Slovak University of Technology
Faculty of Informatics and Information Technologies

FIIT-5208-46174

Bc. Martin Tamajka

Segmentation of anatomical organs in medical data

Master's Thesis

Study programme: Information Systems
Field of study: 9.2.6 Information Systems,
Workplace: Institute of Computer Engineering and Applied Informatics, FIIT STU Bratislava
Supervisor: Ing. Vanda Benešová, PhD.

May 2016

ZADANIE VSEOBECNE

NAVRH ZADANIA

Čestné prehlásanie

Vyhlasujem, že záverečnú prácu som vypracoval samostatne s použitím uvedenej literatúry a znalostí získaných počas štúdia.

Bratislava, 13.5.2016

.....

Vlastnoručný podpis

Anotácia

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
Študijný program: Informačné systémy

Autor: Bc. Martin Tamajka
Diplomová práca: Segmentácia anatomických orgánov v medicínskych dátach
Vedúci práce: Ing. Vanda Benešová, PhD.

máj 2016

Segmentácia medicínskych dát je dôležitou súčasťou medicínskej praxe. Špeciálne pokiaľ sa jedná o prácu rádiológov, segmentácia značne zjednodušuje ich každodenné úlohy a zefektívňuje využívanie ich času, čo je prínosné najmä z dôvodu, že vo väčšine prípadov majú rádiológovia iba určité množstvo času, ktorý môžu venovať vyšetreniu jedného pacienta. Počítačová podpora diagnostiky je taktiež mocným nástrojom na elimináciu možného zlyhania ľudského faktoru.

V tejto práci navrhujeme nový prístup k segmentovaniu ľudských orgánov. Primárne sa pritom zameriavame na segmentáciu ľudského mozgu z MR dát. Naša metóda je založená na presegmentovaní 3D dát do tzv. supervoxelov za použitia algoritmu SLIC. Jednotlivé supervoxely sú opísané množinou príznakov založenou na distribúcií intenzít obsiahnutých voxelov a na pozícií v rámci samotného mozgu. Supervoxely sú klasifikované neurónovými sieťami, ktoré trénujeme, aby rozhodli, či supervoxely patria k danému orgánu či tkanivu. Pre ďalšie spresnenie našej metódy využívame informáciu o tvare a vnútornej štruktúre orgánu. V konečnom dôsledku zavádzame 6-krokovú segmentačnú metódu založenú na použití klasifikácie. Našu metódu porovnávame s ostatnými metódami, ktoré momentálne tvoria vrchol poznania v tejto oblasti.

Okrem globálneho cieľa tejto práce sa zameriavame na aplikovanie inžinierskych zručností a najlepších praktík, aby implementované riešenie bolo ľahko rozšíriteľné a udržovateľné v budúcnosti.

Annotation

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES
Degree Course: Information Systems

Author: Bc. Martin Tamajka
Master's Thesis: Segmentation of anatomical organs in medical data
Supervisor: Ing. Vanda Benešová, PhD.

2016, May

Medical image segmentation is an important part of medical practice. Primarily as far as radiologists are concerned it simplifies their everyday tasks and allows them to use their time more effective, because in most cases radiologists only have a certain amount of time they can spend examining patient's data. Computer aided diagnosis is also a powerful instrument to eliminate possible human failure.

In this work, we propose a novel approach to human organs segmentation. We primarily concentrate on segmentation of human brain from MR volume. Our method is based on oversegmenting 3D volume to supervoxels using SLIC algorithm. Individual supervoxels are described by features based on intensity distribution of contained voxels and on position within the brain. Supervoxels are classified by neural networks which are trained to classify supervoxels to individual tissues. In order to give our method additional precision, we use information about the shape and inner structure of the organ. In general we propose a 6-step segmentation method based on classification. We compare our method with other state-of-the-art methods.

Apart from the global focus of this thesis, our goal is to apply engineering skills and best practices to implement proposed method and necessary tools in way that they can be easily extended and maintained in the future.

Poďakovanie

V prvom rade ďakujem doktorke Wande Benešovej, ktorá mi už počas bakalárskeho štúdia pomohla spoznať čaro počítačového videnia. Zo spoločných konzultácií som si pravidelne odnášal množstvo nových nápadov, ktoré mi pomohli úspešne dokončiť túto prácu.

Môže to vyznieť ako klišé, no určite by som sa nedostal až sem, keby som nebol obklopený skvelými ľuďmi, ktorí ma podporovali počas celého štúdia. Vydržali so mnou, keď som bol nevrlý a nezazlievali mi, že som opakovane vyprázdnil všetky nádoby, v ktorých sa ukrývala káva. Týmito ľuďmi sú moja rodina a priatelia. Nesmierne vám ďakujem.

Martin Tamajka

Contents

1 Introduction

In this part, we briefly describe domain and introduce the main parts of problem we face in this master's thesis, especially emphasizing main challenges and possible problems. We also guide reader through the structure of this document.

In this work, we propose a method for segmentation of specific organ from medical 3D data that requires minimal user interaction. Although our method is tested and evaluated on one specific organ – human brain, we believe it would be easy to use it to segment other organs as well.

In general, medical imaging grows in importance not only as far as treatment is concerned. There was also recorded importance of screening as a mean of lethal-diseases prevention. According to [1] it is more likely to cure a lethal disease if it is detected in early stage of its progression. Unfortunately, there are not enough experienced radiologists which would be able to perform screening of whole population. Authors in [2] claim that radiologist needs approximately 6.83 minutes for reading single MRI. If we wanted to examine every person in Slovakia once in two years we would need 107 radiologists working 8 hours a day every single day in a year only on screening. This is where medical imaging comes to aid. In some countries radiologists are already assisted by computers – computer either preprocesses medical data and makes them easier to read for radiologist or highlights some possible regions of interest. There is also a rule that a single examination should be performed by two radiologists. Such reading is called "Double reading". Although double reading improves sensitivity, it also increases requirements for radiological stuffing [3]. This fact would multiply required number of radiologists by two which would result in 214 radiologists needed in Slovakia.

Our work consists of three parts. In the first part, we provide theoretical background and description of proposed method. We describe how this method contributes to current state of medical images segmentation. We also mention its possible drawbacks, weak points and challenges.

In the second part, we implement proposed method using standard software tools such as programming languages and libraries. We put emphasis on robustness, effectiveness, validity and precision of our software implementation. The main challenges of software implementation are big amount of data (millions to hundred millions of voxels per dataset), noise and size and shape variance of organs [4].

In the last part, we evaluate our method through comparison with state-of-the-art brain segmentation techniques.

Document is divided into eight sections. In second chapter, we introduce user to the domain of radiology and medical imaging and explain why it is important to use

segmentation in medical data. Next, we analyse medical modalities, especially emphasising MR, which we chose to evaluate our method. In this chapter we also explain theoretical background and methods of computer vision that are used in or related to proposed method. In the last part, we describe some state-of-the-art segmentation techniques and methods used in medical domain.

Proposed method is thoroughly described in section three. Here we describe used datasets and all phases of the method.

During our research we tried various approaches to segmentation of medical organs. These approaches that are not yet used in the main method are described in section four.

Considered approaches that we have tried in earlier stages of this project are described in section five. These provided the basis for the finally proposed method.

The sixth section addresses implementation of proposed method. Usage of software engineering techniques, architecture and design of software solution are described in detail. In the first part we describe our first results reached during DP1.

In the seventh section, we evaluate results of our method and compare them with state-of-the-art medical segmentation methods. Proposed method is compared in terms of precision, effectiveness and performance.

The eight section, conclusion, gives an overview of proposed method and reached results. Limits of current approach and future work are also described here.

2 Analysis

2.1 Domain

For some kinds of treatment in medical practice it is necessary to get an image of patient's viscera. Whenever possible, such image is obtained in non-invasive way. Only in necessary cases chirurgic operation is performed only to get information about patient's viscera.

Medical data is obtained by different modalities. The most commonly used are CT, MRI, PET, RTG and ultrasound. Each of these modalities has its own attributes, which makes every one of them more suitable for specific kind of tasks. In the next part, we focus on MRI and CT.

CT stands for Computed tomography. CT uses X-rays to obtain information about patient's body, using fact that every tissue absorbs different amount of energy of rays. From our point of view CT provides more stable results when comparing multiple scans of one patient. This is caused by significantly lower time required to perform a CT scan in comparison with MRI. Similar to MRI, CT suffers from following artifacts: Partial volume, Streak, Motion, Beam hardening, Ring and Bloom. It is also disadvantageous that patient is exposed to radiation during CT scanning and soft tissues do not provide as much contrast as in MRI. In [5] authors state that their method allows to cope with high level of noise present in the CT scan. On the other hand, modern CT scanners provide high resolution, much shorter scan time, higher sensitivity for sub-arachnoids haemorrhage and higher sensitivity to calcification.

MRI stands for Magnetic resonance imaging. It is the most widely used technique in medical imaging [6]. It allows to focus on different tissues by adjusting MRI settings. MRI is suitable for diagnosing brain diseases in their early stage, for example brain tumours. Infarctions or infections can be successfully discovered using MRI, too [6]. Similarly to other modalities, MRI suffers from specific kinds of artifacts [6]: Partial volume, RF noise, Intensity inhomogeneity, Gradient, Motion, Wrap Around, Gibbs Ringing and Susceptibility. **In our method we decided to use MR data to segment brain**.

Data is obtained as a raw, dense set of 3D points, each of them having own intensity. Depending on used modality, its characteristics and examined area, such a set consists of millions of voxels. For example, authors in [4] characterize dataset containing over eight million points as medium-sized. Our own datasets obtained using MRI consists of over 8 380 000 points (more than 128 slices with resolution 256 x 256). Mostly, it is not possible to work directly with all the points from dataset, so image pre-processing techniques are used. Multiple techniques are described in the later parts. Such a huge amount of data is not only complicated from the machine-processing point of view.

Even for experienced radiologists it is complicated to abstract from tissues surrounding his or her current region of interest.

For radiologists it is important to quickly orient in such medical data and concentrate only on regions he or she is interested in. For this reason, it is extremely helpful to have a tool that extracts and renders only regions that are interesting in terms of current task. This is where segmentation comes to aid. If successfully applied, segmentation techniques extract only tissues or organs radiologist needs to work with.

Radiologists cooperate with other medical specialists, giving them information about specific patient's characteristics. For example, when there's a suspicion that there could be a tumour located in patient's liver, oncologist needs to know it for sure before he or she prescribes a chemical or radiation therapy.

Because of big responsibility that lies on radiologists, segmentation methods used in practice have to be reliable and precise. It is not acceptable that such a segmentation method hides a tumour or any other important information from region of interest (ROI). Radiologists wouldn't even trust such method [7] and wouldn't use it. Therefore, we believe, that some minimal human interaction will be needed to obtain optimal results as far as segmentation of medical data is concerned.

2.2 Theoretical background

2.2.1 Segmentation

According to [6], segmentation is a process of dividing digital image into regions (called segments), that share common or similar properties. Therefore, segment is a grouping of pixels (in 2D) or voxels (in 3D) that are in some way similar to each other. Due to simplicity, only term *pixel* will be used next in this section, as all the statements are analogous for voxels in 3D space, too.

It is important to add that pixels in one region often represent a part of the same real-world object (although it is not always true).

2.2.1.1 Similarity criterions

One of the most common and basic properties defining similarity among pixels is their intensity or grey level. Intensity of pixel is in the centre of one of the most basic and segmentation techniques – thresholding.

Similar to intensity of pixel are its brightness and colour, which can be defined in many different colour spaces. Choice of colour space can have dramatic impact on segmentation performance. SLIC [8], one of the most widely used oversegmentation algorithms, uses CIELAB colour space, as this closer to how human perceives differences between colours than RGB colour space [9].

More complex characteristics of pixels that can be used to determine which segment the pixel belongs to are based on neighbouring pixels. In [10] authors use average intensity of neighbouring pixels, differences of maximum brightness values and differences of minimum brightness values. In [11], Achanta et al. segmented mitochondria in electron microscopy image stacks. The authors first oversegmented 3D data obtained by electron microscope, and then merged these groupings (*supervoxels*) according to their similarity. Intensity histogram of voxels in supervoxel was chosen by authors as one of similarity criterions. Histogram of intensities of neighbouring supervoxel was chosen as the next criterion.

Not only criterions based on intensity of pixel or its neighbours can be used. In some cases, mutual distance of pixels in the digital image is important. An extreme example is a standard K-Means algorithm that ignores spatial information. If there were two disjunct green objects in the image, these would be clustered into the same cluster, although they are not connected to each other. On the other hand, slightly modified K-Means algorithm used in [8] oversegments image into superpixels. In the Figure 1 there are two versions of K-Means algorithm applied to the same image. The traditional version clusters image pixels into two classes, because it is only based on intensity information. The modified K-Means takes spatial information (distance of pixels) into consideration, so in result it produces oversegmented image, where every single segment has its own label.

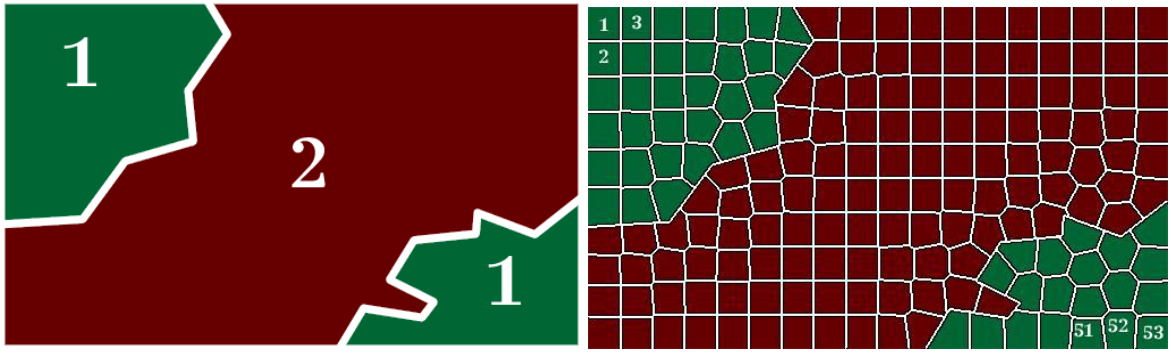


Figure 1- Standard K-Means algorithm and modified K-Means used in SLIC.

2.2.1.2 Segmentation techniques

Segmentation as a complex and difficult task can be performed on multiple levels. There are situations when only some objects on the scene are to be segmented, e.g. radiologist is only interested in some specific organ. Such segmentation is called *partial*. In [12], authors segment only brain tumours. On the other hand, *complete segmentation* splits digital image into multiple segments sharing some common qualities. An example of complete segmentation can be found in [13]. Wang et al. segment brain MR image into white matter (WM), grey matter (GM) and cerebral spinal fluid (CSF). In result whole image is segmented and every pixel except background (in MRI typically air) belongs to one of these three classes. Comparison of partial and complete segmentation can be seen in Figure 2.

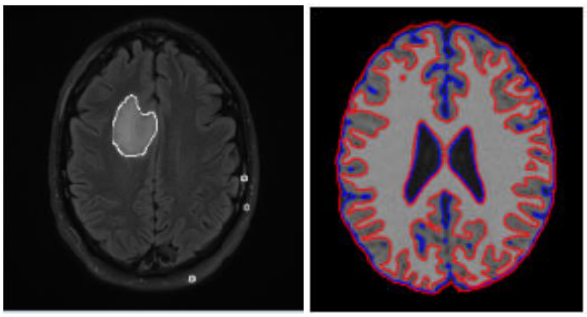


Figure 2- Partial segmentation of brain tumour [12] *and complete segmentation of brain* [13].

Image segmentation techniques can be classified from multiple viewpoints. Authors in [14] classify segmentation techniques into three basic classes: *threshold-based*, *edge-based* and *region-based*. Edge-based techniques can be further classified into *gradient-based* and *laplacien-based*. Region-based techniques can be classified into region-growing and classifier/clustering-based. Complete classification proposed by [14] can be seen in Figure 3. There are also segmentation techniques that are not mentioned: *Split/merge*, *morphological* methods, *template based* methods and *statistical* methods.

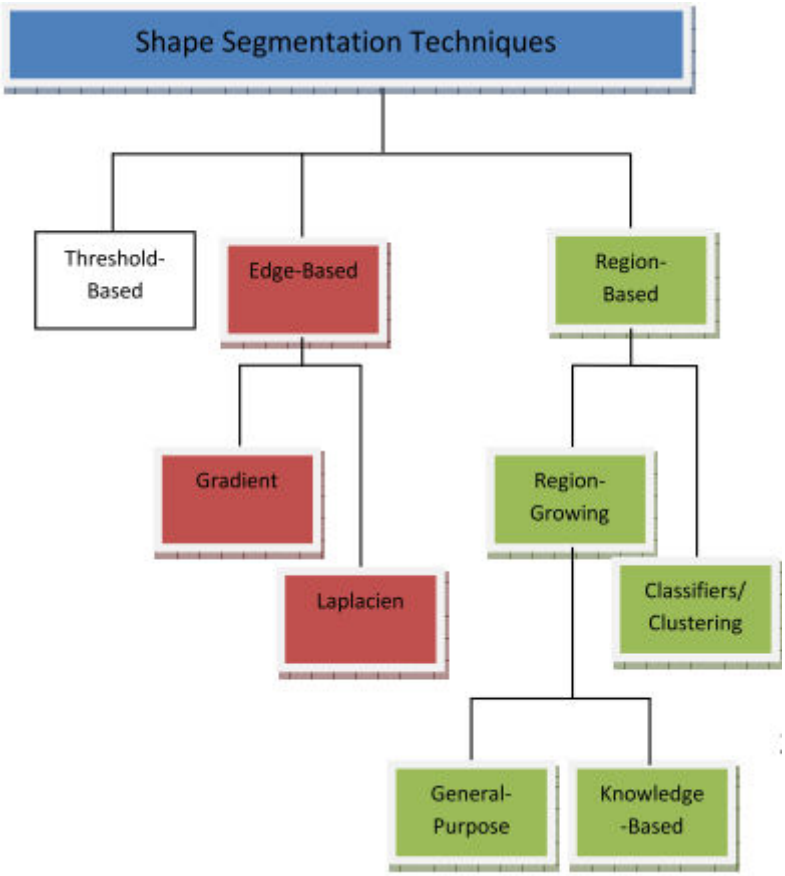


Figure 3- Segmentation techniques classification [14].

Thresholding is a segmentation technique based on idea that regions consisting of pixels belonging to the same real-world object share similar intensity, brightness or colour. In other words, these regions are homogeneous. In its very basic form, thresholding divides digital image into two classes – foreground and background.

Every pixel having intensity above chosen threshold is considered to be foreground, and every pixel having intensity below chosen threshold is considered to be background. Such kind of thresholding is also known as *binarization*. It is also possible to classify individual pixels to more than two classes. The whole range of possible intensities can be divided into intervals, which can (but do not have to) be of the same size, and every pixel is classified according to its intensity value to the class represented by concrete interval of intensity range. Many variations of thresholding can be found in [15]. An example of image segmented using thresholding technique in CIELAB colour space can be seen in Figure 4. Thresholding works well when pixels representing the same real-world object have similar intensity and pixels representing different real-world objects have distinct intensity [4].

Figure 4- Bulls segmented using thresholding technique in CIELAB colour space. Before (left) and after (right).

Edge-based techniques use information about edges detected by some kind of edge-detector. In [12], Canny detector is used to detect outer edges of brain tumour. Edge-based techniques are often used for finer detection of object edges after this object was coarsely segmented using some other technique. Edge-based techniques do not perform well when unclear or unclosed edges are present in the image.

Region-based methods construct regions by grouping spatially near pixels that are homogeneous in some criterion. Region-growing as a basic region-based technique requires initial seed as a base of region. Seed can be a single pixel or a cluster of pixels [4]. From the initial seed, region grows in every direction until a stop criterion is reached – either a boundary is found or a maximal region size is reached. If there is no a priori knowledge about the region being segmented, general-purpose region-growing technique is used. Performance of region-growing technique can be enhanced by using some additional knowledge, e.g. maximal intensity of pixels in region (no pixel with intensity higher than the maximal one will be added to region), statistical shape information (e.g. shape of organ segmented in MR image stack) etc.

Classification-based techniques have found a big use in segmentation tasks. Classification-based techniques are based on machine-learning, where some classifier, such as SVM, is trained to classify data into two or more classes. If there are only two classes, such classification is called *binary*. In [16], authors use SVM classifier to extract

abnormalities in breast MR images. In [17], authors first remove nonbrain tissue from MR images using a combination of anisotropic diffusion filtering, edge detection and morphological operations and then classify brain tissues into six classes using maximum a posteriori classifier. Achanta et al. in [11] trained SVM to classify supervoxels to three classes – boundary, background and mitochondria, whereby SVM does not return class for each supervoxel, but the probability of supervoxel belonging to each class. It is always necessary to specify a set of features characterizing individual classes. In [10] authors use average brightness of the block surrounding pixel, differences of maximum brightness and difference of minimum brightness.

Clustering-based techniques are often used as a preprocessing step in image segmentation. They are used to reach unsupervised classification. Clustering algorithms, compared to classification, do not in general require knowledge of the number of classes before clustering (although some implementations do). Clustering algorithms cluster pixels into clusters based on their similarity rate. Similarity rate is usually defined by the authors of clustering algorithm. Mostly, it is based on intensity homogeneity, brightness and colour, but spatial information can be used, too. Most common clustering algorithms are K-Means, Mean Shift, Fuzzy C-means and Gaussian Mixture Model. Achanta et al. successfully used SLIC to oversegment electron microscope data to supervoxels, decreasing number of elements to be processed by several orders of magnitude (original data consisted of 10^9 voxels, after reduction thousands of supervoxels) [11]. Currently, there are many clustering algorithms, each having its pros and cons. SLIC uses slightly modified K-Means algorithm.

2.2.2 Oversegmentation and supervoxels

Point in 3D space is called voxel. When defining supervoxel, we proceed from Achanta's et al. definition of superpixel qualities. Superpixel is a group of related pixels, that [8]:

- Adhere image boundaries.
- When used in preprocessing, they are fast to compute, memory efficient and simple to use.
- When used in segmentation, they improve speed and quality.

It is needed to add, that pixels belonging to the same superpixel are similar in some quality, mostly in intensity. Supervoxel is a group of related voxels with the same qualities. In our method, there is one more important quality of supervoxels - they should be as homogeneous as possible. In other words, intra-supervoxel variance should be low.

In Figure 5 there are examples of six state-of-the-art oversegmentation algorithms. Deeper comparison can be found in [5].

The most basic categorization of superpixel and supervoxel methods is:

- Graph based algorithms.
- Gradient-ascent based algorithms.

Every patient's scan consists of ca. millions of points in 3D space, each having its intensity. These points are called *voxels*. Usual scan consists of millions of voxels. In [11] authors process data obtained by electron microscope and segment mitochondria. Interesting is that the authors do not use voxels, but *supervoxels*, decreasing computational complexity by several orders of magnitude. Dividing volume into supervoxels is equivalent to *oversegmenting* this volume. Supposing supervoxels adhere to natural boundaries contained in volume, number of supervoxels (and therefore segments) is in general really bigger than the number of real objects in the volume.

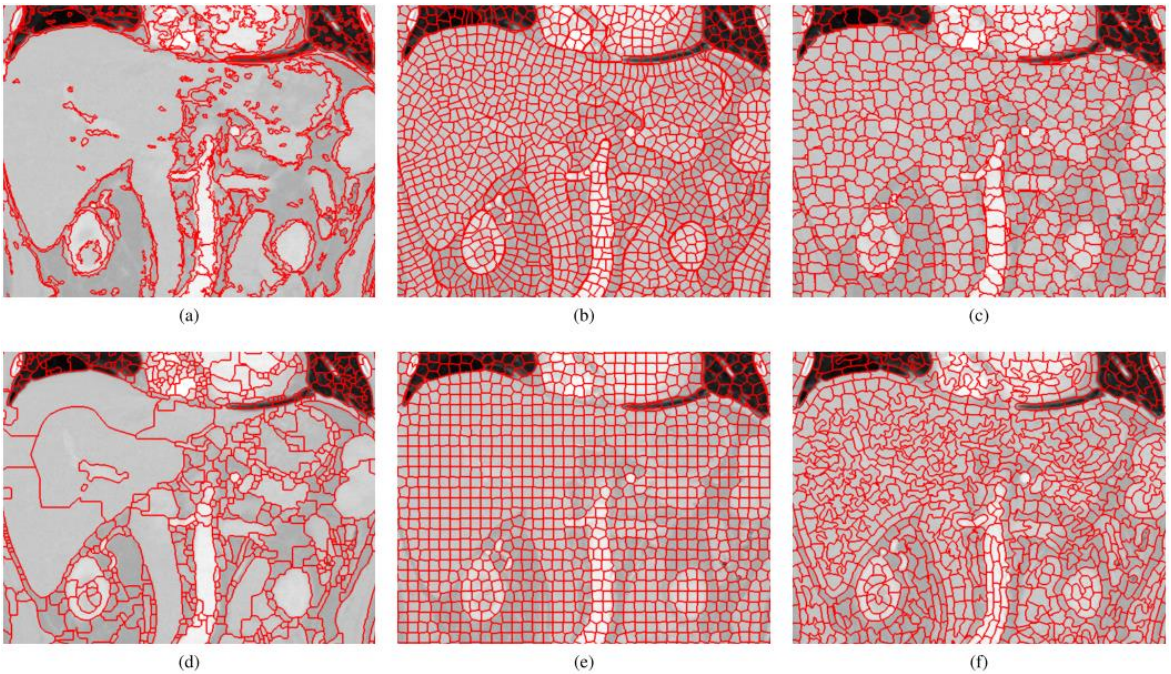


Figure 5 - Examples of different superpixel algorithms [5]. *(a)* - [18], *(b)* - [19], *(c)* - [20], *(d)* - [21], *(e)* -[22], *(f)* - [5].

Usually, the next step after oversegmenting the volume into supervoxels is to merge supervoxels into bigger unions, trying to reconstruct concrete objects (e.g. organs). This technique uses *merging* of supervoxels in order to reconstruct organ. Other approach is to classify supervoxels into desired classes.

In proposed method, we do not work directly with voxels, but first we oversegment volume into supervoxels. Because of its qualities, possibility to define number and size of supervoxels, we decided to use SLIC, which is thoroughly described in section 2.2.2.1.

2.2.2.1 SLIC Superpixels

Paper: SLIC Superpixels Compared to State-of-the-art Superpixel Methods [8]

Achanta et al. proposed a novel approach to oversegmentation of 2D images and 3D data volumes into superpixels and supervoxels.

SLIC, which stands for *Simple linear iterative clustering*, is a superpixel and supervoxel algorithm, that divides given volume (image) into supervoxels (superpixels). The main idea of SLIC is to use slightly modified K-Means clustering algorithm to divide image into homogeneous regions (superpixels). As stated in part 2.2.2, authors claim that good superpixels should:

- Adhere to image boundaries.
- When used as a preprocessing step, they should be fast to compute, memory efficient and simple to use.
- When used in segmentation tasks, they should increase both efficiency and performance.

