’s soft palate doesn’t enter in contact with the pharyngeal wall, leading to the missing of a pathological case which, from a clinical point of view, is the worst situation. However, in this situation, being the precision and recall very close to each other, it has been concluded that the final model is a good candidate to be used as a diagnostic tool for anatomical regions segmentation to support clinical decisions.

MEAN DICE	PRECISION	RECALL
0,9025	0,9056	0,901

Table 6: Final model mean DICE, precision and recall

LIST OF FIGURES

Figure 1: Original image	4
Figure 2: Gaussian noise removed	4
Figure 3: Gaussian noise removed	5
Figure 4: Gaussian noise removed + high-pass filter	5
Figure 5: Original image	6
Figure 6: Gaussian noise removed	6
Figure 7: Gaussian noise removed	7
Figure 8: Gaussian noise removed + enhancement of high frequencies	7
Figure 9: Original image	8
Figure 10: Gaussian noise removed	8
Figure 11: Literature U-Net architecture	10
Figure 12: Common block structure	10
Figure 13: Output channels content	11
Figure 14: U-net with residual blocks	13
Figure 15: Comparison of the segmentation performed with traditional U-net and U-net with residual blocks	13
Figure 16: 'Soft palate' before and after argmax	15
Figure 17: Median filter results when applied to all classes	15
Figure 18: Model evaluation	16

LIST OF TABLES

Table 1: U-Net plus weighted loss function performance	11
Table 2: Model performance with augmented dataset	12
Table 3: Performance of the U-net and U-net with residual blocks	13
Table 4: Mean DICE with different number of filters	14
Table 5: Mean DICE with different batch size	14
Table 6: Final model mean DICE, precision and recall	17