This by nature simple algorithm based on adapted K-Means clustering allows user to define number (or inversely size) of desired superpixels. SLIC is fast compared to other similar algorithms, because for single cluster centre not all points in the volume are examined, but only those that lie in the region around the cluster centre. This dramatically reduces computational complexity of SLIC. In the Figure 6 it is explained, how the adaptation of k-means algorithm helps to create more regular superpixels and decrease computational time.

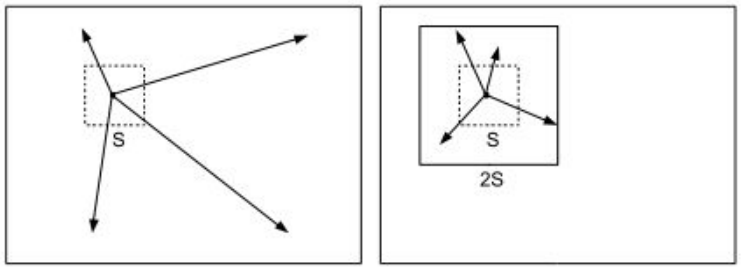


Figure 6 – Comparison of standard k-means algorithm and adapted version used in SLIC [8].

Compared to some other supervoxel and superpixel algorithms, SLIC takes spatial and intensity information into account when clustering voxels. It is possible to prefer superpixel homogeneity (higher weight has intensity) or compactness (higher weight has spatial distance from cluster centre).

Similarity of voxel to supervoxel centre is in SLIC defined by equation (1).

$$D = \sqrt{{d_c}^2 + \left(\frac{d_s}{S}\right)^2 m^2} \tag{1}$$

d_c is colour distance of voxel and supervoxel centre in CIELAB colour space defined as

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

d_s is spatial distance of voxel and supervoxel centre defined as

$$d_c = \sqrt{\left(x_j - x_i\right)^2 + \left(y_j - y_i\right)^2 + \left(z_j - z_i\right)^2} \tag{3}$$

m is parameter that adjusts importance of spatial and intensity proximity of voxel and supervoxel. With higher m the result are supervoxels with higher spatial consistency, which are similar in shape. Effect of adjusting parameter m can be seen in Figure 7.

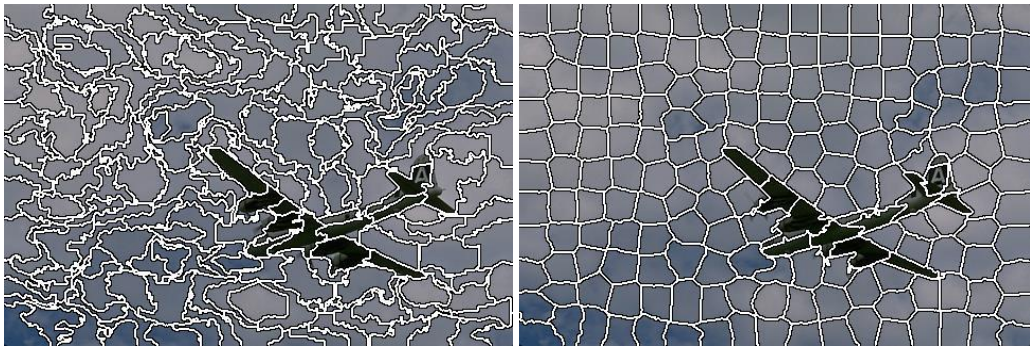


Figure 7 - SLIC superpixels. Left: low value of m. Right: high value of m.

2.2.3 Classification and clustering

Both classification and clustering are widely used as far as image segmentation is concerned. Classification is used when a specified set of classes is defined and pixels (in 2D) or voxels (in 3D) are to be assigned to one of these classes, depending on class and voxel characteristics. For example a brain MR image stack contains voxels that represent real-world brain tissues: white matter (WM), grey matter (GM) and cerebral spinal fluid (CSF). Basically, there are four classes: {WM, GM, CSF, BG}, where BG stands for background, typically air or surrounding tissues and skull. Each of these classes has its own typical characteristics. Typically, in MR images obtained via T1 scan intensity of WM voxels is highest followed by GM. The least bright are those belonging to CSF [10]. Intensity distribution of background has the highest standard deviation, as it contains darkest voxels (air) and also the brightest (skull).

On the contrary, clustering algorithms in general do not need to know exact number of classes. Typically, they only need some criterion (or more) that determines which cluster does concrete voxel belong to. There are many possible criterions – intensity or colour similarity, spatial distance (or their combination as shown in part 2.2.2.1), an edge detected between cluster centre and voxel etc.

Classification and clustering perform well when boundaries between classes (or clusters) are clear. In case of images with high level of noise or smooth gradient classifiers and clustering algorithms may fail. Figure 8 illustrates situation when SLIC (which uses slightly modified K-Means) was not able to find reasonable clusters due to high gradient.

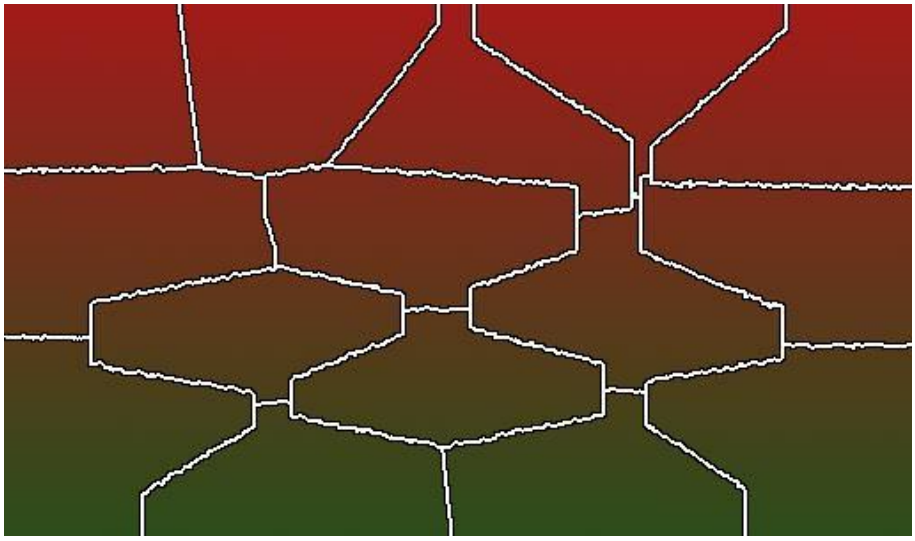


Figure 8 - Modified K-Means clustering applied on image with high gradient.

2.2.3.1 K-Means

K-Means is a clustering algorithm. In its basic form, K-Means does not take spatial information into account. Whole region is represented by its mean value (mean of values of included pixels), centroid or medoid and each pixel is represented by its own value of luminance.

K-Means is an iterative algorithm. Its steps are:

1. Randomly initialize cluster centres values (yet virtual mean values).
2. Calculate distance of every pixel in the image to every cluster centre. Distance is mostly defined as in equation (4). Every pixel is then assigned to the cluster, where this distance is minimal.

$$D = \left(x_n - \mu_j\right)^2 \tag{4}$$

3. Recalculate mean values of cluster centre for every cluster.
4. Go back to step 2 if mean values are not stable (old and new mean values differ too much).

Important to remember is that K-Means allows to set the number of clusters to be created before it is performed. On the other hand it does not guarantee that created clusters will form a connected graph. In Figure 1 pixels belonging to cluster 1 do not form a connected graph whereas pixels from cluster 2 do.

To overcome this limitation, K-Means can be defined on multidimensional space, allowing to separate clusters in way they form connected graphs. Achanta et al. use in their work [8] modified K-Means algorithm, that works on 5-dimensional space {L, a, b, X, Y} for 2D images and on 6-dimensional space {L, a, b, X, Y, Z} for 3D data, defining distance between pixel/voxel and cluster centre as shown in equation (1).

2.2.3.2 SVM

SVM, which stands for Support Vector Machine, is a binary classifier which requires supervised training. Single instances are represented as vectors of finite length. An

example is a supervoxel that can be represented by a 6-dimensional vector {mean_intensity, max_intensity, min_intensity, mean_x, mean_y, mean_z}.

SVM, in principle, tries to separate instances of one class from instances of other class by finding an optimal hyperplane.

A simple example is a set of linearly separable 2-dimensional vectors. Such situations illustrates Figure 9. In this case, no hyperplane is needed to separate instances of two classes.

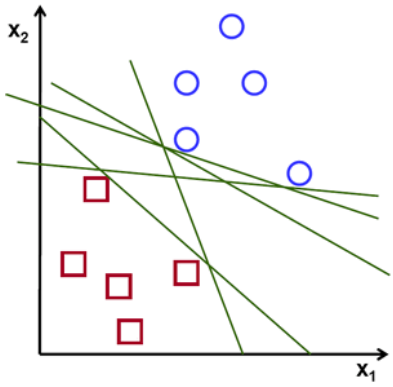


Figure 9 - Linearly separable instances of two classes [23].

Every line in Figure 9 separates instances of one class from instances of the other class. As already said, SVM tries to find an optimal hyperplane that would separate these classes. An optimal hyperplane should have maximal distance to nearest vectors of both classes being separated. Such separation should be least error and noise prone than hyperplane that would be too near to any instance of any class. An illustration of separating hyperplane can be seen in Figure 10.

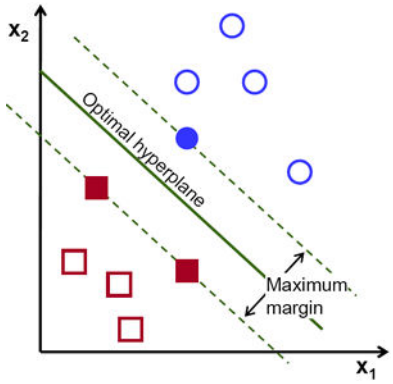


Figure 10 - Optimally separating hyperplanes and support vectors [23].

Vectors from both classes, that are nearest to the separating hyperplane, are called *support vectors*. Formally, hyperplane is defined as [23]:

$$f(x) = \beta_0 + \beta^T x \qquad (5)$$

β is the weight vector, β_0 is the bias and x are the training examples closest to the hyperplane (*support vectors*). Every value f(x) can be expressed as an infinite number of combinations of β_0 and β. As the result the one representation is taken, where the absolute value of equation (5) is equal to 1, as expressed in equation (6).

$$|\beta_0 + \beta^T x| = 1 \tag{6}$$

Now, the distance of vector x and hyperplane (β, β_0) is equal to 1:

$$D = \frac{|\beta_0 + \beta^T x|}{\|\beta\|} \tag{7}$$

From equations (6) and (7) arises fact that the distance of support vectors from separating hyperplane is

$$D\_support\_vectors = \frac{1}{\|\beta\|} \tag{8}$$

From the Figure 10 and equation (8) it is clear, that width of margin is equal to

$$M = \frac{2}{\|\beta\|} \tag{9}$$

The problem of maximizing M in equation (9) is the fundamental principle SVM is based on. It is solved using Lagrangian optimization [23].

Many modifications of SVM exist. Originally, SVM is a binary classifier and therefore it is only able to classify into two classes. Currently there are a lot of variations of SVM, allowing to classify to more than two classes or to perform fuzzy classification, allowing each pixel to belong to more than one cluster. In case of fuzzy classification, every pixel has a degree of membership to every cluster. Finally, pixel is assigned to that cluster its membership degree is the highest. Various similarity (or inversely distance) metrics can be used, too.

An example of use of SVM in segmentation tasks can be found in [24]. Authors use a combination of knowledge-based techniques and multi-spectral analysis based on SVM to detect brain tumours. In [16] authors use SVM to detect abnormalities in breast MRI. Authors claim that SVM has 20% higher performance rate than other classifier, decision tree, but on the other hand, the recall of SVM was 12% worse than that of decision tree.

2.2.3.3 Neural networks

Principle of neural networks is inspired by how the human brain works. The most commonly used form of neural networks is a multi-layer perceptron (MLP). Whole perceptron consists of nodes called *neurons* and their connections. Each neuron has following structure (Figure 11):

- Neuron has multiple weighted inputs from neurons from previous layers
- Neuron has multiple outputs to neurons in the next layer

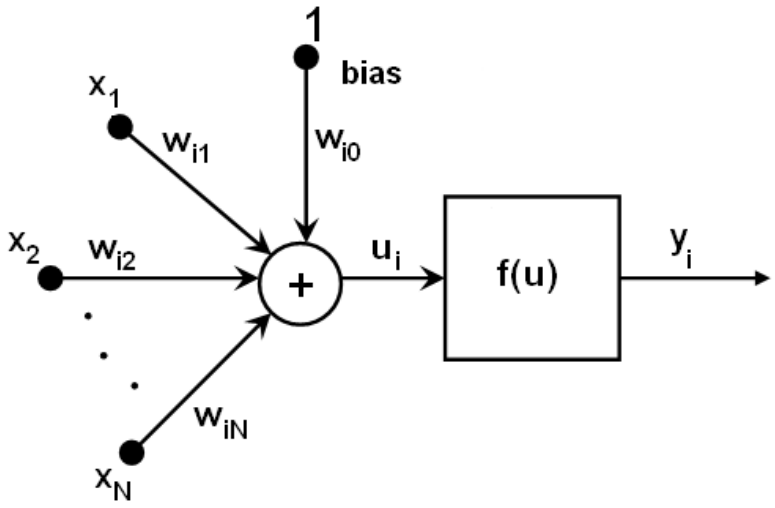


Figure 11 - Neuron structure [25].

The neurons are arranged in layers. The input and output layers are visible to the outer world, while there are one or more hidden layers between these layers (Figure 12). Each hidden layer consists of neurons linked to neurons in previous layer and to neurons in the next layer. Networks in which every neuron is connected to all neurons in previous layer and in next layer are called fully connected neural networks.

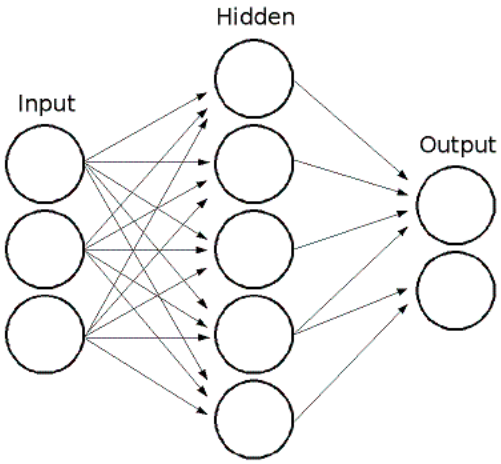


Figure 12 - Multi-layer perceptron [25].

Each neuron in neural network works as following:

1. Neuron sums all inputs (multiplied by weight of connections)
2. Neuron transforms this sum according to its activation function (which may be identity, sigmoid, gaussian or other)
3. Neuron sends transformed value to its outputs

It important to note, that the number of neurons in input layer is equal to the dimension of feature vector. For example, if a feature vector characterizing supervoxel is {mean_intensity, max_intensity, min_intensity}, then there will be three neurons in the input layer. The number of neurons in the output layer is equal to the number of classes that the MLP is supposed to classify to.

To train MLP, it is necessary to have training set including enough input feature vectors (characterizing classified objects) and corresponding output vectors. After MLP classifies feature vector, it is compared to the ground-truth output vector and weights of connections between neurons are adjusted through back-propagation. Different algorithms for weights adjustment can be used. In our method we use Levenberg-Marquardt algorithm.

In the classification phase each output neuron becomes excited at some level. This excitation depends on values of scalars in feature vector. The excitation rate is proportional to probability of instance represented by input feature vector to belong to class represented by output neuron.

2.2.4 Segmentation in medical imaging

As stated in 2.1, digital images used in medical practice can be obtained using wide range of modalities. In this work, we concentrate on those obtained using MR.

Medical images have some specific characteristics:

- Higher dynamic range.
    - Common images in greyscale have 8-bit representation, medical images usually 12-bit.
- Specific kinds of noise.
    - Noise alters voxel intensity and negatively influences classification.
- Partial volume averaging.

Especially at the boundaries, single voxel contains a mixture of tissue classes [6], becoming difficult to classify correctly (such a voxel does not even have to belong to some class). This effect is cause by fact that spatial resolution of a voxel is much higher than the size of contained anatomical structures.

3 State of the art medical segmentation methods and techniques

In this part, we introduce some state-of-the-art segmentation techniques and methods. Special attention is paid to brain segmentation techniques and methods similar to our proposed one.

As every part describes one individual work, we only quote the described paper in the very beginning. Otherwise every description would contain a lot of brackets which would dramatically decrease its readability. Every equation and picture contained in every description comes from the described paper.

3.1 Over-Segmentation based on Monogenic Cues

Paper: Over-Segmentation of 3D Medical Image Volumes based on Monogenic Cues [5]

Similarly to SLIC, monoSLIC, method proposed by authors of this work, is based on modified version of K-Means. MonoSLIC also allows to perform oversegmentation to supervoxels on 3D volumes (it is possible to oversegment 2D image to supervoxels, too). It also allows to choose number and size of supervoxels.

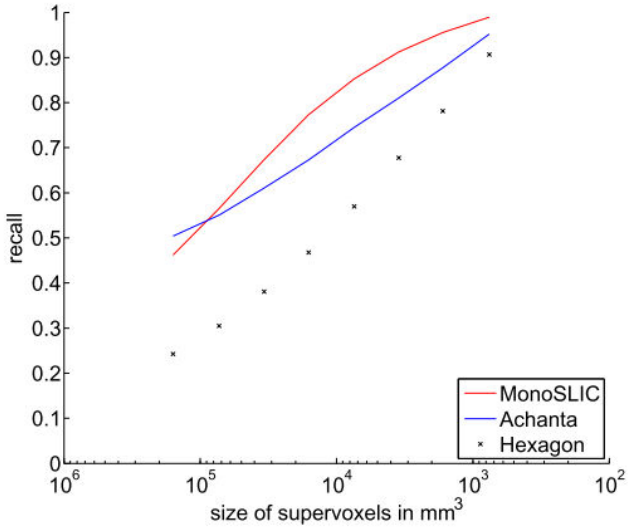


Figure 13 - Recall of monoSLIC is higher than recall of SLIC when supervoxel size becomes smaller [5].

The main difference to SLIC is that monoSLIC transforms image to its monogenic signal that represents the image. Authors claim that such representation is invariant to contrast and brightness changes. If a proper kernel size is chosen, most dominant edges are highlighted.

In result, authors claim that monoSLIC yields regular, robust to noise, homogeneous and edge-preserving oversegmentation of an image or 3D volume. Specially, better results were reached when size of supervoxels was smaller (Figure 13). As far as performance is concerned, compared to other state-of-the-art methods monoSLIC

becomes the fastest one. Authors also claim that monoSLIC is very suitable for work with medical images, which makes it a possible candidate to be compared with SLIC in terms of our method and future work.

3.2 Supervoxel-Based Segmentation of Mitochondria

Paper: Supervoxel-Based Segmentation of Mitochondria in EM Image Stacks with Learned Shape Feature [11]

In this work, Achanta et al. used SLIC in process of segmenting mitochondria from EM image stack. 3D volume from this work contained approximately 10^9 voxels, namely 1000 images each having resolution 1024x1024.

Authors decided to use oversegmentation as a preprocessing step and use created supervoxels instead of single voxels. This step decreased computational complexity by several order of magnitude (authors claim that computation became 1000 times more effective).

Authors trained SVM to classify supervoxels to three classed – mitochondria, mitochondria boundary and background. Feature vector consists of two parts:

- Ray descriptors
    - Distance: normalized distance to the boundary.
    - Norm: normal of gradient in the boundary voxel (crossed by ray).
    - Ori.
- Intensity histograms.
    - Histogram of supervoxel itself.
    - Histograms of neighbouring supervoxels.

Method also uses techniques of graph-cuts. Nodes are supervoxels and edges are neighbourhoods between supervoxels.

From the point of view of our method, we find decrease of computational complexity caused by using supervoxels instead of single voxels very interesting. In our opinion this is proof that our decision to use similar approach is right. We also incorporate intensity histograms in order to describe supervoxels as far as intensity distribution is concerned.

3.3 Unsupervised Segmentation for MR Brain Images

Paper: Unsupervised Segmentation for MR Brain Images [10]

The main purpose of method proposed by authors is to segment brain tissues (CSF, GM, WM) from MR images. Authors based their method on 1-D SOM (Self Organizing Maps) and ART (Adaptive Resonance Theory), which is based only on brightness distribution of MR images.

In this method brightness difference of brain tissues is widely used. As authors segmented T2-weighted MR images, CSF is the brightest followed by GM, then WM. Surrounding tissues are darker than these three brain tissues. Due to these facts authors claim that brightness of pixels is the most informative property. As a feature vector, not the brightness of individual pixel is used, but characteristics of whole neighbourhood of pixel, called *block*. Four features were proposed, whereby every one of them serves another purpose:

- Brightness of pixel itself.
- Average brightness of pixels in block.
- Difference of maximum.
    - Contributes to detection of boundary from the pixel to the tissue with higher brightness.
- Difference of minimum.
    - Contributes to detection of boundary from the pixel to the tissue with lower brightness.

These features characterizing pixel and surrounding block are input to the overall classification process proposed by authors. First step of this process is 1D-SOM. Output from the first step is nonlinearly quantized resulting into vector of weights. Weights are after next processing used in Fuzzy ART. Overall result is membership of pixel in one of defined classes (brain tissues). The overall process can be seen in Figure 14.

One of the challenges authors had to face was fact that boundaries between tissues (e.g. between GM and WM) are not always clear. Voxels near to boundaries between tissues also suffer from partial volume effect.

Authors also performed experiment studying influence of block size taken into consideration when extracting feature vectors. For image resolution 512x512, which is currently the most used in MR and CT, authors recommend to use 5x5 neighbourhood of pixel as block.

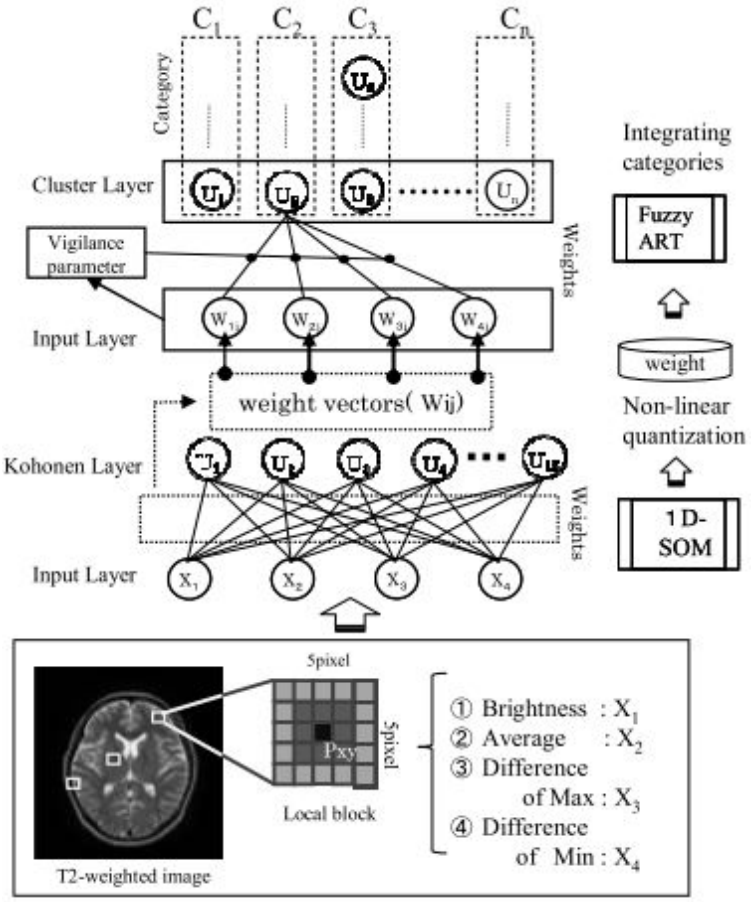


Figure 14 - Method process overview [10].

3.4 Brain MR Segmentation Using Local and Global Intensity Fitting

Paper: Brain MR Image Segmentation Using Local and Global Intensity Fitting Active Contours/Surfaces [13]

This region-based brain segmentation method operates on MR images and uses contour/surface model. It relies on image intensity information using both local information about pixel / voxel neighbourhood and global information.

Authors compare their results with *piecewise models* (PC) models that are fast to compute, but their most serious limitation lies in fact, that PC models assume that segmented tissues have statistically homogeneous intensities. Therefore PC models are inappropriate in situations when intensity inhomogeneity occurs. Similar to PC model is the use of global intensity information. Inhomogeneity of different brain tissues and intensity similarity at boundaries between them make PC not suitable for brain segmentation.

To eliminate the problem with intensity inhomogeneity, authors use local intensity information as well. Local intensity information is based on local binary fitting model [26] (LBF) that utilizes two varying fitting functions (approximations of local intensities at two sides of contour). Local intensity information helps to solve intensity

inhomogeneity problem, but makes LBF models more sensitive to initialization than PC.

Global and local intensity information is used in energy function with two terms: local intensity fitting term that attracts contours and stops at image boundaries and global intensity fitting term that drives contour from object boundaries and makes contour initialization more flexible. Effect of global and local terms can be seen in Figure 15.

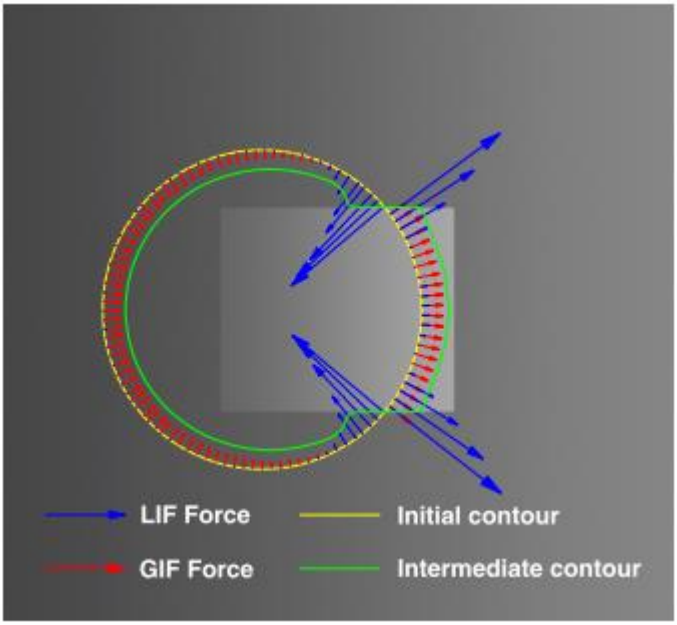


Figure 15 - Effect of global and local terms [13]. *LIF force is dominant near the object boundaries, while the GIF force is dominant at locations far away from object boundaries.*

Authors extended their *local and global intensity fitting* (LGIF) model to multi-phase level set formulation, so that WM, GM and CSF could be segmented simultaneously.

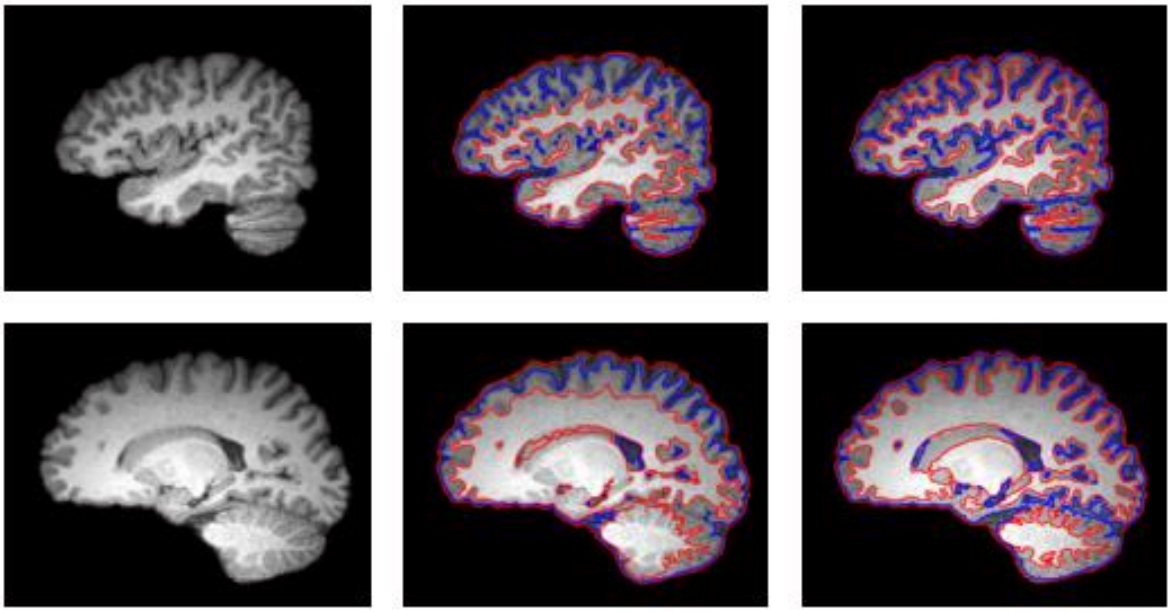


Figure 16 - Comparison of PC (middle column) and proposed method (right column) [13].

From point of view of our method we find interesting idea of combining global and local intensity information. We share the same opinion that to exactly discover boundaries between brain tissues we will need to combine the best of both approaches – global and local.

3.5 Discriminative Clustering and Feature Selection for Brain MRI Segmentation

Paper: Discriminative Clustering and Feature Selection for Brain MRI Segmentation [27]

In this work from May 2015 authors use supervoxels produced by SLIC to segment brain into four classes – BG, CSF, GM and WM. Second property that make this work similar to ours is that authors evaluate their method on the same dataset as we do – IBSR18.

Authors propose two methods utilizing supervoxels – Information Theoretic Discriminative Segmentation (ITDS) and Supervised Information Theoretic Discriminative Segmentation (SITDS), where ITDS is fully unsupervised and SITDS is a supervised variation.

Features that authors use to cluster / classify supervoxels into individual classes are based on intensity, shape and texture. In result their feature vector consists of 228 values dimensions.

SLIC was the algorithm of choice in this work. Authors defined supervoxels of size of about 2700 voxels. In our opinion such size has negative impact on segmentation performance as it increases intra-supervoxel variance of voxels as far as ground truth classes of individual voxels inside supervoxels is concerned.

Thorough description of algorithm with pseudo-code can be found in paper. Therefore we do not show it here. What we find important to show are segmentation results of ITDS and SITDS. These results can be seen in Figure 17. As authors compared their method with other state-of-the-art algorithms for brain segmentation on standard datasets and as they claim that their method outperformed other state-of-the-art methods, **we decided to compare our method with this one, using standard IBSR dataset.**

PERFORMANCE OF DIFFERENT SEGMENTATION METHODS ON IBSR AND BRAINWEB DATASETS

Datasets	IBSR				BrainWeb			
Methods	CSF	GM	WM	time(s)	CSF	GM	WM	time(s)
*k*Means	0.51±0.06	0.75±0.06	0.78±0.04	8	0.86±0.03	0.84±0.03	0.82±0.04	12
MI	0.52±0.08	0.79±0.04	0.80±0.03	19	0.87±0.02	0.86±0.02	0.85±0.02	23
MRF	0.53±0.06	0.76±0.03	0.87±0.03	521	0.89±0.02	0.90±0.01	0.91±0.01	636
ITDS	0.60±0.05	0.81±0.03	0.86±0.02	26	0.92±0.01	0.92±0.01	0.93±0.01	32
WPNN	0.63±0.03	0.83±0.02	0.87±0.03	92	0.93±0.02	0.93±0.01	0.91±0.02	151
SITDS	0.67±0.03	0.86±0.01	0.89±0.02	29	0.94±0.01	0.95±0.01	0.94±0.01	35

Figure 17 - Comparison of ITDS and SITDS with other state-of-the-art brain segmentation methods.

3.6 Superpixels in Brain MR Image Analysis

Paper: Superpixels in Brain MR Image Analysis [28]

As far as computational complexity is concerned, medical image analysis usually relies on algorithms that are highly computationally complex. Authors in this work proposed a use of superpixels as an elementary entity instead of pixels (or voxels). Achanta et al. in [11] proved that use of supervoxels instead of voxels decreased computational complexity by several orders of magnitude.

In first step authors evaluated different superpixel techniques. Some of them were considered too computationally expensive (N-cuts, Quick shift and Felzenszwalb's method). SLIC with its linear complexity O(N), ability to adhere to boundaries and possibility to control desired number of superpixels was chosen as a reference method.

Authors observed multiple aspects of such (superpixel) representation of an MR volume. In context of our work we found interesting these:

- Relation between under-segmentation error and reduction in complexity
- Relation between segmentation performance and reduction in complexity

To quantify segmentation accuracy (segmentation performance) authors used Jaccard overlap metric between obtained tissue segmentation and ground truth segmentation, given by equation (10).

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{10}$$

Authors defined undersegmentation error as,

$$U = \frac{1}{N}\left[\sum_{i=1}^{M}\left(\sum_{s_j | s_j \cap g_i > B} |s_j|\right) - N\right] \tag{11}$$

where N is the number of supervoxels and M is number of MR volumes. $s_j \cap g_i > B$ denotes all supervoxels that overlaps with segmentation of tissue *i* in at least B percent of voxels contained in supervoxel. In other words undersegmentation error increases with every supervoxel that overlaps with more than one tissue.

In Figure 18 we can see relation between under-segmentation error and reduction in complexity. It is clear that error increases as computational complexity decreases. It is always necessary to find equilibrium between these two quantities, as for price of 10% under-segmentation error comes benefit in form of 50-times lower complexity.

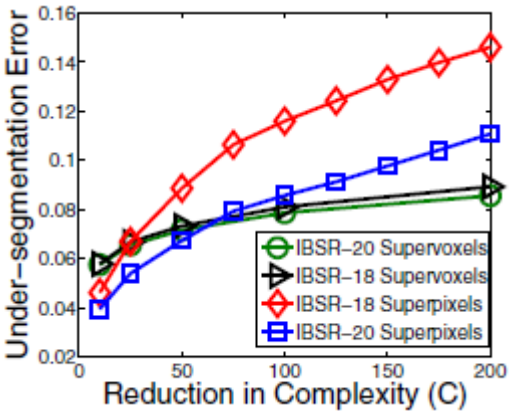


Figure 18 - Relation between reduction in complexity and under-segmentation error.

Authors observed not only influence of use of superpixels on segmentation, but they also used supervoxels in MR segmentation task. Relation between reduction in complexity and performance of segmentation of different types is in Figure 19. Authors observed the same fact as we do (in 5.3.3) – segmentation performance decreases as complexity becomes reduced. We did not measure reduction in complexity directly, but we measured influence of supervoxel size on segmentation success rate. Computational complexity is directly connected with supervoxel size (higher size means lower complexity).

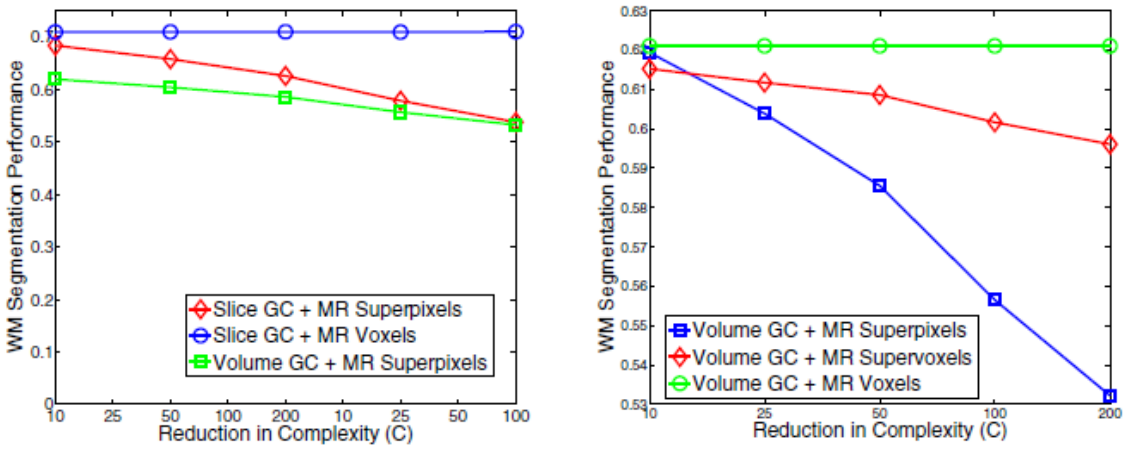


Figure 19 - Relation between reduction in complexity and segmentation performance.

Similarly to our results authors claim that with increasing supervoxel compactness decreases the segmentation success rate.

In context of our work it is also important to say that authors used IBSR dataset.

4 First approaches

4.1 Approach based on merging of supervoxels

4.1.1 Method overview

In this part we proposed a method for organ segmentation from medical images based on oversegmentation of MR volume into supervoxels. As the second step we merged neighbouring supervoxels if they were similar. In this approach we also proposed a method of witnesses which will no longer be used in our final method.

Due to high complexity of calculations, we decided to work with supervoxels rather than voxels, decreasing computational complexity by a few orders of magnitude. The next benefit of working with supervoxels is decreasing of noise. Supervoxel itself is represented by mean values (or other statistical values) calculated from individual voxels belonging to this supervoxel, as there is a good chance that the influence of outliers will be supressed or even eliminated.

To classify single supervoxels to classes representing different tissues, we train classifier SVM. We also plan to train at least one more classifier, neural network, to compare their results. Features representing supervoxels in training process are based on intensities of voxels contained in supervoxel. We also consider using characteristics of neighbouring supervoxels, too.

Next important (and novel) method is use of ***witnesses***. Witnesses are thoroughly described in part 4.1.2. Witnesses are used in combination with classifier in merge procedure. They describe shape of organ in scale invariant way.

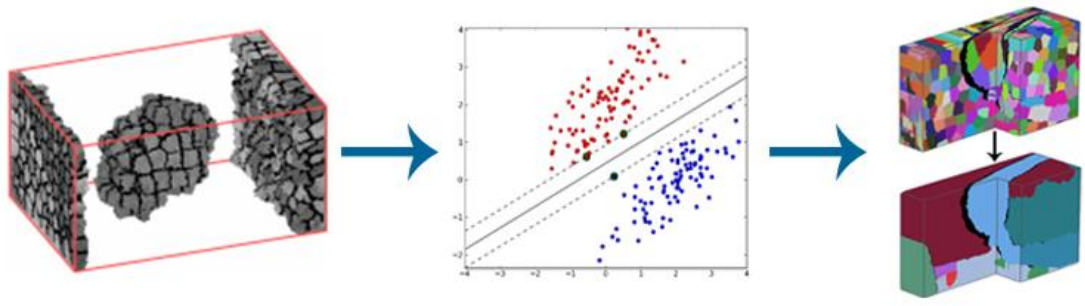


Figure 20 - Method overview. Oversegmentation - Classification - Merging supervoxels [11].[1][2]

4.1.2 Merging of supervoxels

To build organ from single supervoxels, we use the third step of our method – merging supervoxels. In the beginning of merging, algorithm needs to know the initial

[1] http://www.mblondel.org/images/svm_linear.png
[2] http://www.danvil.de/images/science/dasv.jpg

supervoxels that surely belongs to organ. We involve user that gives our method an initial seed. Our method relies that the initial seed surely belongs to segmented organ.

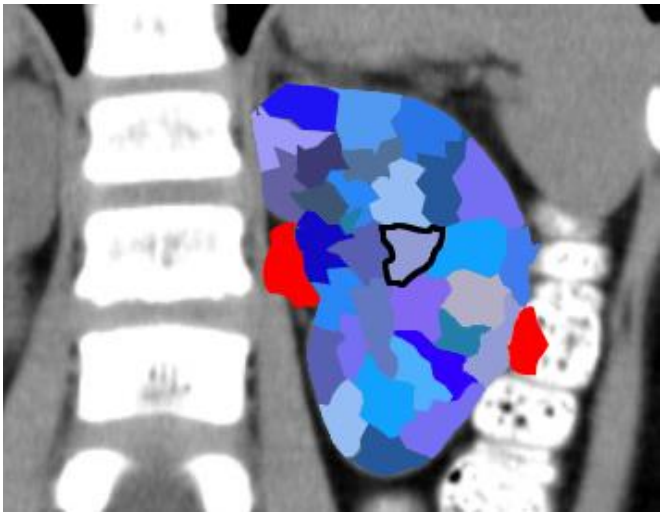


Figure 21 - Initial seed. Our method relies on user that initial seed really belong to segmented organ.[3]

Given initial seed, in the next step method examines every neighbouring supervoxel. If SVM classifies supervoxel as brain tissue, this supervoxel becomes a candidate that can be added to already merged supervoxels. If more structures are classified as candidates, witnesses come to aid.

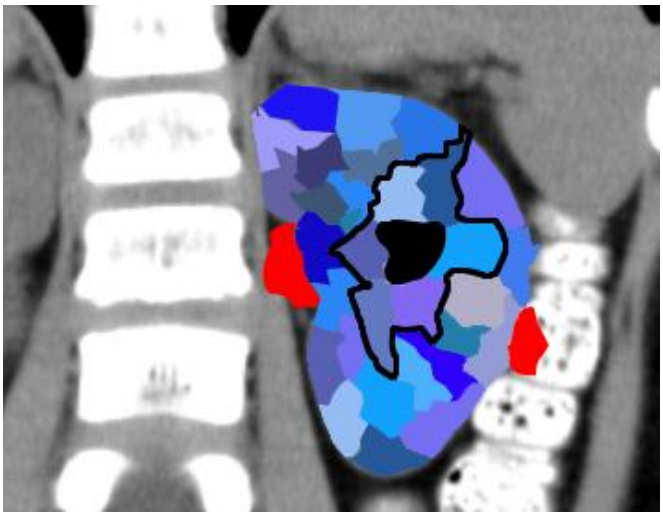


Figure 22 - Neighbouring supervoxels classified as brain tissue. They become candidates[3].

Let's have yet merged supervoxels that were till now merged to builded organ. Such structure is also a supervoxel and has its known centroid and boundaries. In this situation, we imaginary cast rays from centroid to boundaries in regular way (we are in 3D space, so these rays will be casted in a spherical angles). From centroid, two parallel lines having centroid as common point are casted. Each of these two lines has its own length from centroid to boundary. Ratio between these two lines is called

[3] http://www.kidneystoners.org/wp-content/uploads/2012/02/pediatric-kidney-stone-CT.jpg

witness. Shape of single organ is described by set of witnesses, where their number can be chosen by user. Illustration of witnesses in 2D space can be seen in Figure 23. The ground-truth ratios (witnesses) is calculated from set of manually segmented organs of different patients.

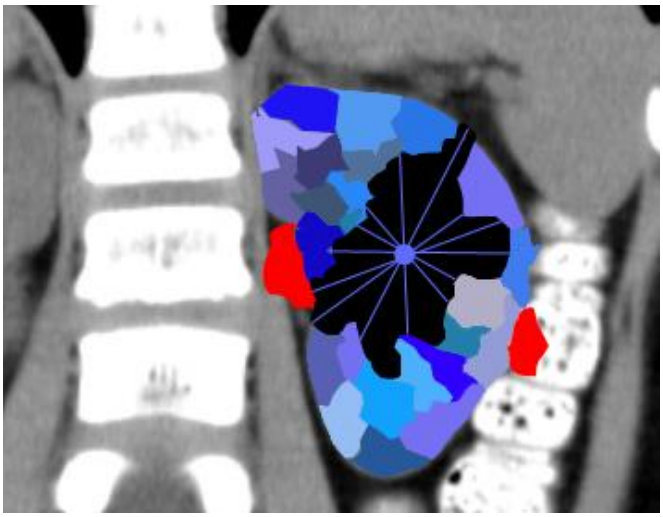


Figure 23 - Illustration of witnesses in 2D space[3].

For every candidate, following is done:

1. Try to add candidate to already merged supervoxels.
2. Recalculate centroid.
3. Calculate witnesses.
4. Calculate RMSE of witnesses by ground-truth witnesses.

The candidate with lowest RMSE will be merged into.

Merging procedure ends after there is no other neighbouring supervoxel to be added, or after more than 90% witnesses are close to ground truth witnesses, which means, that the merged supervoxel has approximately shape of ground-truth segmented organ.

In this prototype we implemented basic steps from method based on merging of supervoxels (part 4.1). This prototype was not further used in DP2 and later parts of this project. Nevertheless it gave us experience and knowledge that we used later.

First prototype was created in subject DP1. It is designed to easily load and process medical images in different formats. To reach this, ITK library was used. To present processed images OpenCV was used.

The first prototype was focused on creating basic infrastructure that will be later used as base of entire application. This goal was completed in the early stage of DP1, so we perform first experiments to examine structure and properties of medical data.

In the first step, we loaded medical data. Currently we use NIFTI format. Loading was easy thanks to ITK, which supports many formats used in medical imaging. Next, we had to decide, how to present loaded data. Problem is, that medical images are mainly 12-bit images, but common displays (such as the one used by us) only support 8-bit depth per channel. In this early phase we only shift value of each pixel (we are displaying in 2D, so we use term pixel for this purpose) 4 bits left, getting 8-bit representation. In the future, we consider using some more advanced approach, such as power law transformation. An example of MR image can be seen in Figure 24.

The next step was transformation of 3D ITK volume to SLIC compatible format, which is *unsigned int***. To perform this, we created conversion module SLICBridge.

In the next step, we used SLIC to oversegment given 3D volume, creating supervoxels. Each supervoxel has its own label, distinguishing it from other supervoxels. In the Figure 25 (left) we can see oversegmented image from Figure 24. We prefer intensity homogeneity before shape consistence of supervoxels.

It can be seen, that SLIC superpixels adhere image boundaries and adapt their shape so they follow homogeneity of region rather than shape consistence. In the next step, we merged supervoxels their average intensity was similar. We also took spatial information into consideration, so we only merged neighbouring supervoxels. The merging procedure was following:

1. Choose randomly one supervoxel SV.
2. Go through all its neighbours N.
3. If ratio of SV.intensity and N.intensity >= 0.95, merge N into SV (not only points are added to SV, but also neighbours of N become neighbours of SV).
4. Repeat given number of times or until convergence.

Although intensity similarity criterion is very simple, its results are quite good and in the future will be possibly used to coarsely segment organs before applying more precise and fine method proposed in chapter 0. In the Figure 25 (right) we can see results of merge procedure.

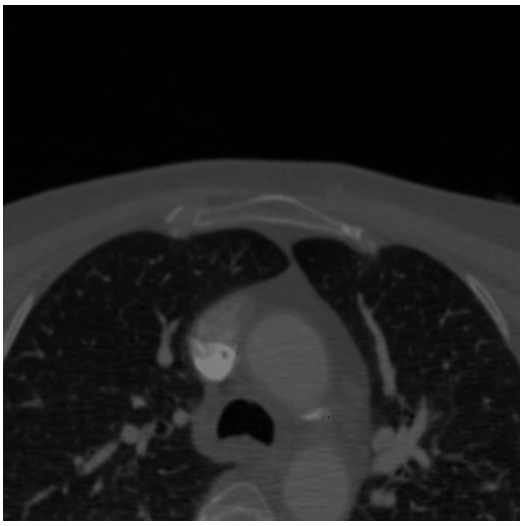


Figure 24 - MR image of abdomen.

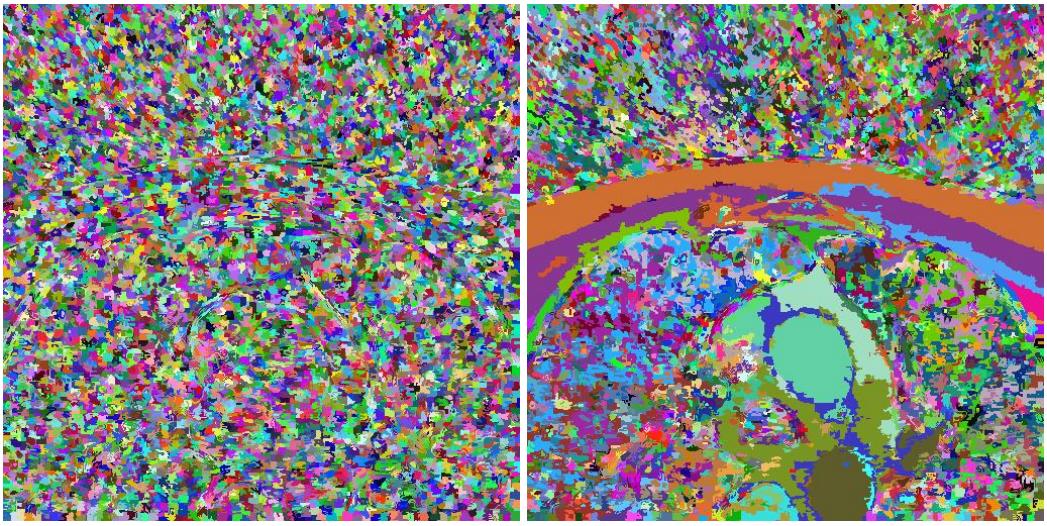


Figure 25 – Left: Oversegmented MR image. Right: Result of merge procedure on supervoxels.

4.2 Approach based on use of multiple classifiers

One of the approaches that we considered is based on use of multiple classifiers. On that account we defined three different classification approaches based on two classifiers-SVM and MLP. Pipeline proposed in this section was not completely implemented, as we decided to use only MLP in the finally proposed method.

4.2.1 SVM

In the first approach we only used SVM to classify supervoxels into BG, CSF, GM and WM classes. As the output of SVM contains only class into which supervoxel was classified, we only assign this class to given supervoxel.

4.2.2 MLP

MLP gives us a better image of level of certainty of classification. Its output neurons become excited according to feature vector of supervoxel being classified. Supervoxel is considered to be significantly excited if its level of excitation is greater than a given threshold. We recognize two levels of excitation:

- **Level 1**: excitation of an output neuron is greater than 0.40.

- **Level 2 (significant excitation)**: excitation of an output neuron is greater than 0.70.

If the feature vector matches some class with high level of certainty, only one output neuron becomes significantly excited (level 2) and no other neuron becomes excited. Otherwise no neuron reaches excitation level 2 or more neurons reach some level of excitation.

In the first case is the classification task trivial. We only assign supervoxel to the class represented by output neuron with highest excitation rate.

In the second case, we find classification uncertain if either no neuron was significantly excited or more than one neuron was excited at level 1 or level 2. If such situation occurs we are not able to certainly classify given supervoxel and therefore such supervoxel remains unclassified.

4.2.3 SVM and MLP

Sometimes it happens that MLP is not able to classify supervoxel with required level of certainty (as described in 4.2.2). In such cases we use SVM to classify given supervoxel to a concrete class. This procedure can be described by following pseudocode:

```
classes = ARRAY.CREATE

FOR each fv in supervoxels.features
        response = MLP..classify(fv)

        IF number of level 1 excitations in response <> 1
                PUSH(classes, SVM.classify(fv))
        ELSE IF number of level 2 excitations in response <> 1
                PUSH(classes, SVM.classify(fv))
        ELSE
                PUSH(classes, get_class(response))
        END IF
END
```

Pseudocode 1 - Classification using SVM and MLP.

5 Method proposal

5.1 Input data

In clinical practice clinicians use various data formats. Most of them do not only contain data from concrete examinations (e.g. voxel intensities in MR volume), but also information about the patient such as his or her name, age, gender etc. According to [29] there are four major file formats currently used in medical imaging: *Analyze, Nifti, Minc and DICOM*. Authors in this work categorize these formats into two groups:

- Formats for standardization of images generated by modalities (DICOM).
- Formats for facilitation of postprocessing analysis (Analyze, Nifti, Minc).

Analyze file format is the oldest of these four formats. It used to be the standard for medical imaging post-processing [29].

Minc is the most locally used from these four standards and therefore we do not further discuss it, as we do not plan to support it.

DICOM is the most general and most robust of these formats. According to its documentation, it is not only format for storing data, but also a protocol for network communication [30]. DICOM has become the main part of infrastructure of most medical imaging departments all around the globe [29]. Similarly to Nifti, Dicom allows (and actually forces to) store metadata and patient's data to be stored together with the medical imaging data from concrete examination. The philosophy of DICOM is that data without metadata that describe context is meaningless. Interesting is that DICOM allows to use standard image formats to store data (such as JPEG-2000), just by wrapping them into DICOM shell [29].

Nifti can be considered to be the next incarnation of Analyze. Nifti allows to store data and metadata in two different ways: it is possible to store data and metadata in separated files or to store both of them in one file. In most cases both types of data are stored in one file [29]. As stated above the main purpose of Nifti data is to use them for medical imaging post-processing. In comparison to Analyze, Nifti contains more information about the context of examination – the most interesting for us is orientation of patient and spacing between voxels in a slice and between slices. Data in Nifti can be stored in integer of floating point format. **Later in this work we use Nifti as a medical imaging data format.**

It is important to note that data from a particular examination can be stored in different orientations. Each specialist can use different orientation. For example, neurologists use "RAS" convention for axes whereby radiologists use "LAS" [31]. Nifti allows to store data in different orientations and also to contain related metadata. In IBSR different types of volumes are stored in different orientations. Figure 26 explains the meaning of different orientations.

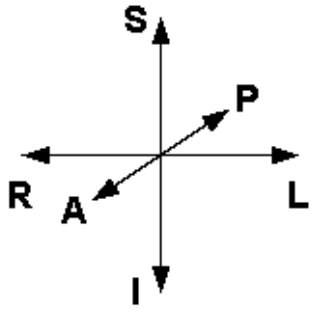
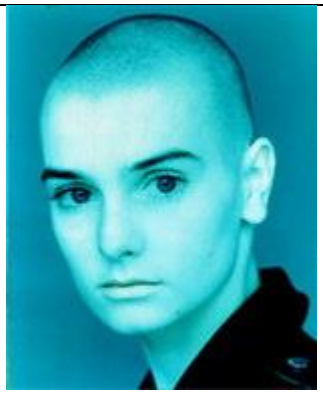
<table>
<tr><td colspan="2"> 
</td><td>  </td></tr>
<tr><td>Right</td><td>Left</td><td rowspan="3">Head orientation reference</td></tr>
<tr><td>Anterior</td><td>Posterior</td></tr>
<tr><td>Inferior</td><td>Superior</td></tr>
</table>

Figure 26 – Directions in medical imaging [31].

5.1.1 IBSR dataset

In order to evaluate our method we needed to find a standard annotated dataset of brain MR volumes. We use IBSR [32][33]. This dataset was used in many other works related to segmentation of brain tissues – for instance in [34], [28], [27]. There are two versions of IBSR dataset. We use standard IBSR dataset which contains volumes from 18 different examinations. Each examination contains six volumes (NN means number of examination). All data have the same resolution, 256x256x128.

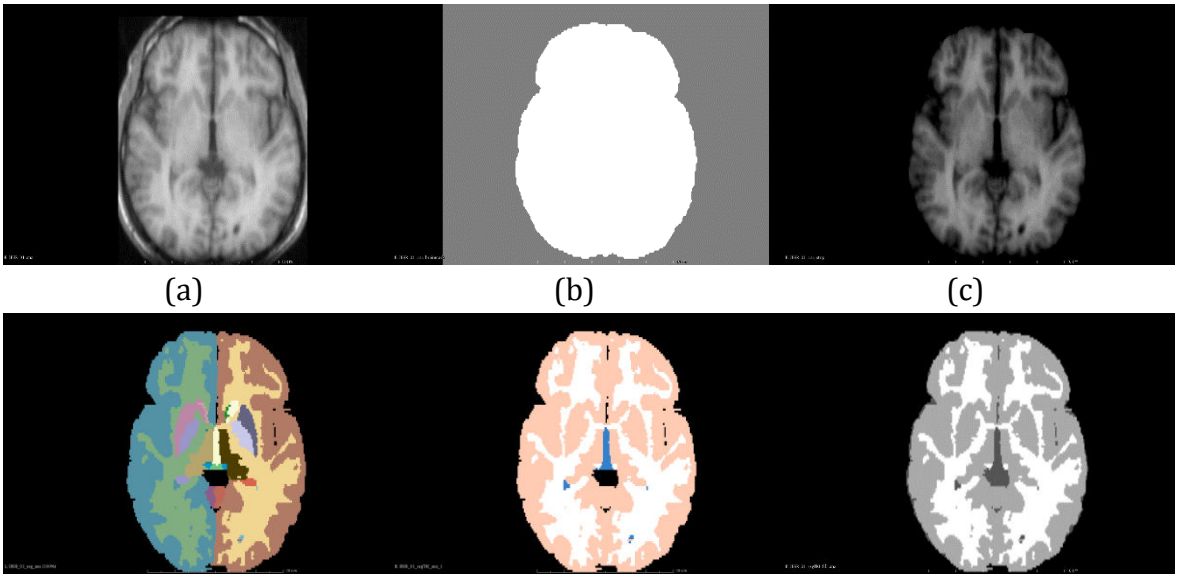
Table 1 contains information about every volume contained in single examination record. For every volume we also show an example picture (pictures were captured in 3D Slicer[4]).

Table 1 - Description of volumes contained in each examination of IBSR.

IBSR_NN_ana.nii **(**Figure 27 a**)**	• Data format: 16bit unsigned integer • Contains raw voxel intensities of whole volume
IBSR_NN_ana_brainmask.nii **(**Figure 27 b**)**	• Data format: 16 bit unsigned integer • Contains mask for segmentation of whole brain from raw data. Voxels have value 1 for brain and 0 for background
IBSR_NN_ana_strip.nii **(**Figure 27 c**)**	• Data format: 16bit unsigned integer • Contains stripped brain data. Value of all background voxels not belonging to WM, GM or CSF was set to 0.
IBSR_NN_seg_ana.nii **(**Figure 27 d**)**	• Data format: 16bit unsigned integer

[4] 3D Slicer homepage: http://www.slicer.org/

	• Contains label for every voxel in IBSR_NN_ana volume. It is the ground truth segmentation of volume. This is a detailed segmentation that segments brain tissues into tens of different tissues.
IBSR_NN_segTRI_ana.nii (Figure 27 e**)**	• Data format: 16bit unsigned integer • Contains ground truth segmentation of IBSR_NN_ana volume. This segmentation segments volume more coarsely than IBSR_NN_seg_ana, concretely to {GM, WM, CSF, BG}
IBSR_NN_segTRI_fill_ana.nii (Figure 27 f**)**	• Data format: float • Contains ground truth segmentation of IBSR_NN_ana volume. This segmentation segments volume more coarsely than IBSR_NN_seg_ana, concretely to {GM, WM, CSF, BG}



(a) (b) (c)

(d) (e) (f)

Figure 27 - Volumes for one subject in IBSR18.

In our method we decided not to segment brain into tens of small structures. We segment brain MR into four classes: {GM, WM, CSF, BG} and therefore we only use IBSR_NN_ANA, IBSR_NN_ana_strip and IBSR_NN_segTRI_ana.

5.2 Method overview

In this work we propose a method for organ segmentation from medical images. We focus on segmentation of brain tissue from MR image stack and its classification into four classes: {WM, GM, CSF, BG}.

Because our method mostly relies on classification, it is divided into two main phases – *Training phase* and *Classification phase*. Both phases start with the same five steps: *Data loading and conversion*, Preprocessing, *Oversegmentation*, *Identification of neighbourhoods* and *Features extraction*. Then Training phase continues with *Training* and Classification phase continues with *Classification*. Both phases and steps are described in later parts of this work. Sequence of steps can be seen in Figure 28.

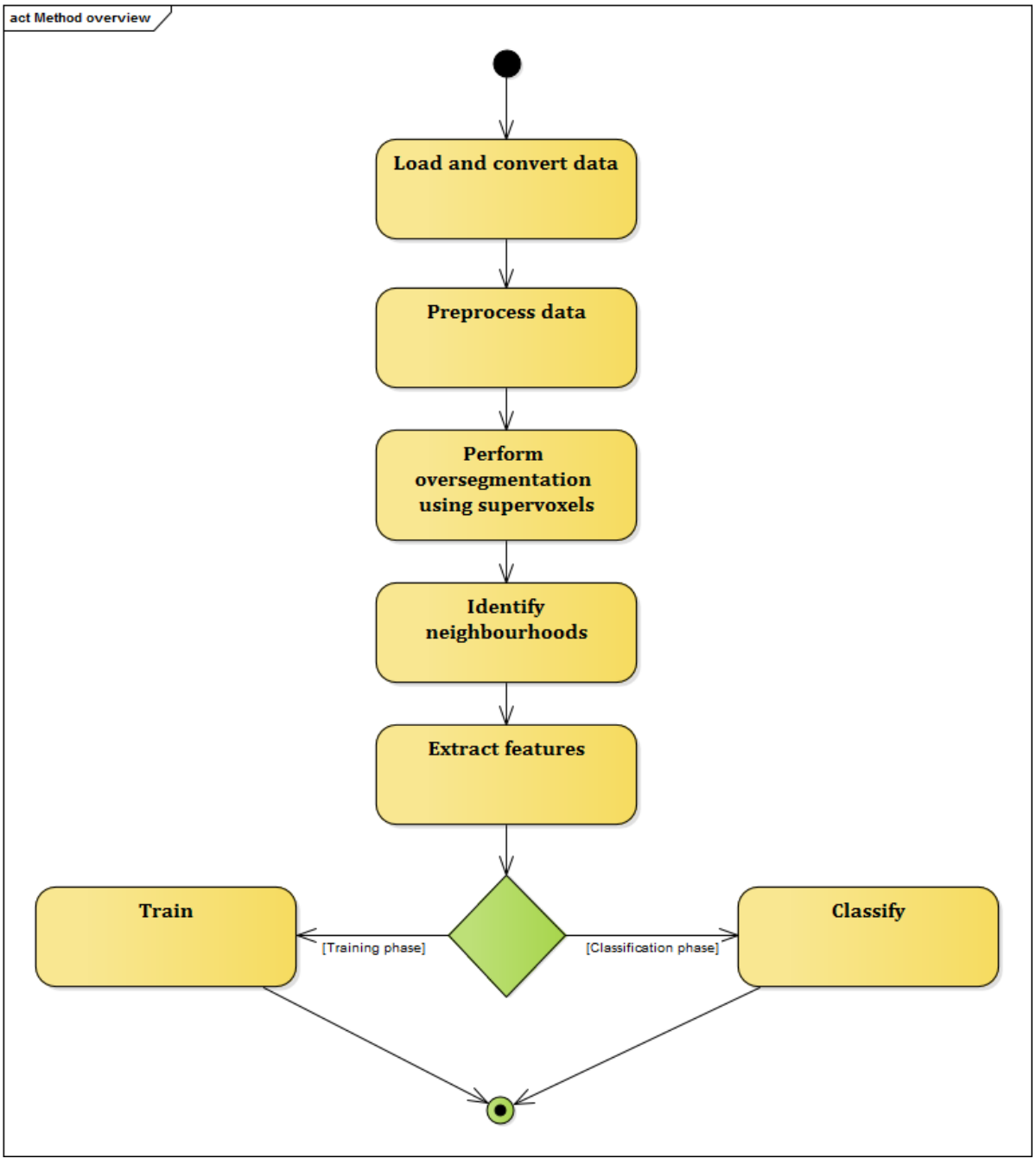


Figure 28 – Proposed method overview (activity diagram).

5.3 Training phase

Because our method is based on classification, we had to split data into train and test set. We did this in two different ways – first, we used volumes 3 to 15 for training the

others for test. Second, we split data on the level of supervoxels, where we used 80 percent of supervoxels for training and the rest for test.

5.3.1 Step 1 – Data loading and conversion

In 5.1 we mentioned four data formats used in medical imaging. In this work we use Nifti for its simplicity and ability to store valuable information about data. This format stores data as raw voxels in a specific data type. As there are various tools and libraries for loading Nifti data we did not implement our own.

Medical images can be stored in various orientations (as described in 5.1). IBSR contains volumes in different orientations. In order to process all volumes in unified way we need to transform all data to the same orientation. **We chose RSA orientation of axes**, because in such case all voxels with the same third coordinate lie in the same slice and were physically taken by MR in the same time.

We also find RSA orientation more intuitive. One of the most important characteristics of MR device is its spatial resolution. To date, the most frequently used resolutions are 256x256, 512x512 and 1024x1024. If we use RSA then we can imagine first two coordinates of every voxel as usual "spatial coordinates" of an image with certain resolution (e.g. 256x256) and the third coordinate represents the number of an image (or slice) in an image stack. Of course, it would be possible make the same assumption with an image in another orientation (e.g. RAS). It is more a matter of taste and it does not influence results of our method. On the other hand it is important to work with MR images with the same orientation.

5.3.2 Step 2 – Preprocessing

There are two main purposes of preprocessing in our method – *Intensity normalization* and *Non-brain tissue removal*.

5.3.2.1 Intensity normalization

As volumes in IBSR-18 vary in dynamic range, all data are normalized into interval [0,1] using quantile normalization. In order to avoid the usage of noisy values in normalization process, we consider all values above $Q_{0.99999}$ equal 1.

5.3.2.2 Non-brain tissue removal

Second, it would be good to remove some non-brain tissue before the actual processing starts. By removing non-brain tissue from volume we want to supress erroneous classification of background supervoxels to brain tissue classes. There are *two main requirements* that we lay on this step:

1. The method must remove significant number of non-brain voxels from processed volume.
2. The method must not remove more than 0.25% of voxels belonging to brain.

Under non-brain voxels we understand voxels that, according to ground truth segmentation, do not belong to brain tissue. We do not remove non-brain voxels by their physical deletion from volume, but by setting their value to zero (which is in MR intensity mostly assigned to air voxels).

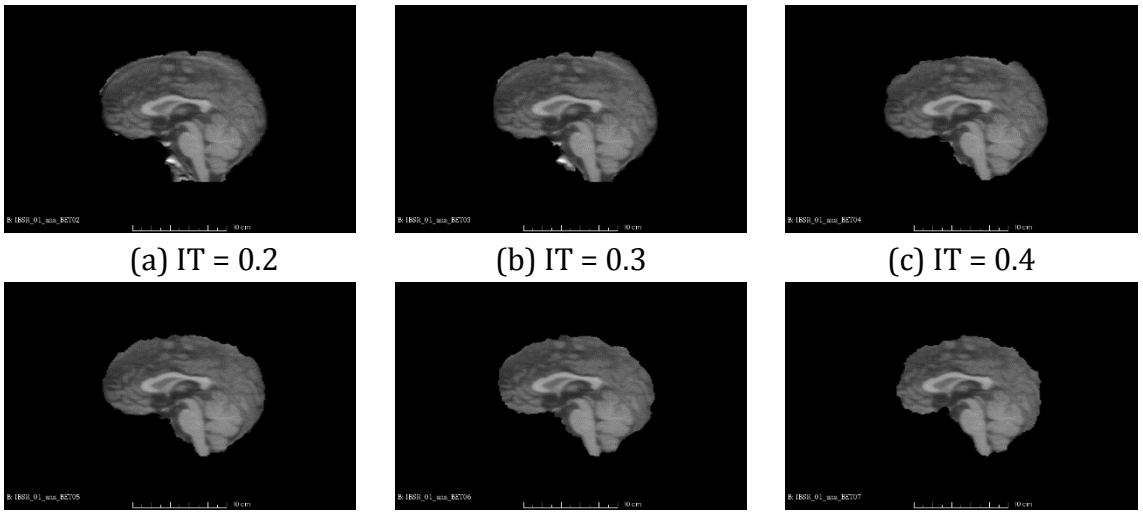
Requirement 1 states that as many non-brain-non-zero voxels as possible should be removed. Success rate of this step is expressed by equation (12) which expresses how many percent of non-brain voxels were preserved.

$$success = 1 - \frac{|NONBRAIN - REMOVED|}{|NONBRAIN|} \tag{12}$$

Theoretically if we removed every voxel from volume, we would maximize the equation (12), which would obviously lead to volume consisting only from one segment – every voxel would have value equal to 0. Therefore we set the second requirement, which satisfaction guarantees that after this preprocessing step almost all brain voxels stay in the preprocessed volume. This requirement is expressed by inequality (13).

$$0.0075 \leq \frac{|REMOVED \cap BRAIN|}{|BRAIN|} \tag{13}$$

Authors in [35] proposed a method for automatic segmentation of MR volumes into brain and non-brain. This method, Brain extraction tool (BET) was used by authors in [34] as a preprocessing step to strip skull from the volume. We also decided to use the same method to remove as much non-brain voxel as possible.



(a) IT = 0.2 (b) IT = 0.3 (c) IT = 0.4

(d) IT = 0.5 (e) IT = 0.6 (f) IT = 0.7

Figure 29 - BET applied on the same subject using different Intensity Threshold (IT) values.

From the number of implementations of BET we use a BET plugin[5] in Multi-image Analysis GUI application (Mango)[6]. This implementation of BET allows to apply segmentation, overlay brain surface, generate brain mask and generate skull image. In this work use BET to remove as many non-brain voxels as possible from segmented volumes. BET allows to set two main parameters – Intensity threshold and Threshold gradient. Our main goal was to choose a combination of values of parameters that would satisfy requirements expressed by equations (12) and (13). We did not change the default value of Threshold gradient as we found this parameter too sensitive (based on visual observation). On the other hand we applied BET with five different values of Intensity threshold. The influence of Intensity threshold value was observer on all volumes from IBSR dataset. In Table 2 we sum up our observations.

Table 2 - Influence of parameters on BET performance (observer on first five IBSR volumes).

Intensity threshold	Threshold gradient	Average remaining non-brain voxels (%)	Average missing brain voxels (%)
0.3	0	2.4595	0.5254
0.4	0	1.4548	1.5555
0.5	0	0.5473	3.9954
0.6	0	0.2109	7.9358

It is clear that with growing value of Intensity threshold also grows the number of brain voxels removed from the volume, which lowers the success rate in context of equation (13). This can be also seen in Figure 29. Our priority is to keep the number of removed brain voxels as small as possible as the removed brain voxels will also be missed in the final segmentation and therefore the overall success rate will be decreased. In Figure 30 we can see that value 0.3 for Intensity threshold satisfies requirement 2 for all five volumes and about half of volumes satisfy this requirement for value 0.4. Figure 31 shows average values of missing brain voxels in volume after BET has been applied.

[5] BET plugin Home page: http://rii.uthscsa.edu/mango/plugin_jbet.html
[6] Mango Home page: http://rii.uthscsa.edu/mango/

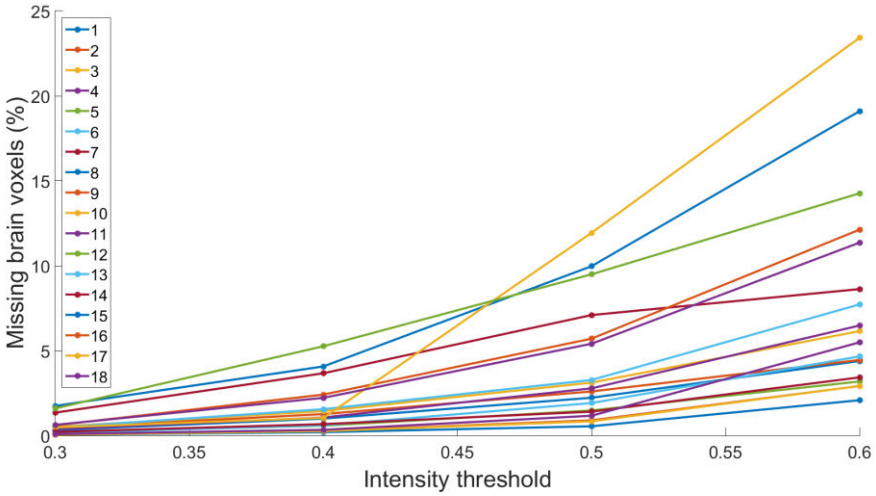


Figure 30 - Relation between Intensity threshold (BET parameter) and number of removed brain voxels in individual subjects of IBSR dataset.

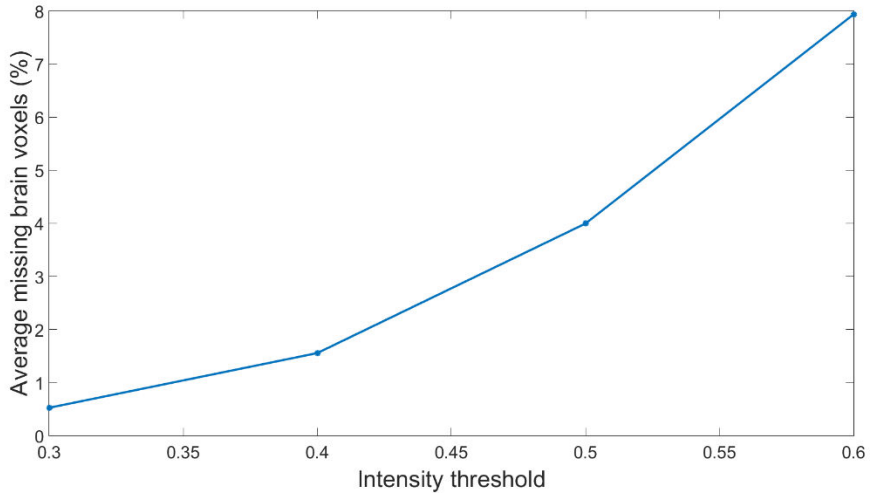


Figure 31 - Relation between Intensity threshold and average number of removed brain voxels.

On the other hand with growing Intensity threshold falls the number of non-brain voxels remaining in the volume after BET has been applied. We measured ratio between remaining non-brain voxels and all non-brain voxels in volume for the first five volumes in IBSR dataset (Figure 32). We also calculated the mean values among all five volumes (Figure 33).

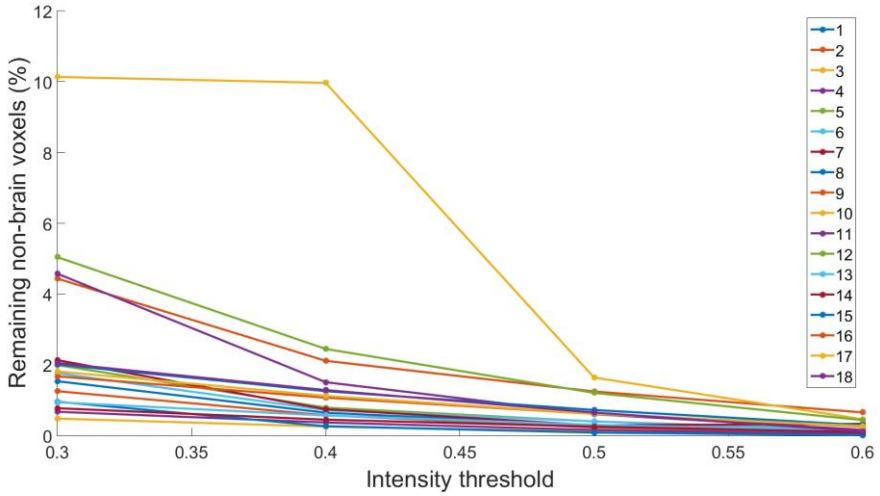


Figure 32 - Relation between Intensity threshold (BET parameter) and number of remaining non-brain voxels in individual subjects of IBSR dataset.

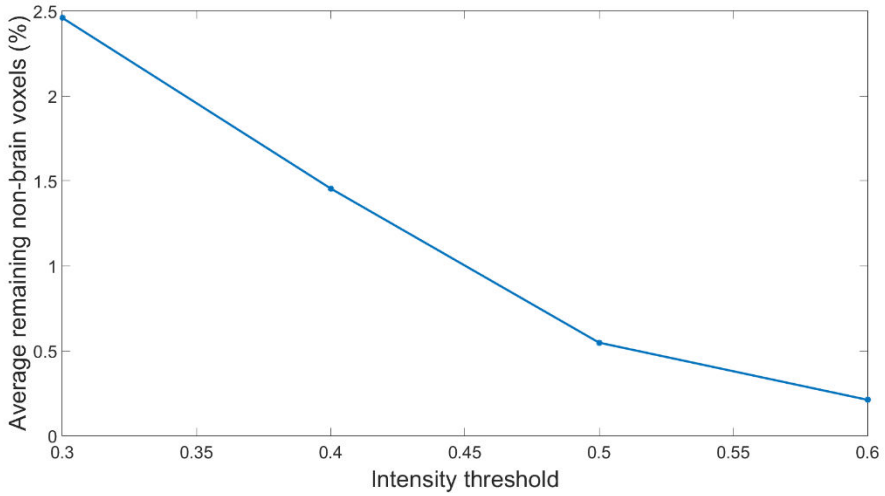


Figure 33 - Relation between Intensity threshold and average number of remaining non- brain voxels.

It is clear that with increasing Intensity threshold grows success rate of requirement 1 and falls success rate of requirement 2. Our goal is to balance these two and get an optimal value for this parameter of BET. Equation (13) says that no more than 0.25% of brain voxels may be removed. In Figure 30 and Table 2 we can see that the ideal value lies between 0.3 and 0.4. As the relation between Intensity threshold and number of removed brain voxels is not linear, we assumed that the ideal value closes to 0.30. We evaluated the value 0.30 for all examinations from IBSR dataset. In Figure 34 we can see that in major cases is the requirement 2 fulfilled. There are two subjects where number of missing brain voxels was close to maximal desired value (0.75%). Concretely in examination 7 was this value equal to 1.35%, in 8 it was 1.75% and in 12 it was 1.60%. In these cases we accept this value, because if we lowered value of Intensity threshold even more, success rate of requirement 1 would dramatically decrease (in examination 10 even the value 0.3 caused that number of remaining non-brain voxels exceeded 10% as it can be seen in Figure 35). **In order to balance requirements 1 and 2 we accept 0.3 as a value of choice for Intensity threshold**.

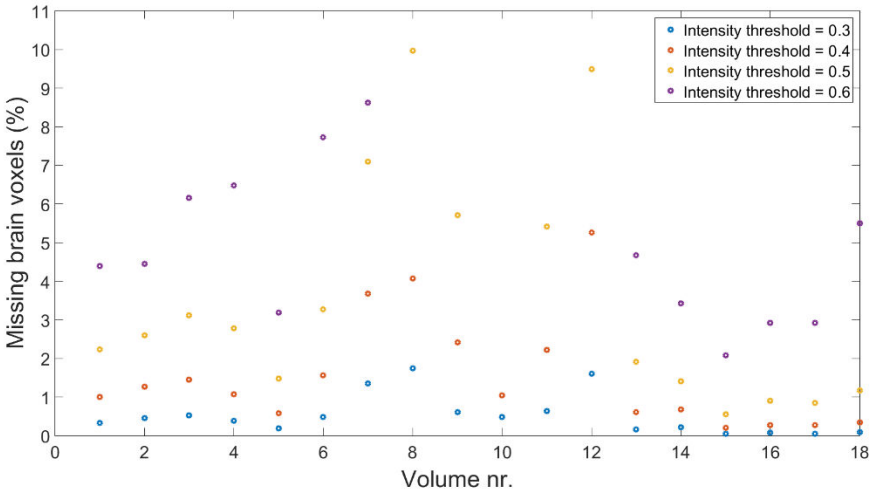


Figure 34 - Number of missing brain voxels in all examinations for all subjects and different Intensity Thresholds.

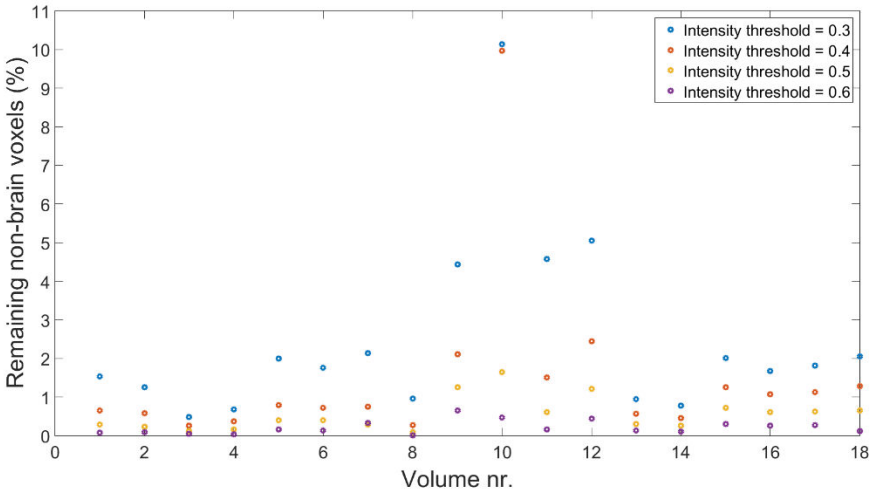


Figure 35 - Number of remaining non-brain voxels in all examinations for all subjects and different Intensity Thresholds.

5.3.3 Step 3 – Oversegmentation using supervoxels

To oversegment image into supervoxels, we use SLIC algorithm proposed in [22] and compared to other state-of-the-art superpixel and supervoxel algorithms in [8]. In Berkley dataset, commonly used to compare effectiveness and efficiency of superpixel algorithms, SLIC proved its qualities as the algorithm with best quality of results / computation complexity ratio. Method proposed in [19] gave accurate and visually pleasing supervoxels, but at cost of extremely high computational complexity and time. Besides, [19] implementation does not allow to oversegment 3D volumes.

Decrease of computational complexity is not the only benefit provided by supervoxels. Sometimes CT and MR images suffer from high level of noise. Supervoxels, represented by statistical data of included voxels, in its nature resist to noise better than single voxels. In [13] authors also use neighbouring pixels to describe single pixel, but they

do it in 2-dimensional space. Our approach allows to model supervoxel in context of 3D neighbourhoods.

Ideally, a supervoxel should contain only voxels belonging to the same tissue. For many reasons this is often not possible, because in special cases even a single voxel contains information belonging to more tissues (partial volume effect). Success rate of supervoxelization in terms of homogeneity of individual supervoxels can be expressed by following equation:

$$success = \frac{1}{N} \sum_{s \in Supervoxels} \frac{major(s)}{size(s)} \tag{14}$$

Meaning of individual symbols in equation (14) is following:
N – Number of supervoxels
s – Single supervoxel
size(s) – Number of voxels in a supervoxel s
major(s) – Number of voxels that belong to the most occurring class in a supervoxel

We also evaluated equation (14) for individual classes, because the number of supervoxels belonging to background was much higher than the number of supervoxels belonging to other classes (WM, GM, CSF). SLIC allows to set desired size and compactness of supervoxels. Figure 36 shows results of the evaluation as a function of supervoxel compactness at fixed supervoxel size and Figure 37 shows results as a function of supervoxel size at fixed supervoxel compactness.

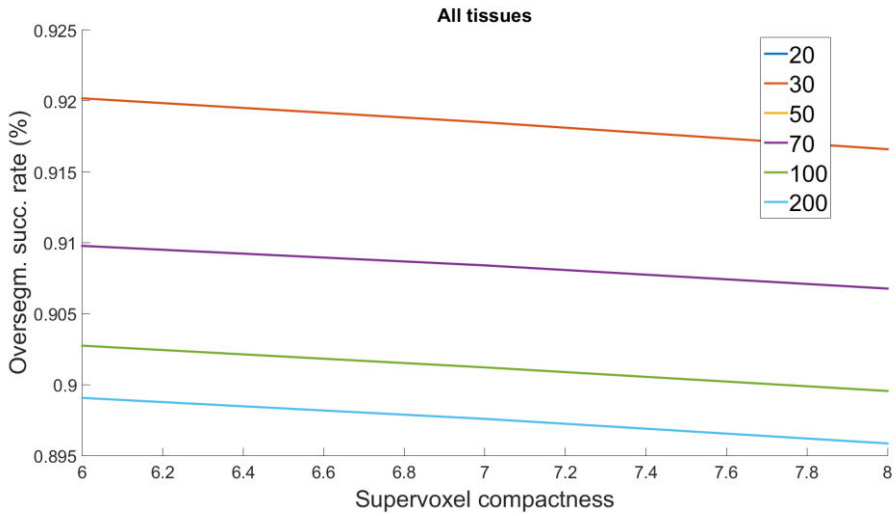


Figure 36 - Success rate of oversegmentation as a function of supervoxel compactness at fixed size.

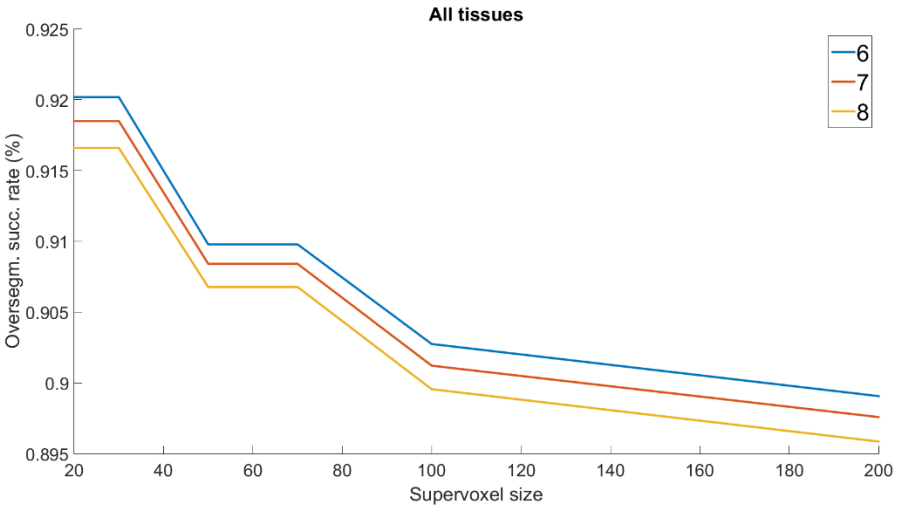


Figure 37 - Success rate of oversegmentation as a function of supervoxel size at fixed compactness.

In Figure 36 and Figure 37 we can see that increase of both compactness and supervoxel size leads to decrease of oversegmentation success rate. For compactness equal to 6 at fixed supervoxel size success rate defined in equation (14) reaches its maximum. As we are not interested in supervoxel regularity compactness, we accepted this value and used it in oversegmentation procedure.

More difficult was to define the right value of supervoxel size. We had to find an equilibrium between success rate and the amount of statistical information contained by supervoxel. We assume that the more voxels are contained in a supervoxel the more information can be retrieved (such as histogram of intensities). Although very small supervoxels maximize equation (14), we decided to use supervoxel size 120 in order to balance oversegmentation success rate and amount of information in supervoxel.

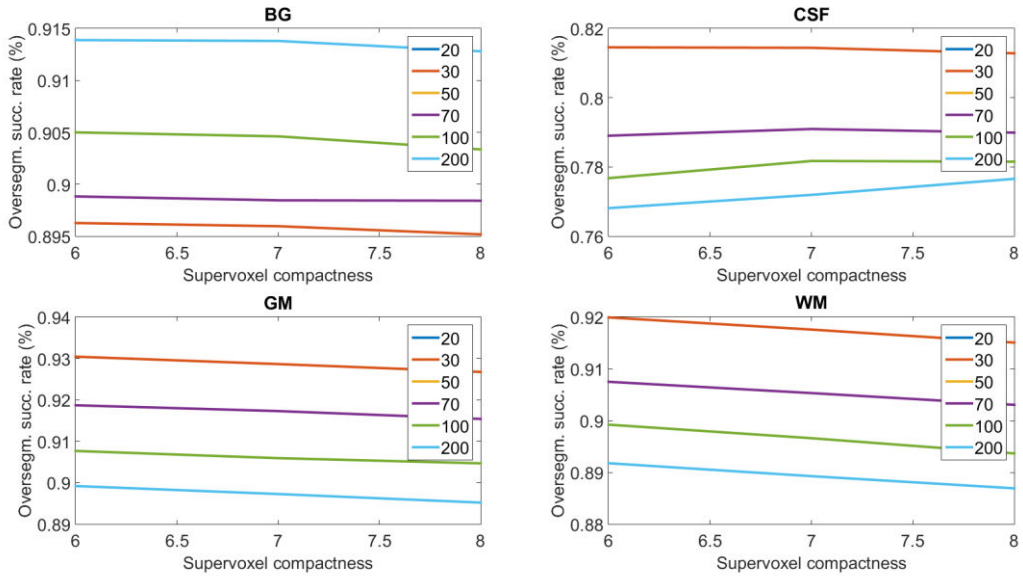


Figure 38 - Success rate of oversegmentation as a function of supervoxel compactness at fixed size among all classes.

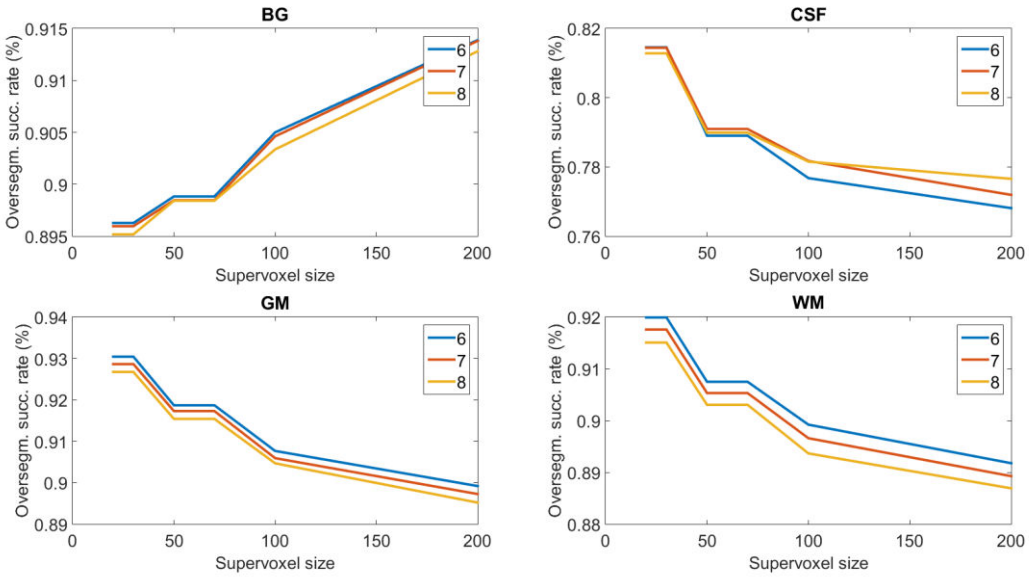


Figure 39 - Success rate of oversegmentation as a function of supervoxel size at fixed compactness among all classes.

Among classes we observed big inter-class variance in context of influence of compactness and supervoxels size on success rate (Figure 38 and Figure 39). We found out that supervoxels with major class equal to GM and WM (classes 2 and 3) reacted on change in compactness and supervoxel size similarly to average. On the other hand BG reacted inversely and CSF was the most unpredictable. **Values 6 for compactness and 120 for supervoxel size are a compromise among all four classes**.

5.3.4 Step 4 – Identification of neighbourhoods

As our supervoxels do not have regular shape, it is not possible to hold them in a regular structure such as rectangular multidimensional field. Therefore it is not straightforward to identify neighbours of every supervoxel.

On the other hand individual voxels (if we abstract from supervoxels) form a rectangular three-dimensional array. We use this structure in this phase to identify neighbourhoods between supervoxels. Let's call this structure *voxel_array*. Voxels with the same third dimension are considered to form a single slice.

First of all we need to define "Supervoxel neighbourhood" between two supervoxels. In our method **supervoxel S1 and S2 is considered a neighbour of S2 if at least one voxel from S1 is in 8-neighbourhood with at least one voxel from S2 in a single slice of *voxel_array***. Although voxels in a single supervoxel are unordered, it is possible to address corresponding voxel in *voxel_array*, because every voxel in supervoxel knows its three-dimensional coordinates.

5.3.5 Step 5 – Features extraction

Compared to individual voxels is information contained in supervoxel much more descriptive. As supervoxel consists of many voxels (in our case more than hundred) the information it contains is an aggregation of information caught by contained voxels.

Apart from that supervoxel makes it possible to study its structure and relations between individual voxels.

Although voxel can be characterized by its neighbouring voxels, the information contained in its neighbours is rather limited. In case of supervoxel we can characterize its neighbours in much more descriptive way. For example it neither makes sense to describe single voxel with an intensity histogram nor is there a possibility to calculate histogram for its nearest neighbours (it would be possible with sufficiently big kernel).

We describe supervoxel with following features:

- **Normalized intensity histogram of voxels in supervoxel (24 bins).**
- **Normalized intensity histogram of all voxels in neighbouring supervoxels (24 bins).**
- **Normalized Euclidean distance of supervoxel centroid from the centre of the brain.**
- **Angle between supervoxel centroid and brain centre in XY, XZ and YZ plane.**

Authors in [27] used intensity histogram in combination with features that described texture and shape of a supervoxel, creating a feature vector with length 228.

Normalized intensity histogram of voxels in supervoxel is the main characteristic of supervoxel. Although intensity distribution in individual classes is not normal, there are intensity values that are more characteristic for specific classes than for the others. Intensity histogram in Figure 40 shows the distributions of individual classes. It is clear that WM contains the voxels with the highest intensity, whereby background contains the darkest voxels. This fact allows us to assume that it is possible to classify supervoxels by their intensity distribution. For readability reasons we ignored voxels with intensity equal to 0 as they form the majority of all voxels.

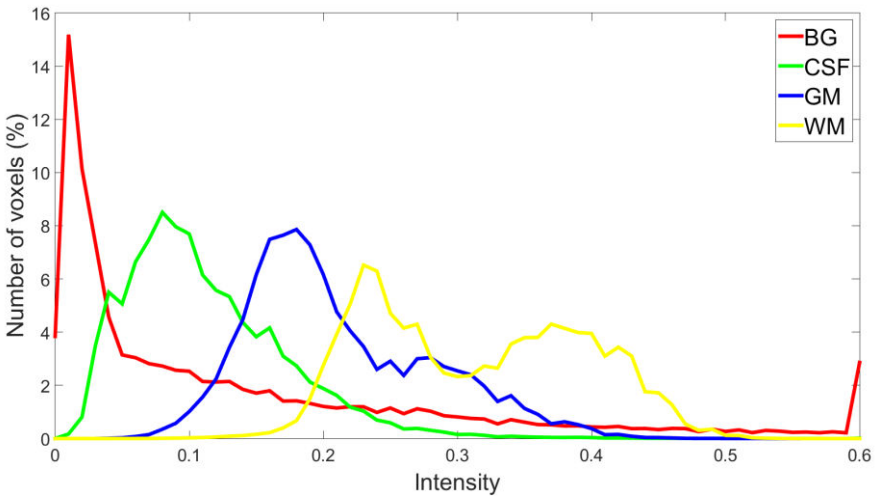


Figure 40 - Intensity histograms of individual classes in supervoxel (volume from IBSR taken by T1 MR).

Normalized intensity histogram of all voxels in neighbouring supervoxels should help to classify supervoxels according to their neighbours. If a supervoxel is surrounded by supervoxels with intensity histograms typical for some class (e.g. GM) it is a good chance that the supervoxel will also belong to the same class. On the other hand if intensity histogram of neighbours contains intensities from all classes it is clear that the classified supervoxel lies between multiple classes, at the boundary of some tissue.

The last feature, **normalized Euclidean distance of supervoxel centroid from the centre of the brain,** is based on morphology of brain. Outer boundary of GM and CSF is typically the most distant from the centre of brain. On the other hand some structures belonging to CSF are closest to centre. It would not make sense to extract absolute distance calculated from three-dimensional coordinates of brain centre and supervoxel centroid as the resolution of MR image can change and also physical thickness of one slice can differ. In IBSR dataset one slice has a thickness of 1.5mm. Therefore we calculate normalized Euclidean distance of supervoxel centroid as following:

$$dist = \frac{\sqrt{(x_S - x_B)^2 + (y_S - y_b)^2 + (z_S - z_B)^2}}{norm} \tag{15}$$

norm denotes normalization factor and equals to:

$$norm = \sqrt{\left(\frac{\max(x)}{2}\right)^2 + \left(\frac{\max(y)}{2}\right)^2 + \left(\frac{\max(z)}{2}\right)^2} \tag{16}$$

x_S, y_S and z_S are coordinates of supervoxel, x_b, y_b and z_b are coordinates of brain centre, norm is normalization term and *max(x/y/z)* is maximal coordinate in MR image in particular direction.

Distribution of distances from brain centre for individual tissues can be seen in Figure 41. Many supervoxels belonging to different classes have similar distance from brain centre. Therefore, we added **three angles between the centroid of supervoxel and brain centre (in XY, XZ and YZ plane)**. In combination with the distance from brain centre, position of supervoxel is described much more precisely and uniquely.

Our method was meant to work with minimal user's interaction. Therefore we had to find some method to estimate the centre of the brain. We claimed that the centre of the brain is the centroid calculated from all non-zero voxels that remained after application of BET in preprocessing

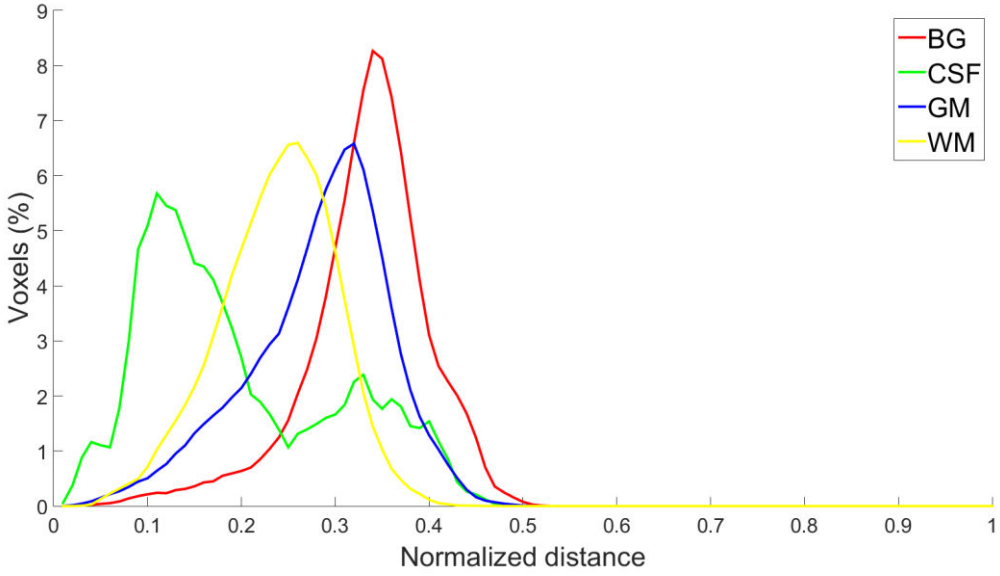


Figure 41 - Distribution of normalized distances from brain centre for individual tissues.

5.3.6 Step 6 – Training

Qualities of supervoxels allow to easily extract wide variety of features for all of them. This gives us an opportunity to train a classifier to classify every voxel to specific tissue.

Multi-layer perceptron (MLP)

Each supervoxel is assigned to either BG, CSF, GM or WM. In training phase, we train multilayer perceptron (MLP) with two hidden layers having 52 and 8 neurons, sigmoidal activation function and Levenberg–Marquardt training function. A significant number of supervoxels consists of voxels from even more than two different classes. In training process, we do not include supervoxels having less than 87% voxels from single class and in overall classification we do not include supervoxels having mean intensity equal less than 2 (a priori background).

One of the advantages of MLP is that every voxel can excite multiple output neurons at different level of excitation. This gives us an opportunity to specify threshold of certainty. If the level of excitation is higher than given threshold we find classification of specific supervoxel successful, otherwise we can try to apply some post-classification technique to given supervoxels. In fact, this is proposed as the core of our future work. As we can see in Figure 40 borders between intensities of different tissues is not absolutely clear. This uncertainty is mainly caused by partial volume effect. Especially in these unclear areas we expect noticeable excitation of multiple output neurons.

5.4 Classification phase

5.4.1 Step 1 – Step 5

These steps are equivalent to steps 1 to 5 performed in training phase.

5.4.2 Step 6 – Classification

Classification of brain tissues is not a trivial task. Current state-of-the-art method claim performance in terms of Dice similarity coefficient 0.69±0.03 for CSF, 0.86±0.01 for GM and 0.89±0.02 for WM [27]. In our opinion this rather high variance in segmentation performance is caused by heterogeneity of different brain tissues. Mostly intensities of BG are spread almost all over the intensity spectrum and intersects with intensity intervals of all other tissues (it can be seen in Figure 40). Our goal is to reach the highest possible segmentation performance with lowest possible computational complexity.

The first step in classification is the use of prior knowledge – supervoxels having mean intensity less or equal to 2 are considered background.

In the second step we classified individual supervoxels. Each supervoxel was classified as either background (BG), cerebrospinal fluid (CSF), grey matter (GM) or white matter (WM). Afterwards all the voxels in a particular supervoxels are assigned to the same class. Here is where the oversegmentation error becomes obvious – if a supervoxel contains voxels from more than one class, some voxels will be necessarily assigned to incorrect class.

6 Results

In this part we evaluate performance of our method in segmentation of human brain tissues. We compare our results with current state-of-the-art method that uses similar approach based on supervoxels and that claims to reach better results than other methods [27].

We trained MLP with two hidden layers having 52 and 8 neurons. We used Levenberg-Marquardt training function, sigmoidal activation function.

For training, classification and evaluation purposes, we took supervoxels from all subjects in IBSR-18 and split them to training set and testing set (80:20). Proposed method is compared with the current state-of-the-art method SITDS [27] using the same evaluation metric – DSC, which stands for Dice similarity coefficient.

The authors compared SITDS using the IBSR-18 with other state-of-the-art methods and reported the best results. We thoroughly evaluated performance of proposed method for every tissue individually Table 3 and conclude that our results are clearly comparable to those of current state-of-the-art methods.

Table 3 - Performance comparison of proposed method and current state-of-the-art method SITDS in terms of Dice similarity coefficient (DSC).

	Cerebrospinal fluid	**Grey matter**	**White matter**
Proposed method (DSC)	0.67	0.86	0.85
SITDS (DSC)	0.67	0.86	0.89

Performance of proposed method can be increased even more. In classification evaluation we observed that misclassified supervoxels tend to have bigger standard deviation, lower percentage of major class voxels and lower MLP excitation rate (Table 4). Therefore, we can identify missclassified supervoxels, split them and classify individually. We assume that such supervoxels will be more homogeneous.

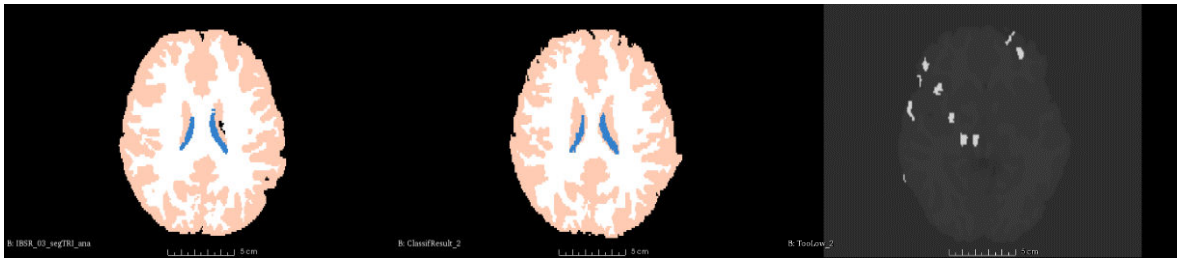
Table 4 - Correctly classified supervoxels have greater MLP excitation and major class percentage. Contrary, intensity standard deviation is lower among them.

	Intensity σ	**Major class perc.**	**MLP excit. rate**
Correctly classified	8.1692	92.67%	0.9883
Misclassified	10.0361	72.5%	0.8964

As far as computational complexity is concerned, the most demanding step is training. However, does not degrade our method, as training is only performed once. Other demanding steps are oversegmentation, creation of volume representation and features extraction. Last two steps can be, however, optimized, as they are currently

implemented in C# language, which can be outperformed by C++ implementation. We did not evaluate segmentation run time thoroughly, but roughly measured complete segmentation of one subject took less than 40 seconds.

We also trained second MLP using all supervoxels from subjects 4-15 of IBSR-18. Subsequently, we segmented subject 3 from IBSR-18. Results can be seen in Figure 42. As it can be seen, our segmentation was very similar to the ground truth. In the Figure 42 (c) there are highlighted supervoxels where MLP excitation rate did not exceed value 0.9. In most cases these supervoxels were either completely misclassified or contained significant amount of voxels from at least two tissues.



(a) Ground truth (b) Proposed method (c) Low MLP excitation

Figure 42 - Segmentation result. (c) highlights supervoxels that excited MLP in rate lower than 0.9.

7 Technical realization

In this chapter, we describe implementation details of our application. In this project our aim was to implement proposed method and all tools needed to verify its performance, robustness, stability and compare it with other state-of-the-art segmentation methods.

Our goal was not to create advanced GUI concentrated on user experience. In this way we followed minimalistic approach. On the other hand, we paid attention in design phase of main and support libraries, so they are easy to extend and maintain, using software engineering skills and best practices including architectural and design patterns.

7.1 Software means

As the main language to use C++/CLI. In the computer vision domain, C++ is together with C the main language in which the algorithms and libraries are developed. The reason why this is true is that many algorithms are computationally complex and in order to be fast and efficient they need to be implemented in a low level language. Besides we did not want to spend a lot of time maintaining memory allocations and deallocations. Therefore we did not use vanilla C++, but C++/CLI, which supports work both with native and managed resources.

The next reason why we used C++/CLI is that it allows to use the .NET framework and library written in this language can be easily used in C#, which we used for creation of GUI and some other specific tasks including classification.

As far as libraries are concerned, we mostly relied on ITK [36] (computer vision) and Accord.NET [37] (machine learning and statistics).

As platform we use 64-bit version of Microsoft Windows 8.1.

Due to its debugging and other useful tools, we use Microsoft Visual Studio 2013 Ultimate.

Besides C++ and C# we used MATLAB in first stages for prototyping. We also used it for evaluation, statistics, graph generation etc.

7.2 Hardware

Computations and experiments were performed on Asus G750-JZ. Important parameters:

Processor	**Intel core i7 4710HQ, 2.5GHz (3.5GHz in Boost), 4 cores**
Main memory	32GB DDR3
Graphic card	NVIDIA GeForce GTX 880M, 1536 cores

7.3 Design overview

During design phase of development cycle of our software project we proceeded from requirements defined in Figure 52.

Conceptually, our project is composed of multiple modules. Our goals was to keep the modules as independent of each other as possible. For this purpose we have rather extensively used interfaces and abstract classes to that individual volumes do not depend on a concrete implementation of a particular class or method. These modules are *Classification*, *Conversion*, *Loading*, *Processing*, *Serialization*, *Statistics*, *Visualization* and *GUI*. Besides them there are some auxiliary modules – *Utils* and *Helpers*. All the modules can be seen in Figure 53.

Physically, we realised our project as a Visual Studio solution with multiple projects. One of the biggest advantages of Visual Studio solution using .NET framework is that individual modules can be easily spread over many projects. The solution consists of following projects:

- **DP_CLR**
  - C++/CLI .dll library project.
  - The main part of the whole solution. Contains most of the algorithms used by proposed method.
  - Contains definitions of main data structures (Volume, Voxel, Supervoxel).
  - Responsible for data loading, conversion, serialization and processing. Also contains classes responsible for statistics and features extraction.
  - Extensively uses template methods and template classes.
  - As it is possible to use native C++ in this project, it contains classes responsible for usage of ITK and SLIC, which are written in native C++.
  - Directly uses CLICSuperpixels.
- **DP_CS**
  - C# .dll library project.
  - Mainly responsible for features extraction, classification and statistics extraction.
  - Directly uses structures defined in DP_CLR.
- **DP_Visualization**
  - C# .dll library project.
  - Contains implementation of Maximum intensity projection (MIP) visualization of a Nifti volume.
  - Directly uses DP_CLR and DP_CS.
- **DP_WPF_prototype**
  - C# WPF project.
  - Defines graphical user interface and is responsible for interaction with user.

- Through window controllers defines complete training and classification flows which can be adjusted by setting various parameters in GUI.
- Contains GUI for visualization of Nifti volumes.
- Directly uses CP_CLR, CP_CS, DP_visualization.

- **SLICSuperpixels**
  - 3rd party native C++ library.
  - Contains implementation of SLIC algorithm.

Complete overview of the solution projects together with related modules (conceptual level) can be found in Figure 53. The basic overview of dependencies between projects can be seen in Figure 43.

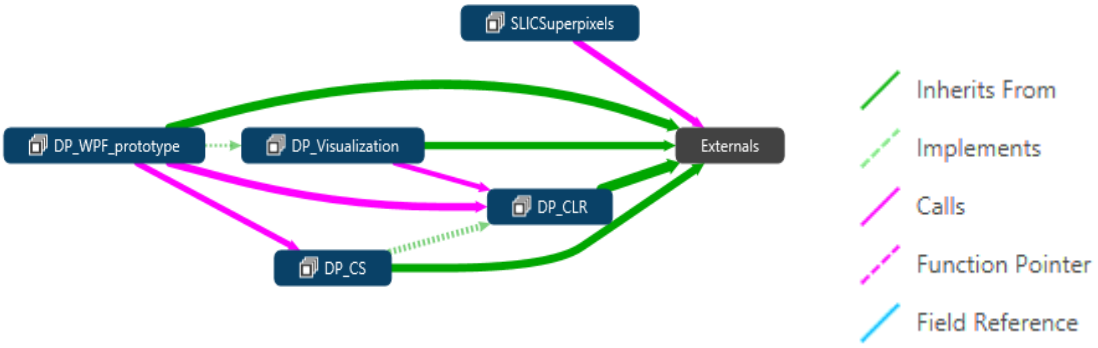


Figure 43 - Basic overview of relations between projects in Visual Studio solution.

7.4 Data loading and conversion

As stated in 5.1, in our method we work with medical data in Nifti format. Nifti format is rich and rather complex (although simpler if compared with DICOM). Therefore we use ITK library to load, process and export data in this format. ITK (like many others C++ libraries) extensively uses concept of templates. For loading ITK requires to define dimension of medical image and data type of individual voxels. We work directly with 3D volume where individual voxels are stored as floats. Because this data type is used throughout whole project we defined it in one place using following code:

```
/// DIMENSIONS
const unsigned int Dimension3D = 3;
/// PIXEL TYPES
Typedef float ITKPixelType;
/// IMAGE TYPES
typedef itk::Image<ITKPixelType, Dimension3D> ITKImageType3D;
```

Our method is based on oversegmentation of medical volumes into supervoxels. To best of our knowledge there is no library or other software mean which would allow to store raw voxels, supervoxels and information about neighbourhoods between

supervoxels and to extract some additional characteristics of that volume. Therefore we designed and implemented own hierarchy of classes and structures with these qualities. It can be seen together with classes responsible for data loading in Figure 57.

In general, data in Nifti format are loaded to a native structure of ITK library. Then, these data are converted to an instance derived from *IVolume* interface using our conversion module (Figure 44). It is also important to note that in this step all the loaded data are transformed to the same orientation RSA.

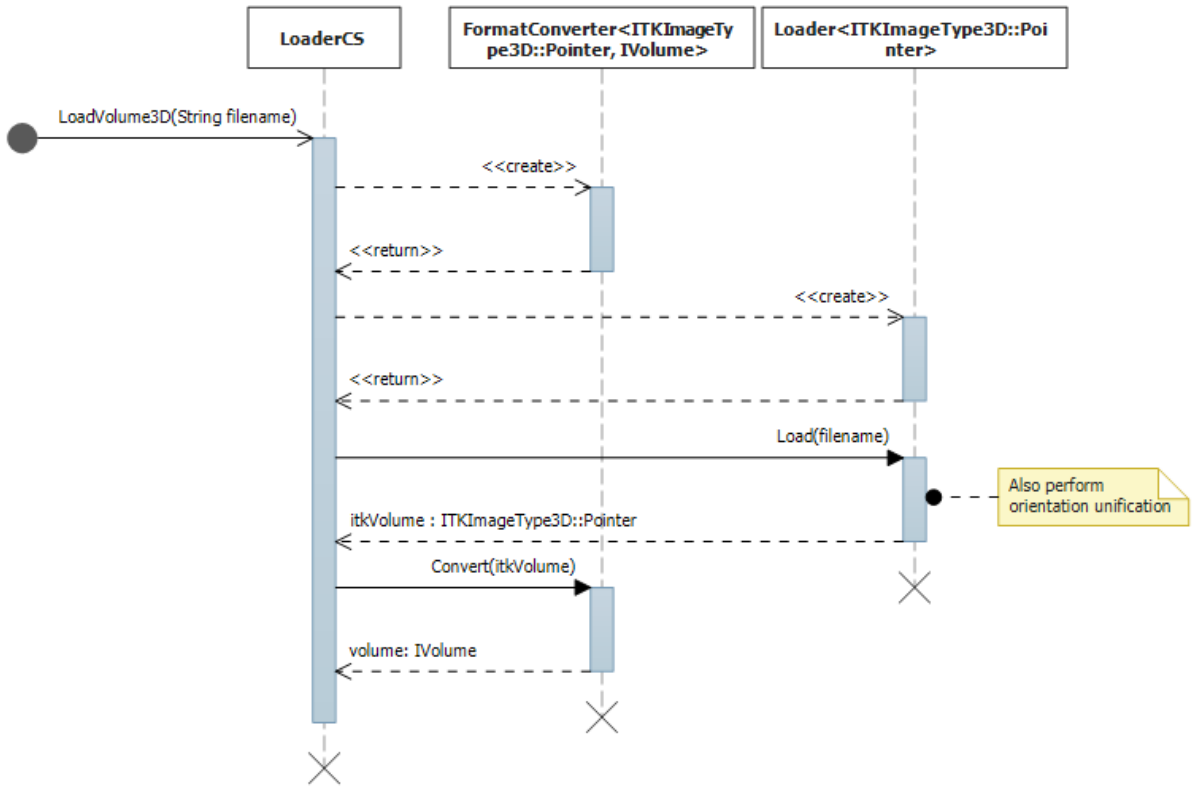


Figure 44 - Nifti volume loading and conversion to internal IVolume format.

7.5 Preprocessing

Except for brain extraction, every other preprocessing is performed in our application. Individual preprocessing procedures can be considered to be *filters*. We do use this term as it is usual in ITK (e.g. *OrientImageFilter*). We put emphasis on consistence of filters. Therefore every single filter used in our application must implement interface *IFilter*. This interface guarantees that every filter can be applied and reverted in the same way. The hierarchy can be seen in Figure 45. It is important to add that some filters, for example *NormalizationFilter*, cannot be reverted algorithmically. Therefore it is necessary to save original intensities of voxels in the filter so it can be reverted.

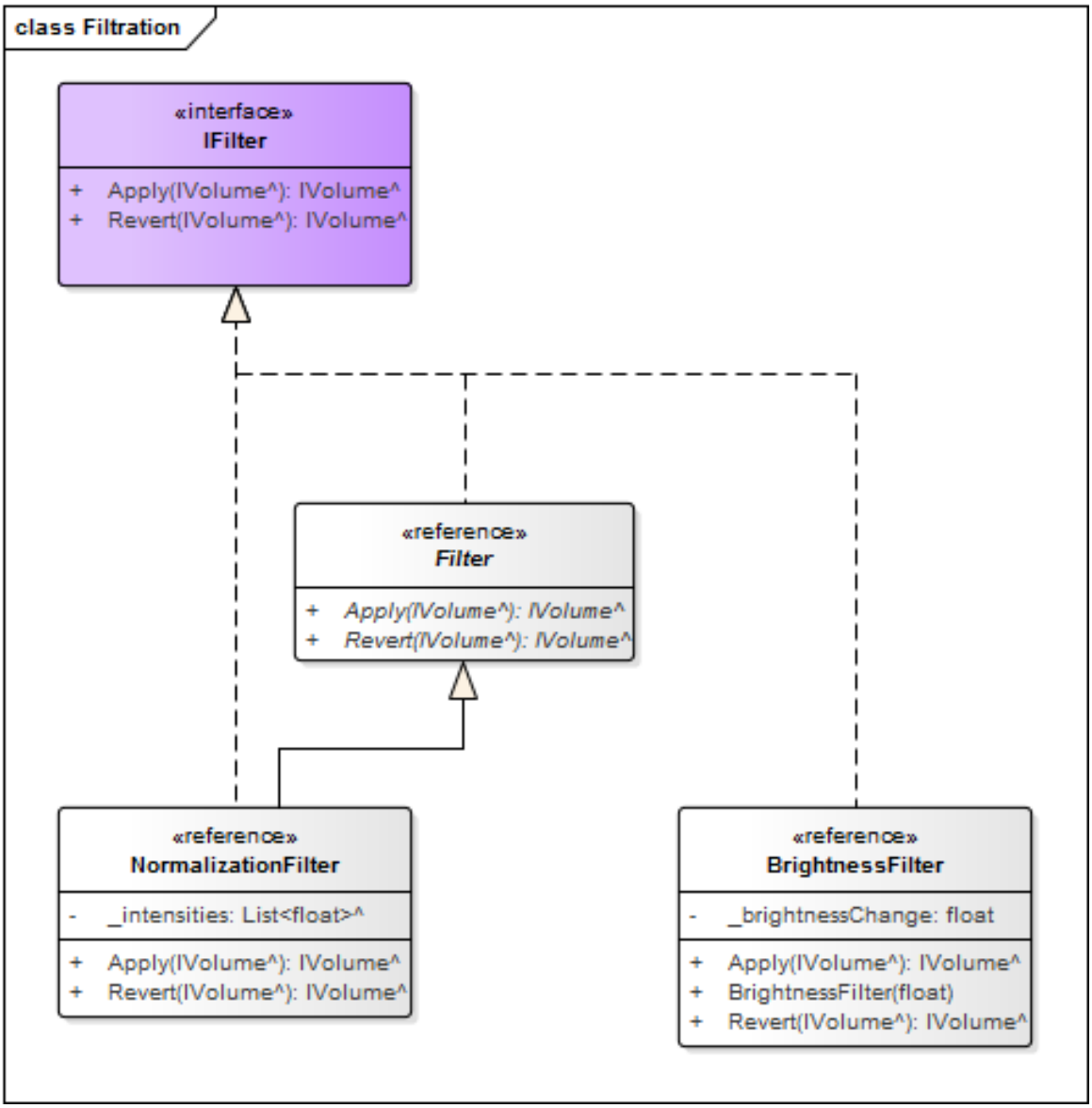


Figure 45 – Hierarchy of interfaces and classes responsible for data filtration (processing).

Source code of Apply and Revert methods of *NormalizationFilter* can be seen in section A5.

7.6 Oversegmentation and supervoxel neighbourhoods

Segmentation together with filtration belongs to the same module – Processing. The main class, which is responsible for oversegmentation of a volume into supervoxels, is *SlicSegmenter*. *SlicSegmenter* is one of a few classes that are partially native C++. In constructor, *SlicSegmenter* requires two parameters to be set – supervoxel size and compactness. It provides one public method – Apply, which requires an instance of *IVolume* as an input. Class diagram of *SlicSegmenter* can is in Figure 46.

SlicSegmenter uses *SLICSuperpixels* native C++ library to get supervoxel label for each voxel in volume. Afterwards instances of class Supervoxel are created and calculated label values are assigned to individual Voxels. Next, created supervoxels are assigned

to *IVolume* instance and neighbourhoods between supervoxels are identified. The overall process can be seen in sequence diagram in Figure 47.

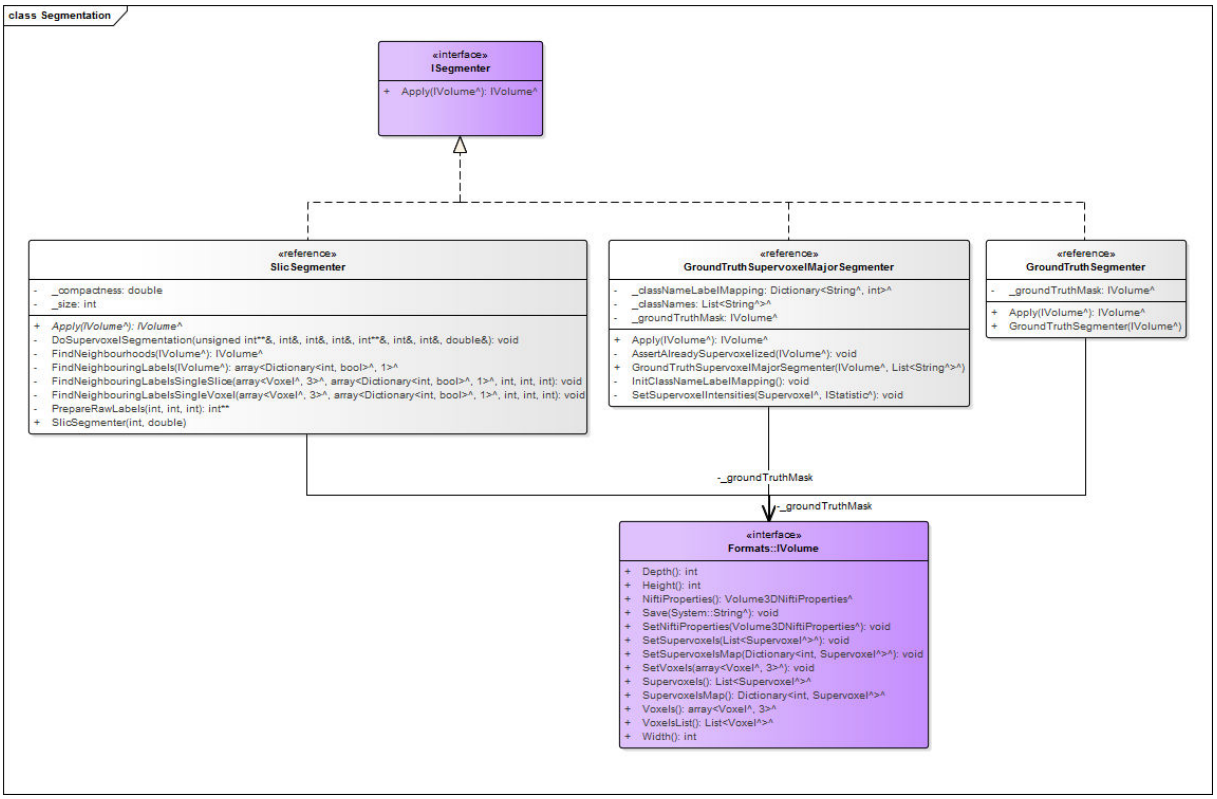


Figure 46 - Hierarchy of interfaces and classes responsible for data segmentation.

Besides oversegmentation into supervoxels, there are two other segmentation classes:

GroundTruthSegmenter

Sets ground truth label of every voxel in a volume according to ground truth mask and returns the volume itself.

```
IVolume^ GroundTruthSegmenter::Apply(IVolume^ volume)
{
	auto voxelsList = volume->VoxelsList();
	auto labelsList = _groundTruthMask->VoxelsList();

	for (int i = 0, numVoxels = voxelsList->Count; i < numVoxels; i++)
		voxelsList[i]->groundTruthLabel = labelsList[i]->value;

	return volume;
}
```

GroundTruthSupervoxelMajorSegmenter

Sets value of every voxel in every supervoxel equal to the most occurring label. Important is that it only makes sense to use this segmenter after supervoxelization, otherwise there were no supervoxels at all in the volume. In fact, it is the main purpose of this segmenter - set all voxels in a supervoxel to the same value and therefore create a segmentation mask.

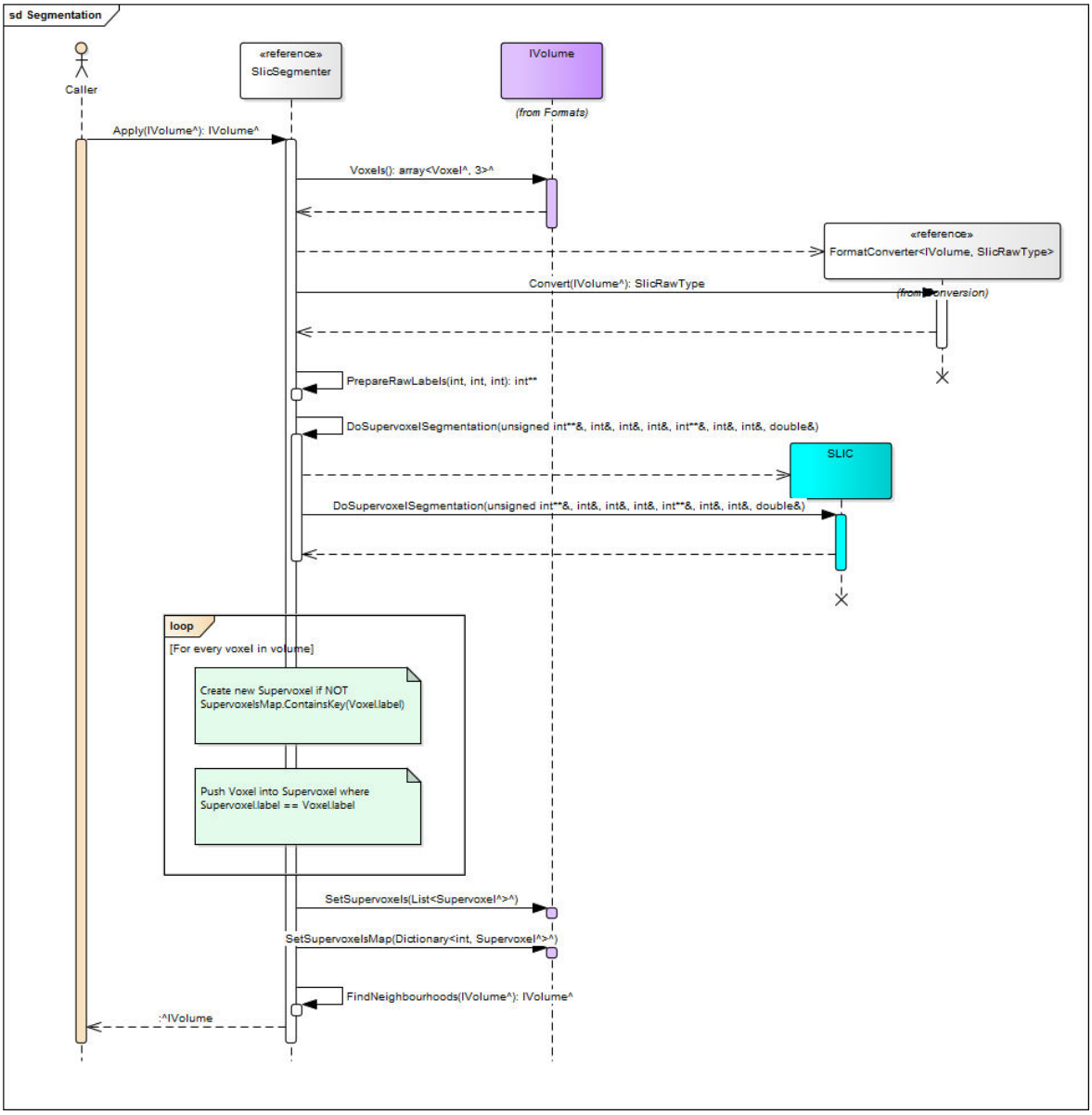


Figure 47 - Procedure of oversegmentation of volume into supervoxels - sequence diagram.

7.7 Features extraction

In terms of implementation, features extraction together with statistics extraction belongs to the most complex parts of the solution. Both are heavily templated and incorporate template method and template class patterns. Thanks to template method, it is possible to use the same features extractor to extract features from different classes – volumes, supervoxels and various combinations of them (e.g. from all supervoxels in a particular volume).

There are two base interfaces in the hierarchy of features and feature extractors – *IFeature* and *IFeaturesExtractor*. Besides other concrete features, there is one more important class – *CompositeFeature*. This feature contains a list of instances of classes realising interface *IFeature*. Because *CompositeFeature* itself realises *ICompositeFeature* interface (which is derived from *IFeature*), it is possible to compose

features. An overview of hierarchy can be seen in Figure 48. There are only a few examples of features and feature extractors in this figure.

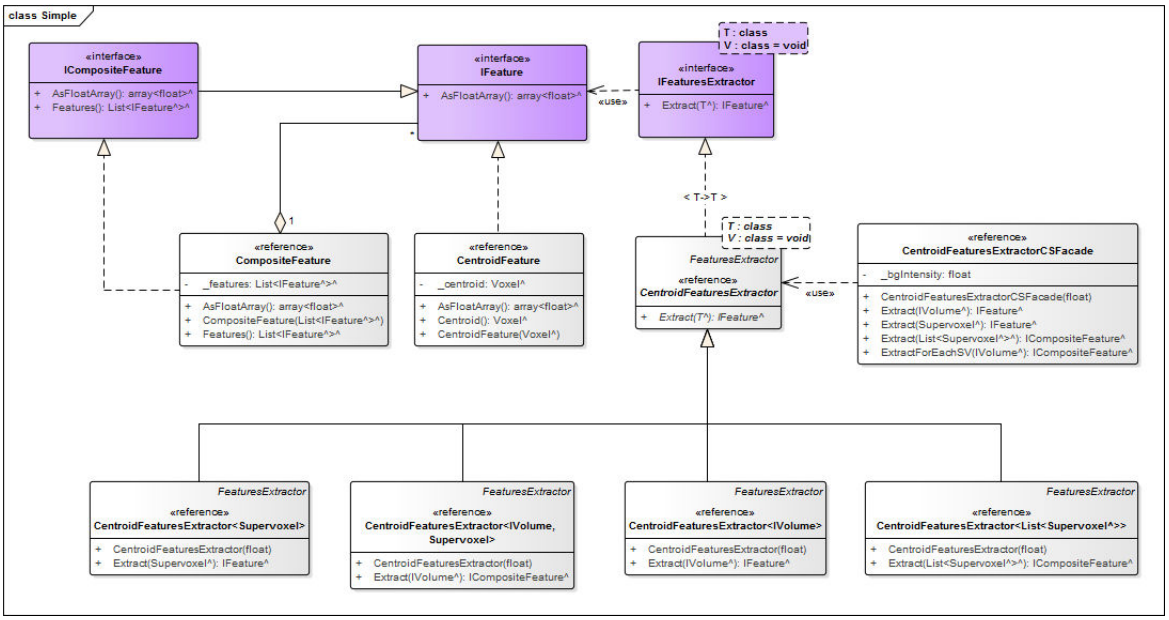


Figure 48 - Interfaces and classes representing features, composite features and features extractors. These classes are further used to extract features from volumes, supervoxels and their combinations in training and classification process.

IFeature guarantees that every derived class is able to export contained features as a float array. These arrays are used in training and classification process as an input for classifiers.

Because features and features extractors are heavily templated, they cannot be directly used in C# as common .NET structures. Therefore we had to implement *Façades*, which provide public methods for features extraction that can be directly used in C#. These *Façades* internally use templated extractors. An example of such Façade is a *CentroidFeaturesExtractorCSFacade*.

There is one more important class – *FeaturesEqualizer*. For MLP it is inappropriate if a set of training samples of one class is much bigger than training set of another class. In our case, the CSF has significantly smaller training set than other classes. Therefore, it is necessary to equalize the number of training samples of individual classes. This is where *FeaturesEqualizer* is used. Algorithm it uses is described by sequence diagram in Figure 49.

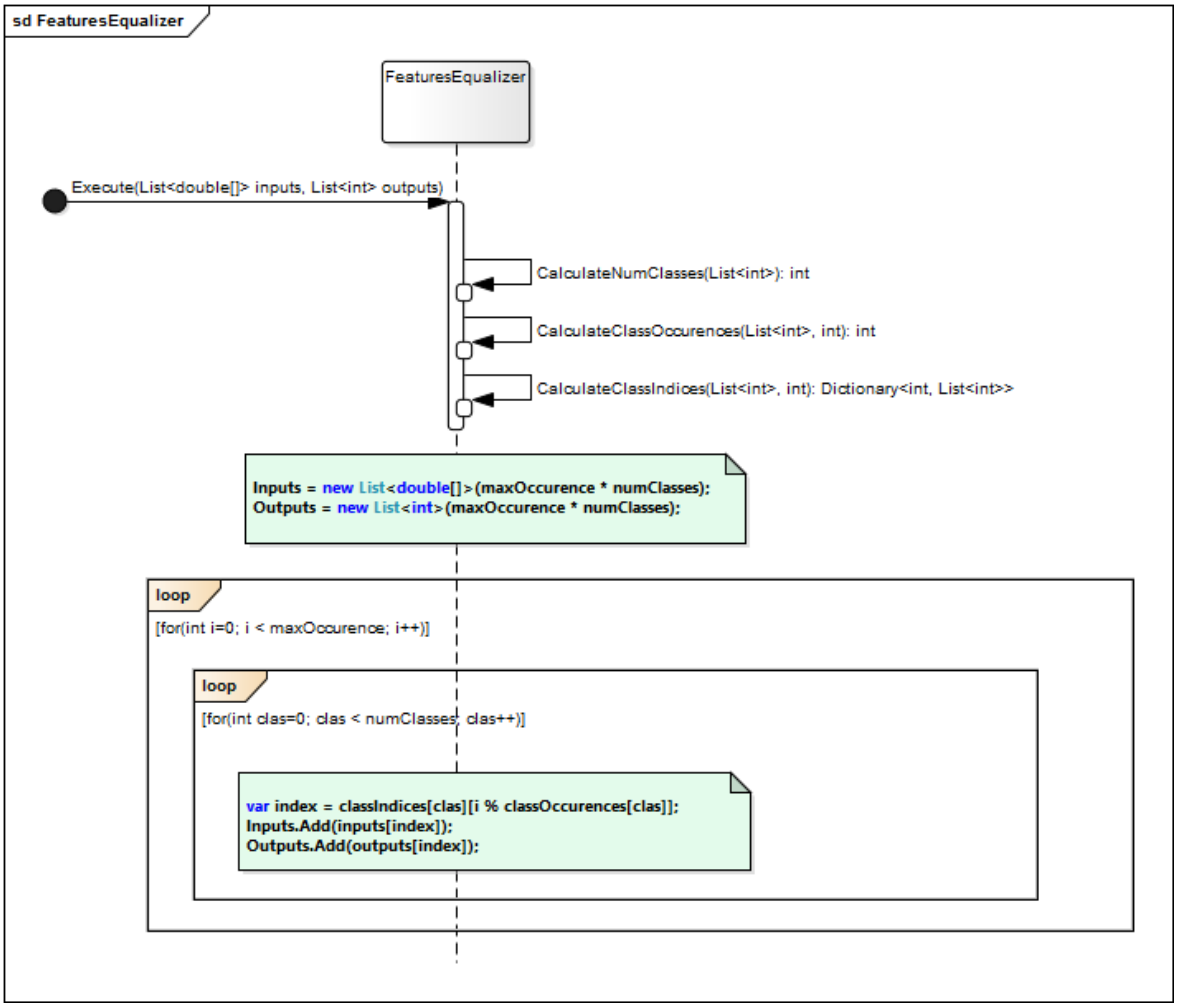


Figure 49 - Procedure features equalization. When training MLP, it is important to have equally large training sets among all classes.

7.8 Training

Compared to features extraction, training and classification are fully implemented in a C# library. For classification we use implementation of multilayer perceptron provided by Accord.NET library [37].

The main class that represents perceptron is *MLP*. This class provides public methods for training, classification and serialization (so that perceptron can be saved after training).

If we include features extraction, equalization and other activities needed for training, we get a complex sequence of method calls. In Annexes we show a simplified sequence diagram (Figure 59).

Training sequence is implemented in *TrainingWindowController* class which also serves as an interface between GUI and application logic. It also gathers data from training window that were set by user.

7.9 Classification

Overall classification process is very similar to training. The biggest difference is that after classification rather complex statistics about results are extracted (Annexes, Figure 60).

7.10 Visualization

For visualization, we mainly used 3rd party tool Slicer3D. Nevertheless, we implemented our own Maximum intensity projection (MIP) visualizer. Implementation is based on OpenTK[7], which is a low level wrapper of a popular library OpenGL[8]. We incorporate two visualization windows – overview and detail. The MIP visualization is shown in Overview window (which is also the interaction window – Figure 50 left) and highlighted voxels are shown in a Detail window (Figure 50 right).

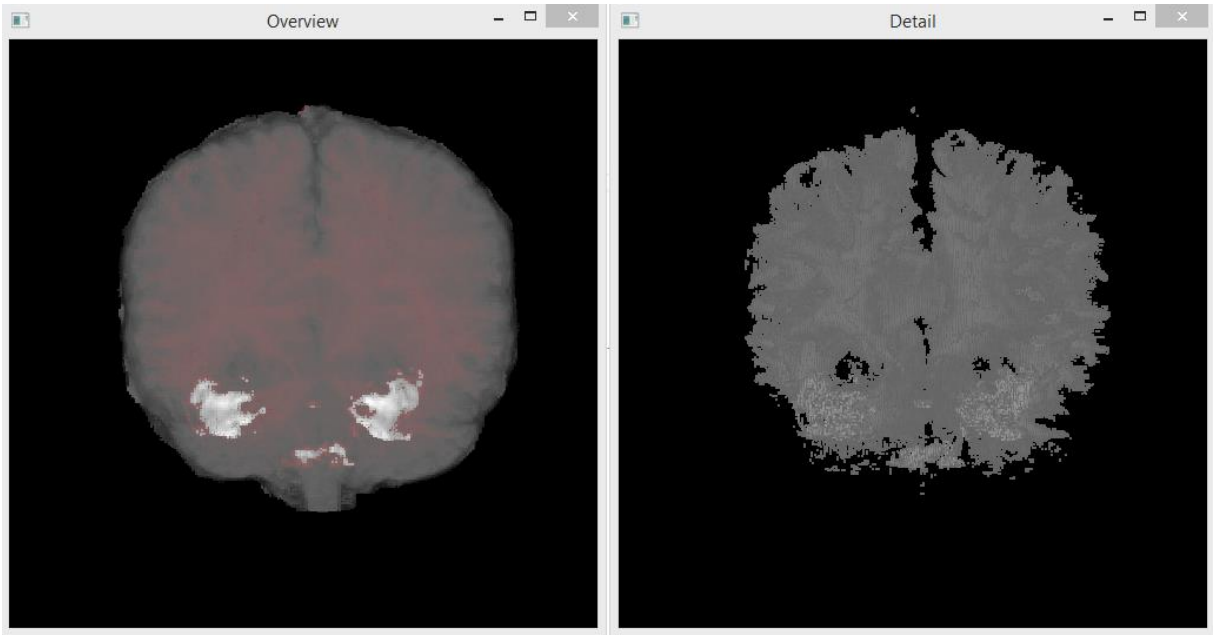


Figure 50 - MIP Overview window and Detail window.

As it is in case of Training and Classification windows, Visualization window has also its controller. Controller serves as a mid-layer between low-level visualization and Visualization window (Figure 51). For example, it is possible to highlight certain intensities both in Visualization window and in Overview window. Through two way data binding provided by WPF, controller keeps parameters influencing visualization synchronized between windows.

[7] OpenTK - http://www.opentk.com/
[8] https://www.opengl.org/

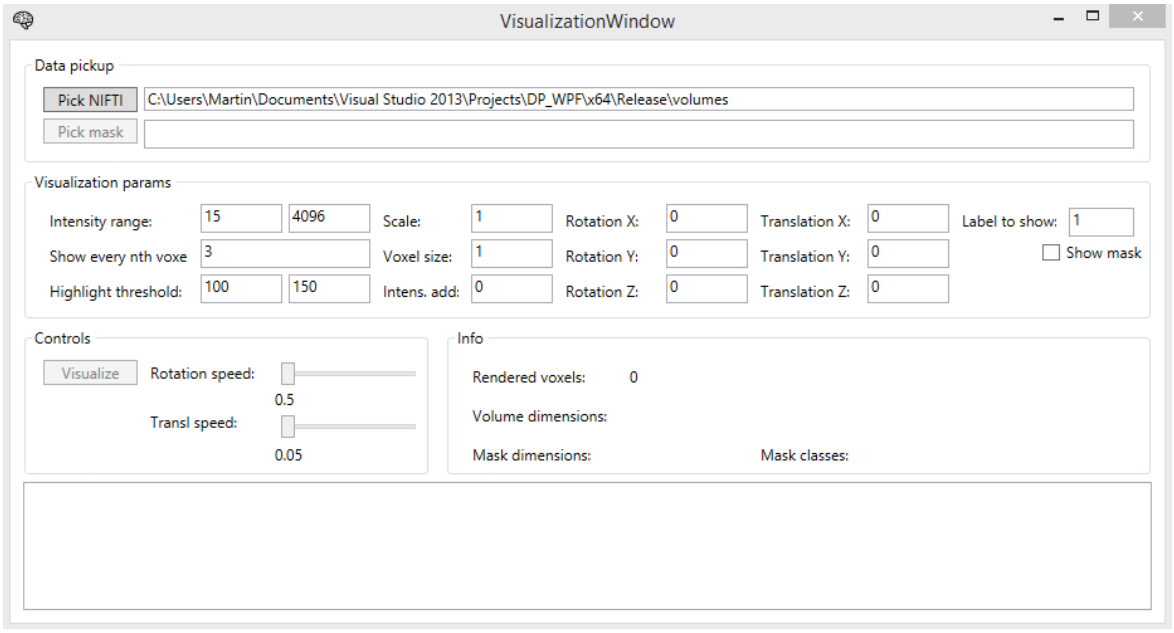


Figure 51 - MIP Visualization window.

8 Conclusion and future work

In the first term we analysed and described domain of clinical practice in context of medical imaging. We concentrated on radiologists and consulted our assumptions about their needs with experts from clinical practice. In this term we also looked for technical means that would help us in later phases of this project.

In the second term we proposed first method for tissue segmentation. In this term we did not fully concentrate on one concrete tissue. We tried to apply oversegmentation using SLIC algorithm on many different volumes and observed its performance (visually). As mentioned we proposed and partially implemented method for tissue segmentation based on merging neighbouring supervoxels with similar qualities. This approach was found to be promising if we wanted to use coarse-to-fine technique, as the merge process produced visually pleasing regions (Figure 25 right).

In the third term we proposed a final method for segmenting brain tissues from MR volumes. We thoroughly evaluated performance of SLIC algorithm as far as supervoxelization is concerned. After supervoxelization we applied extracted features from every supervoxel and trained two different classifiers – MLP with 6 hidden layers and SVM using quadratic kernel. We evaluated classification performance and proposed ways how to improve it (these will be applied in the next term). Based on evaluation of SLIC and classification performance, we estimated overall segmentation performance that we were able to reach at this moment.

Finally, we proposed, implemented and evaluated a fully automatic method for segmentation of brain from MR images. Our method is based on oversegmentation using supervoxels and classification using MLP. Method differs from most other state-of-the-art brain segmentation methods as it uses supervoxels instead of voxels as elementary unit. Supervoxels are classified into four classes *{BG, CSF, GM, WM}* and they are described by set of features *fsv = {Normalised intensity histogram, Normalised intensity histogram of voxels in neighbouring supervoxels, Normalized Euclidean distance from brain centre, Angles between supervoxel centroid and brain centre}*. Supervoxels resist to noise better than individual voxels as they can be directly described by statistical features based on contained voxels. As shown in Figure 41, position of supervoxel is characteristic for every class.

We have successfully implemented and evaluated proposed method using combination of C++, C++/CLI and C#. As shown in Table 3, our results are highly promising in context of current state-of-the-art methods and proposed method (as is) will be subject of further research.

Problems we face are mainly caused by noise in MR data, oversegmentation error and misclassification. In the future work, we are going to handle all these opened issues in order to increase segmentation success rate. First of all we are going to thoroughly

evaluate different techniques for noise reduction. Many subjects in IBSR-18 contain rather obvious periodic noise. Supervoxels with higher oversegmentation error (which tend to be misclassified, too) have usually bigger standard deviation and lower MLP excitation rate. We are going to use this information to identify potentially oversegmented and/or misclassified supervoxels and either split them into smaller supervoxels (which will be classified individually) or use some other segmentation technique, e.g. majority voting using non-rigidly registered atlases. Next option is to use and evaluate performance of another oversegmentation algorithm. MonoSLIC [5] reports promising results in medical imaging oversegmentation. In preprocessing we are going to investigate other options for brain extraction and compare it with BET. Last, we are going to include shape into features describing supervoxel.

References

[1] IZAWA, Jonathan I., Laurence KLOTZ, D. ROBERT SIEMENS, Wassim KASSOUF, Alan SO, John JORDAN, Michael CHETNER and Alla E. IANSAVICHENE. Prostate cancer screening: Canadian guidelines 2011. *Journal of the Canadian Urological Association* [online]. 2011, vol. 5, no. 4, pp. 235–240. ISSN 19116470.

[2] FLEISHON, Howard B., Mythreyi BHARGAVAN and Cristian MEGHEA. Radiologists' reading times using PACS and using films: One practice's experience. *Academic Radiology* [online]. 2006, vol. 13, no. 4, pp. 453–460. ISSN 10766332.

[3] KHOO, Lisanne A L, Paul TAYLOR and Rosalind M GIVEN-WILSON. Computer-aided detection in the United Kingdom National Breast Screening Programme: prospective study. *Radiology* [online]. 2005, vol. 237, no. 2, pp. 444–449. ISSN 0033-8419.

[4] SHAREEF, N, D L WANG and R YAGEL. Segmentation of medical images using LEGION. *IEEE transactions on medical imaging* [online]. 1999, vol. 18, no. 1, pp. 74–91. ISSN 0278-0062.

[5] ZUZANA, K, Markus HOLZER and Rene DONNER. Over-Segmentation of 3D Medical Image Volumes based on Monogenic Cues. 2014.

[6] SHARMA, NEERAJ, et al. Automated medical image segmentation techniques. *Journal of medical physics*. 2010, vol. 35, no. 1, pp. 3.

[7] HEIMANN, Tobias and Hans Peter MEINZER. Statistical shape models for 3D medical image segmentation: A review. *Medical Image Analysis* [online]. 2009, vol. 13, no. 4, pp. 543–563. ISSN 13618415.

[8] ACHANTA, Radhakrishna, Appu SHAJI, Kevin SMITH, Aurelien LUCCHI, Pascal FUA and Sabine SÜSSTRUNK. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. 2012, vol. 34, no. 11, pp. 2274–2281. ISSN 01628828.

[9] PASCHOS, G. Perceptually uniform color spaces for color texture analysis: an\nempirical evaluation. *IEEE Transactions on Image Processing* [online]. 2001, vol. 10, no. 6, pp. 932–937. ISSN 10577149.

[10] SATO, Kazuhito, Sakura KADOWAKI, Hirokazu MADOKORO, Momoyo ITO and Atsushi INUGAMI. Unsupervised segmentation for MR brain images. *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and*

Communication Technologies - ISABEL '11 [online]. 2011, pp. 1–5.

[11] LUCCHI, Auŕlien, Kevin SMITH, Radhakrishna ACHANTA, Graham KNOTT and Pascal FUA. Supervoxel-based segmentation of mitochondria in em image stacks with learned shape features. *IEEE Transactions on Medical Imaging* [online]. 2012, vol. 31, no. 2, pp. 474–486. ISSN 02780062.

[12] GAIKWAD, D. P., P. ABHANG and P. BEDEKAR. Medical image segmentation for brain tumor detection. *Proceedings of the International Conference & Workshop on Emerging Trends in Technology - ICWET '11* [online]. 2011, no. Icwet, pp. 63.

[13] WANG, Li, Chunming LI, Quansen SUN, Deshen XIA and Chiu-yen KAO. Brain MR image segmentation using local and global intensity fitting active contours/surfaces. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008*. 2008, pp. 384–392. ISBN 354085987X.

[14] ZUVA, Tranos, Oludayo OLUGBARA, Sunday OJO and Seleman M. NGWIRA. Image Segmentation, Available Techniques, Developments and Open Issues. *Canadian Journal on Image Processing and Computer Vision*. 2011, vol. 2, no. 3, pp. 20–29.

[15] SAHOO, P.K, S SOLTANI and A.K.C WONG. A survey of thresholding techniques. *Computer Vision, Graphics, and Image Processing* [online]. 1988, vol. 41, no. 2, pp. 233–260. ISSN 0734189X.

[16] MOON, Jucheol and Hyun I KIM. A Support Vector Machine based Classifier to Extract Abnormal Features from Breast Magnetic Resonance Images. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008*. 2012, pp. 148–152.

[17] SHATTUCK, D W, et al. Magnetic resonance image tissue classification using a partial volume model. *NeuroImage* [online]. 2001, vol. 13, no. 5, pp. 856–76. ISSN 1053-8119.

[18] FELZENSZWALB, Pedro F. and Daniel P. HUTTENLOCHER. Efficient graph-based image segmentation. *International Journal of Computer Vision* [online]. 2004, vol. 59, no. 2, pp. 167–181. ISSN 09205691.

[19] MORI, Greg. Guiding model search using segmentation. In: *Proceedings of the IEEE International Conference on Computer Vision* [online]. 2005, pp. 1417–1423. ISBN 076952334X.

[20] VEKSLER, Olga. Superpixels and supervoxels in an energy optimization

framework (Code). 2010.

[21] ENGEL, David and Cristobal CURIO. Scale-invariant medial features based on gradient vector flow fields. *2008 19th International Conference on Pattern Recognition* [online]. 2008, pp. 1–4. ISSN 1051-4651.

[22] ACHANTA, Radhakrishna, Appu SHAJI, Kevin SMITH, Aurelien LUCCHI, P FUA and S SUSSTRUNK. SLIC Superpixels. *EPFL Technical Report 149300* [online]. 2010, no. June, pp. 15. ISSN 149300.

[23] TEAM, opencv dev. *Introduction to Support Vector Machines* [online]. 2015. Available at: http://docs.opencv.org/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

[24] SHUBHANGI, D. C. and P. S. HIREMATH. Support vector machine (SVM) classifier for brain tumor detection. *Proceedings of the International Conference on Advances in Computing, Communication and Control - ICAC3 '09* [online]. 2009, pp. 444.

[25] TEAM, opencv dev. *Neural Networks* [online]. 2015. Available at: http://docs.opencv.org/2.4.9/modules/ml/doc/neural_networks.html

[26] LI, Chunming, Chiu-yen KAO, John C GORE and Zhaohua DING. Implicit Active Contours Driven by Local Binary Fitting Energy. nedatováno, pp. 12–15.

[27] KONG, Youyong, Yue DENG and Qionghai DAI. Discriminative clustering and feature selection for brain MRI segmentation. *IEEE Signal Processing Letters* [online]. 2015, vol. 22, no. 5, pp. 573–577. ISSN 10709908.

[28] VERMA, Nishant, Matthew C COWPERTHWAITE and Mia K MARKEY. Superpixels in brain MR image analysis. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Annual Conference* [online]. 2013, vol. 2013, pp. 1077–80. ISSN 1557-170X.

[29] LAROBINA, Michele and Loredana MURINO. Medical image file formats. *Journal of Digital Imaging* [online]. 2014, vol. 27, no. 2, pp. 200–206. ISSN 1618727X.

[30] ASSOCIATION, National Electrical Manufacturers. *DICOM Homepage* [online]. 2015. Available at: http://dicom.nema.org/

[31] WIDEMAN, Graham. Orientation and Voxel-Order Terminology: RAS, LAS, LPI, RPI, XYZ and All That. *Grahamwideman Web site* [online]. 2015. Available at: http://www.grahamwideman.com/gw/brain/orientation/orientterms.htm

[32] ROHLFING, Torsten. Image similarity and tissue overlaps as surrogates for image registration accuracy: Widely used but unreliable. *IEEE Transactions on Medical Imaging* [online]. 2012, vol. 31, no. 2, pp. 153–163. ISSN 02780062.

[33] NITRC. IBSR MR Dataset. *NITRC Web site* [online]. 2015. Available at: https://www.nitrc.org/projects/ibsr

[34] VERMA, Nishant, Gautam S. MURALIDHAR, Alan C. BOVIK, Matthew C. COWPERTHWAITE, Mark G. BURNETT and Mia K. MARKEY. Three-dimensional brain magnetic resonance imaging segmentation via knowledge-driven decision theory. *Journal of Medical Imaging* [online]. 2014, vol. 1, no. 3, pp. 034001. ISSN 2329-4302.

[35] SMITH, Stephen M. Fast robust automated brain extraction. *Human Brain Mapping* [online]. 2002, vol. 17, no. 3, pp. 143–155. ISSN 10659471.

[36] ITK. *ITK* [online]. 2015. Available at: http://www.itk.org/

[37] *About: Accord.NET. Accord.NET Web site* [online]. 2016. Available at: http://accord-framework.net/index.html

Annexes

A. Technical documentation

In technical documentation we present technical details and engineering documentation of selected parts of this project. In most cases we use UML v. 2.0 as a modelling standard and technique. We also show snippets of code where appropriate.

A1. Requirements specification

There were various requirements as far as proposed method is concerned. These requirements are shown on Use case diagram in Figure 52.

In context of our method we identified one main role as far as users are concerned. In most cases MR images are used by radiologists. When we defined requirements for our method we based them on radiologists' needs.

In practice radiologists work with medical data in standard format. As we mentioned in 5.1, to date the most used formats are DICOM and Nifti. For simplicity we decided to use Nifti. Nevertheless we designed our solution to be modular and able to simply add support for other formats (for this modularity is mostly responsible use of template classes *FormatConverter<TInput, TOutput>*). Although radiologists use mostly one tool provided by their hosptital, sometimes it is necessary to exchange data between radiologists from different hospitals or even countries. Beside this sometimes it is required to anonymise exchanged data (DICOM contains information about the patient).

Single MR volume can be visualized in different forms, based on current radiologist's needs. Radiologist needs to be able to filter and process such volume. An example of such filtration is increase or decrease of brightness, power law transform, Gaussian filtering and many others. For this reason we defined volume filtration as one of requirements.

All these requirements would be useless if it were not possible to visualize processed data. Therefore we require some way of data visualization.

The last requirement comes out of the assignment of this work. We require from our method to segment desired tissues (in our case CSF, GM and WM) with minimal or no user interaction.

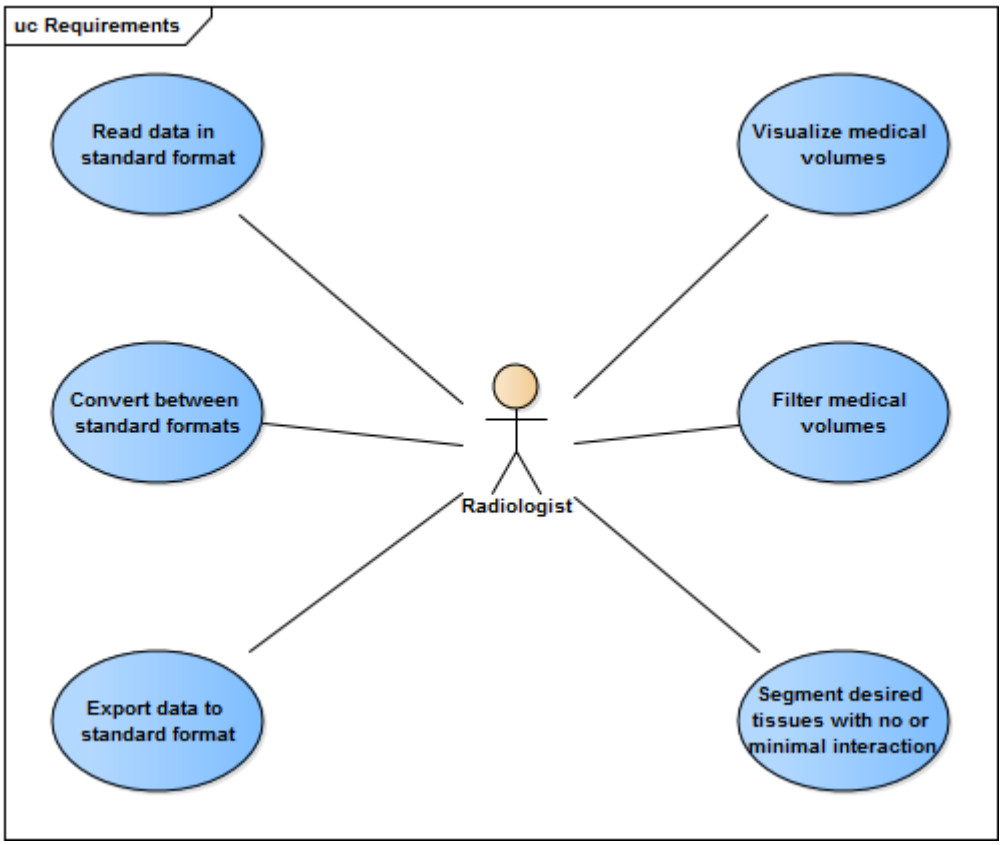


Figure 52 - Method requirements from radiologist's point of view. Use case diagram.

A2. Application modules and solution organisation

Application was designed as a Visual studio solution that physically contains multiple projects. On the conceptual level, application consists of multiple modules. Except *Statistics* and *Classification*, all modules are implemented in single project. Overview of projects and modules are shown in Figure 53.

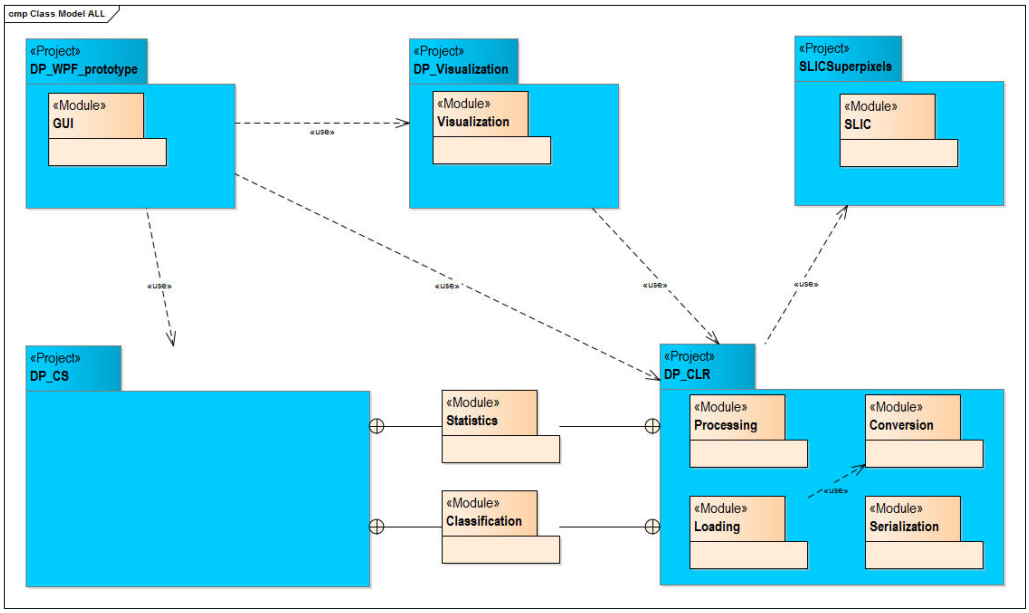


Figure 53 - Design of application in terms of modules and Visual studio projects.

In the later stages of software development lifecycle we decided to implement as many parts of final product as possible in C# language. This decision was supported by fact that prototyping in C# is much more rapid than in C++. This is where DP_CS was introduced. Nevertheless, before this decision many modules were developed either in pure C++ or in managed C++/CLI in DP_CLR project (in fact, before DP_CLR project there was DP_Native that did not support managed code at all). DP_CS is strongly dependent on original DP_CLR library.

A3. User interaction and graphical user interface

The entry point to the application is shown in Figure 54. It is a simple WPF[9] window. The only purpose of this window is to navigate user to either Training, Classification or Visualization windows. SVM training and classification windows are also active, but currently they are only experimental.

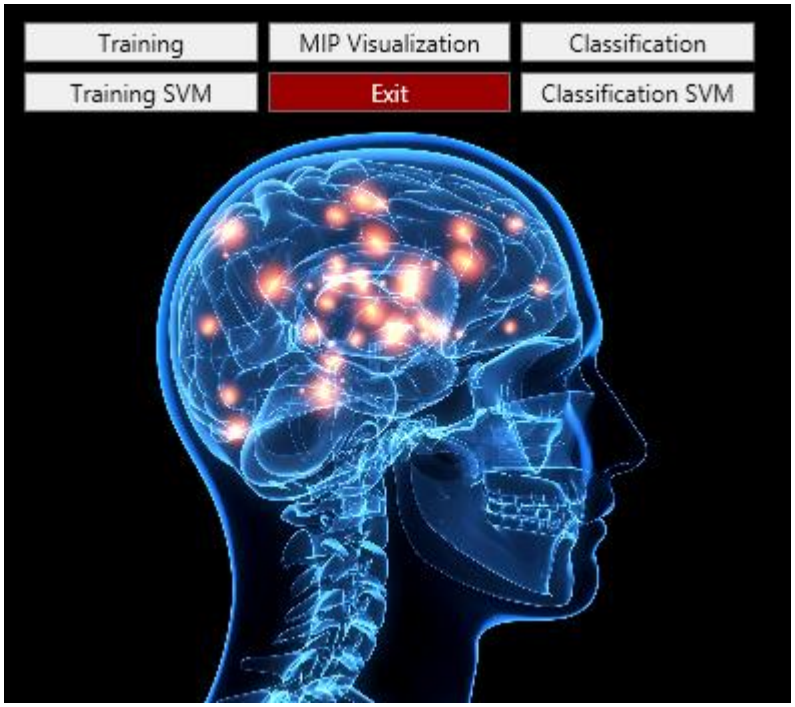


Figure 54 - Application entry point (Main window).[10]

Two most important windows – Training and Classification (Figure 55) are also designed as WPF windows. The main focus was on simplicity so that all the most important settings are always visible.

[9] WPF: https://msdn.microsoft.com/en-us/library/aa970268(v=vs.100).aspx
[10] Brain image taken from http://www.mixscoop.com/wp-content/uploads/2015/06/web-brain-getty-c-DONTUSEAGAIN1.png

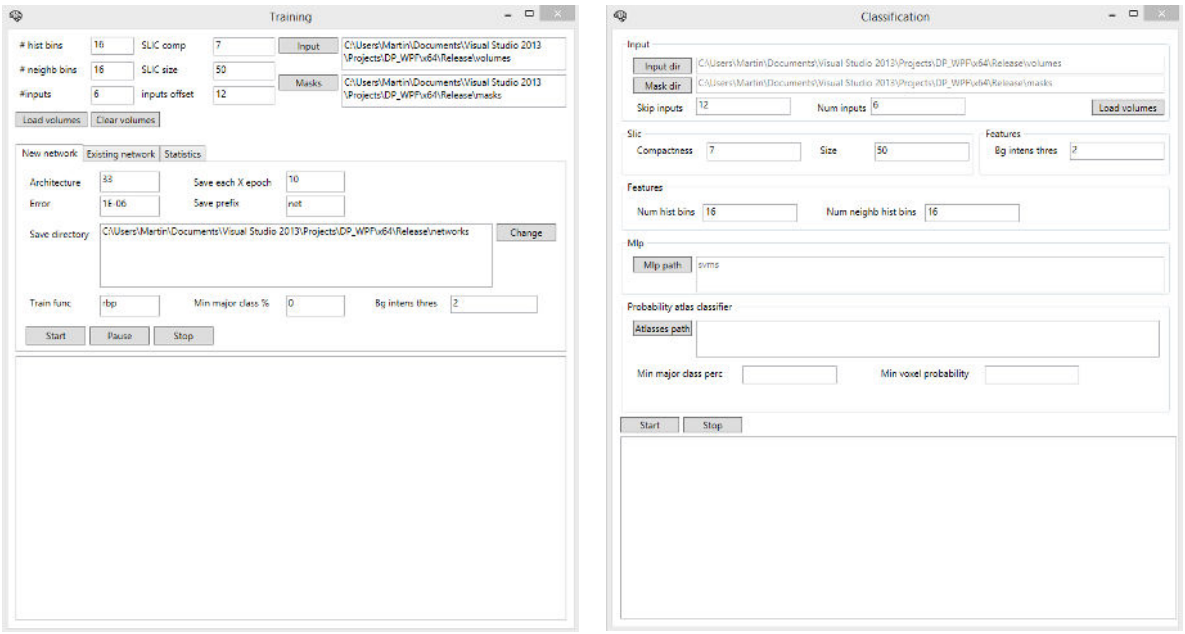


Figure 55 - Training and Classification windows.

WPF windows themselves handle only interaction events, but the application logic (training and classification) are delegated to window controllers. Every window has its own controller. If a user changes some setting in the graphical interface, the change is immediately reflected in controller and vice versa through two way data binding provided by WPF. Besides data binding and complete flows of training and classification, controllers are also responsible for data loading, processing and serialization and feedback. Feedback is given to the user in text form. The complete hierarchy of window controllers is in Figure 56.

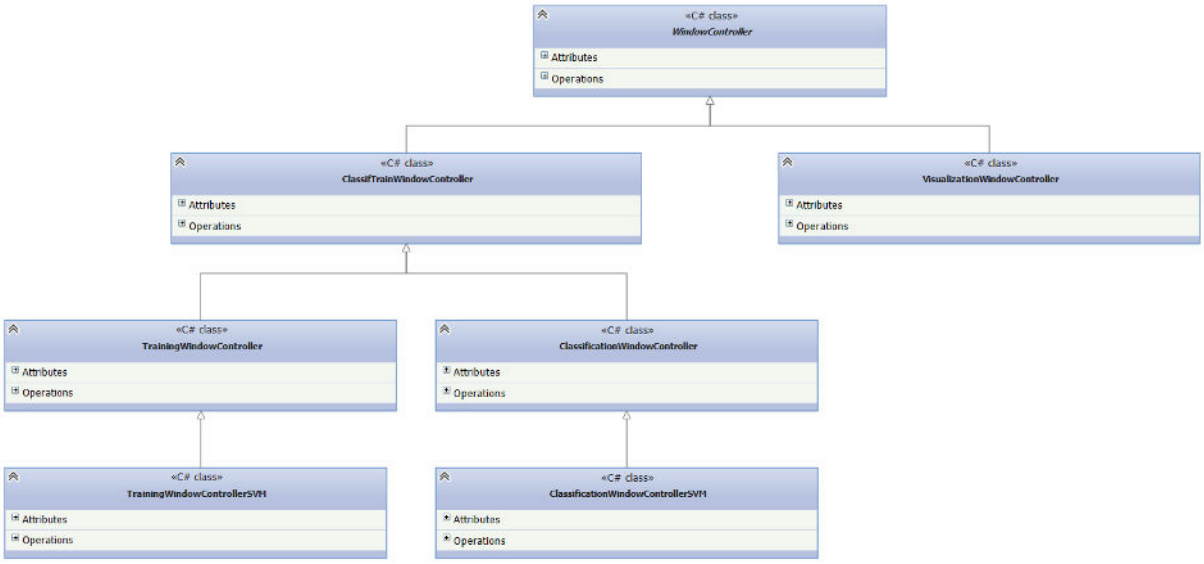


Figure 56 - Hierarchy of window controllers.

The main method of *TrainingWindowController* is *Train.* For classification, it is *Classify* in *ClassificationWindowController*. Source codes of both methods are shown in section A7.

A4. Internal formats and loading

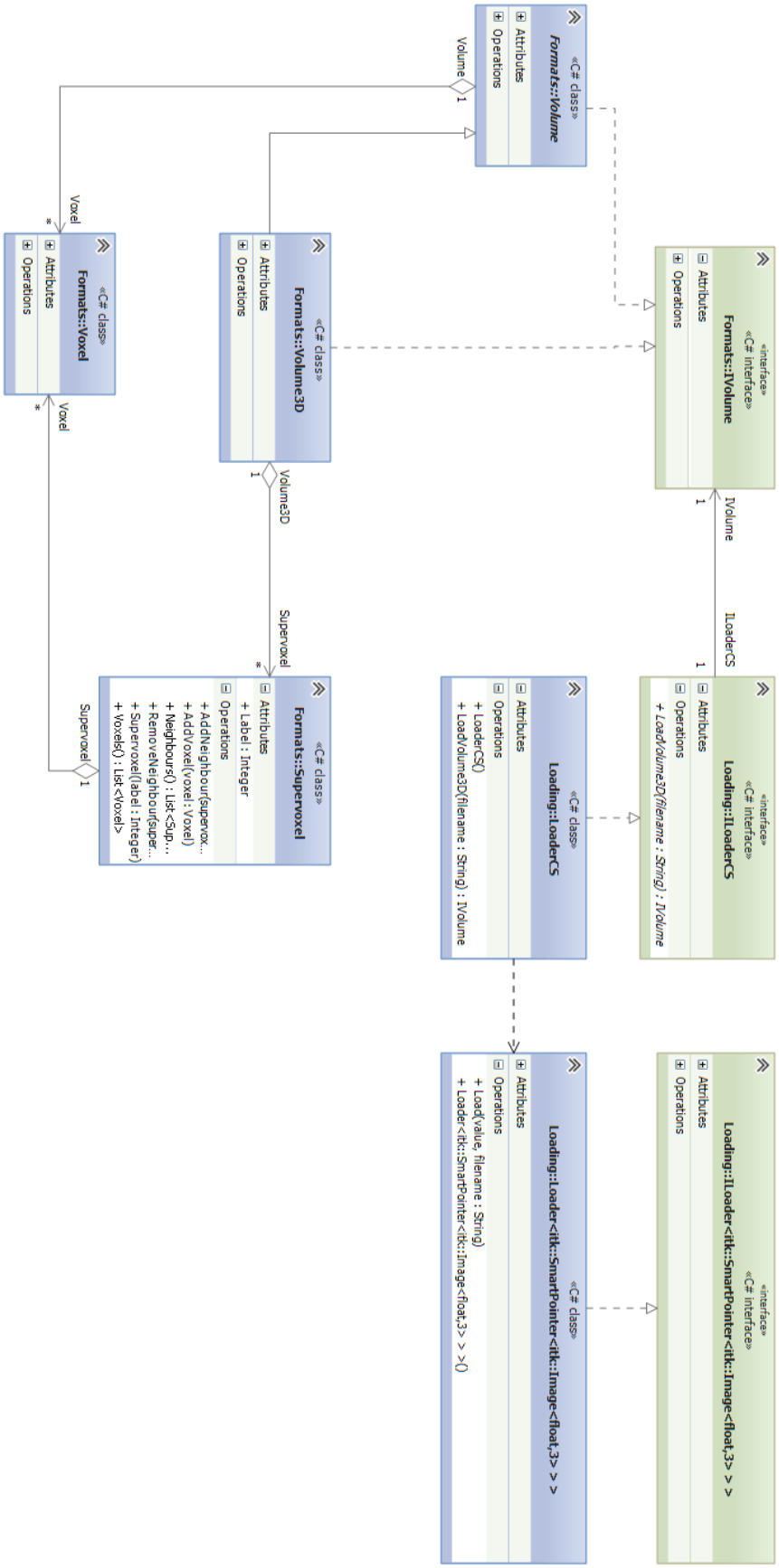


Figure 57 - Internal data structures (formats) designed to work with volumes, supervoxels and voxles and structures responsible for data loading.

A5. Filtration

Every filter used in our application implements *IFilter* interface. This guarantees that filters can be applied and also reverted. Obviously, some filters cannot be reverted algorithmically, as after they are applied, the original information gets lost. Therefore these filters must store original values so they can be restored. An example of such filter is *NormalizationFilter* class. Source codes of its *Apply* and *Revert* methods are shown below.

```
IVolume^ NormalizationFilter::Apply(IVolume^ input)
{
	auto voxelsList = input->VoxelsList();
	_intensities = gcnew List<float>(voxelsList->Count);

	for each(auto voxel in voxelsList)
		_intensities->Add(voxel->value);

	List<float> sortedIntensities(_intensities);
	sortedIntensities.Sort();

	int quantileIndex = Math::Min(sortedIntensities.Count - 1,

(int)(sortedIntensities.Count * 0.99999));

	for each(auto voxel in voxelsList)
	{
		voxel->value =
		Math::Min(1.0f, voxel-
>value / sortedIntensities[quantileIndex]);
	}

	return input;
}

IVolume^ NormalizationFilter::Revert(IVolume^ input)
{
	auto voxelsList = input->VoxelsList();

	for (int i = 0, numVoxels = voxelsList->Count; i < numVoxels; i++)
		voxelsList[i]->value = _intensities[i];

	return input;
}
```

A6. Format conversion

One of the main requirements was to enable user to convert medical data from any standard medical format to another. As we wanted to provide a modular solution that would be able to simply add another format, we decided to use C++ templates. The use of templates is convenient as it allows to use class with the same name to convert between various kinds of formats. All converters implement following interface:

```
template<class TInput, class TOutput>
public interface class IFormatConverter
{
```

```
        TOutput^ Convert(TInput input);
};
```

In our application we needed to convert between various formats which gave rise to converters shown in Figure 58.

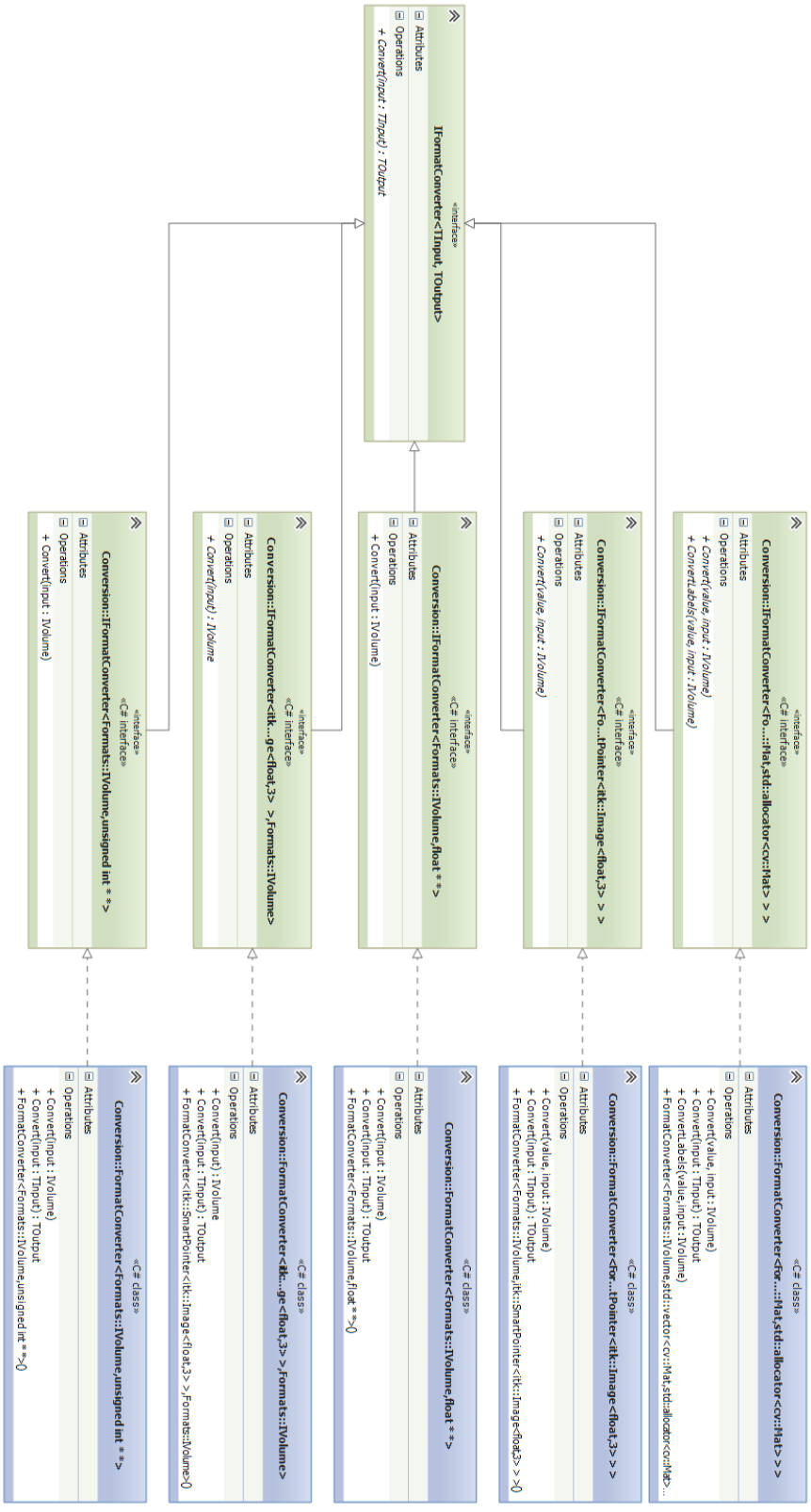


Figure 58 - Concrete implementations of converters that convert data between various formats.

A7. Training and classification

Both training and classification use wide variety of classes and methods from both DP_CS and DP_CLR libraries.

Complete training flow is defined in *TrainingWindowController* in method *Train*, which also uses other private methods of this class. The source of *Train* method is in shown below:

```
protected virtual void Train(double bgIntensThreshold, int numInputs, int inputsOffset, int slicSize, double slicComp, int histBinCount,
    int neighbHistBinCount, int[] mlpArchitecture, double mlpError, string mlpTrainFunc, double minMajorClassPercentage, int startIteration,
    string volumesDir, string masksDir, List<IVolume> volumes, ActivationNetwork network = null)
{
    if (volumes == null || volumes.Count == 0) { UpdateWindow("Volumes were not loaded"); return; }

    var featuresEqualizer = new FeaturesEqualizer();
    List<Supervoxel> aggrSupervoxels;
    List<ICompositeFeature> aggrFeatures;
    List<IStatistic> aggrMajorLabels;
    List<double[]> inputs;
    List<int> outputs;
    ExtractFeatures(volumes, (float)bgIntensThreshold, histBinCount, neighbHistBinCount, _ClassNames.Values.ToList(), minMajorClassPercentage, 5,
        out aggrSupervoxels, out aggrFeatures, out aggrMajorLabels, out inputs, out outputs);

    featuresEqualizer.Execute(inputs, outputs);

    MLP classifier = network == null ? new MLP(inputs[0].Length, outputs.Distinct().Count(), mlpArchitecture, mlpError) : new MLP(network, mlpError);
    classifier.OnEpochComplete += UpdateWindow;
    UW("Start training");
    classifier.Train(featuresEqualizer.Inputs, featuresEqualizer.Outputs, mlpTrainFunc, startIteration);
}
```

In the first step the method checks whether volumes are loaded. In the next step, it prepares an instance of *FeaturesEqualizer* class and instances of collections of supervoxels, features and other statistics. These instances are assigned in *ExtractFeatures* method, which is responsible for extraction of particular features from both supervoxels and volumes. There are two extra variables – *inputs* and *outputs*. These lists of primitive types are used as inputs and outputs for neural network in training. After these features are extracted, they must be equalized so that every class has the same amount of training samples. In the last step, classifier is instantiated and trained. The instantiation depends on input arguments of *Train* method – either new classifier is created, or an existing classifier is loaded. Simplified sequence diagram of training flow is in Figure 59.

Classification, on the other hand, is defined in *ClassificationWindowController* in method *Classify*. Source code of *Classify* method is below:

```
protected virtual void Classify(double bgIntensThreshold, string masksDir,  int histBinCount,
    int neighbHistBinCount, List<IVolume> volumes,  string networkPath)
{
    if (volumes == null || volumes.Count == 0)
    {
        UpdateWindow("Volumes were not loaded");
        return;
    }

    /* Features extraction */
    List<Supervoxel> aggrSupervoxels;
    List<Supervoxel> aggrDefaultBgSupervoxels;
    List<ICompositeFeature> aggrFeatures;
    List<IStatistic> aggrMajorLabels;
    List<IStatistic> aggrMajorLabelsPerc;
    List<IStatistic> aggrStandDeviations;
    List<double[]> inputs;
    List<int> outputs;
    ExtractFeatures(volumes, (float)bgIntensThreshold, histBinCount, neighbHistBinCount, _ClassNames.Values.ToList(),
        5, out aggrSupervoxels, out aggrFeatures, out aggrMajorLabels, out aggrMajorLabelsPerc, out aggrStandDeviations,
        out inputs, out outputs, out aggrDefaultBgSupervoxels);

    /* Classification */
    ActivationNetwork network = (ActivationNetwork)ActivationNetwork.Load(networkPath);
    MLP classifier = new MLP(network, 1e-8);
    //UW("Start classification");
    var classifResult = classifier.Classify(inputs).Select(res => res >= 0 ? res : 0).ToList();
    //UW("End classification");
    var trueResult = (from label in aggrMajorLabels select (int)label.Values()[0]).ToList();

    /* After classification statistics and outputs */
    AfterClassifActions(classifResult, trueResult, classifier, volumes, aggrSupervoxels, aggrDefaultBgSupervoxels,
                        aggrMajorLabelsPerc, aggrStandDeviations, masksDir);

    /* Clear volumes so they must be reloaded */
    volumes.Clear();
}
```

Classification flow is very similar to training. It starts with the same steps as training. The main difference is that in case of classification a neural network is always loaded from disc, because it would not make sense to classify supervoxels using a newly created (and therefore not trained) network. The result of classification is a list of true labels. These can be afterwards compared with ground truth results. Multiple statistics are calculated after classification. This happens in *AfterClassifActions* method. Finally, collection containing volumes is cleared so that these volumes can be garbage collected. Simplified sequence diagram of classification flow is in Figure 60.

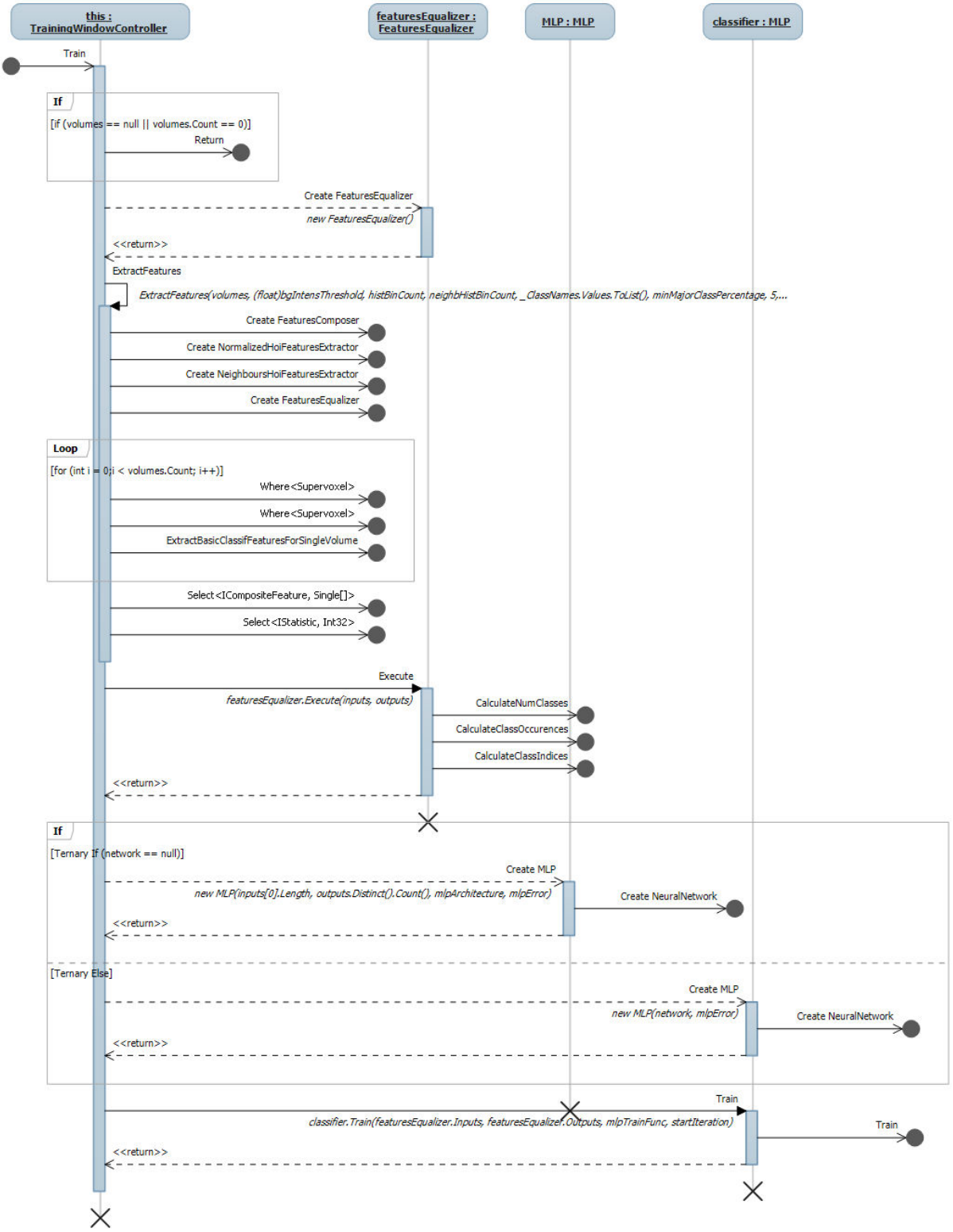


Figure 59 - Simplified training procedure. Only calls in depth 1 are directly shown.

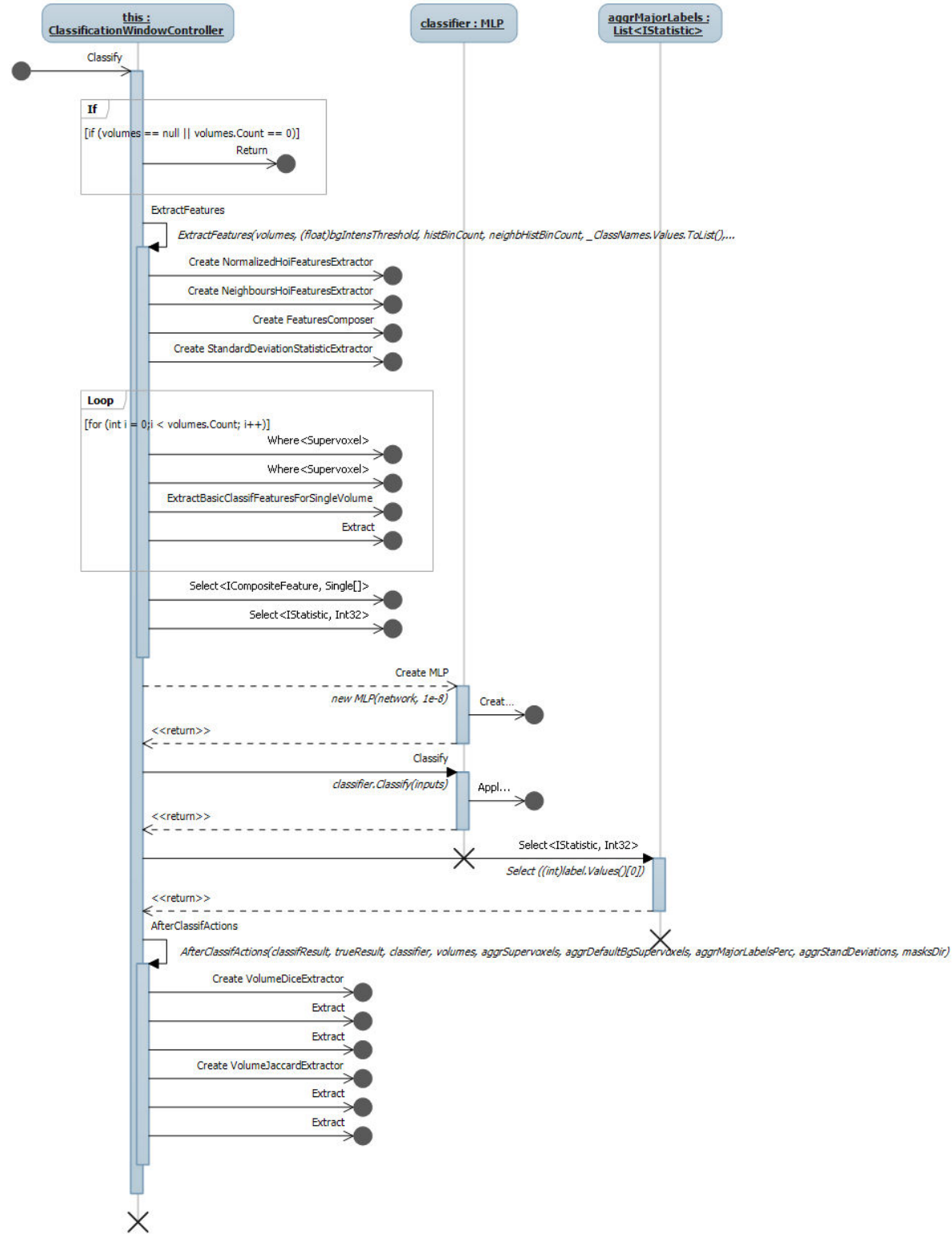


Figure 60 - Simplified classification procedure. Only calls in depth 1 are directly shown.

B. Content of attached media

Attached media contains:

Directory	*Content*
/Sources/Old	*C++ code of first prototype, which is no longer used.*
/Sources/Old/Tested approaches	*Prototype that is no longer fully used. Some parts may be used in current version, though.*
/Sources/Final	*Source codes of final product*
/Sources/MATLAB	*MATLAB implementation of selected parts of method. Mostly data analysis and classification.*
/Setup	*Installation files of product.*
/Documents/User guide	*User guide for product.*
/Documents/Masters thesis	*PDF version of this work.*
/Documents/Annotations	*EN and SK versions of annotation.*
/Samples/Volumes	*Sample Nifti volumes.*
/Samples/FiitMedical	*Sample text data in FiitMedical format.*

C. User guide

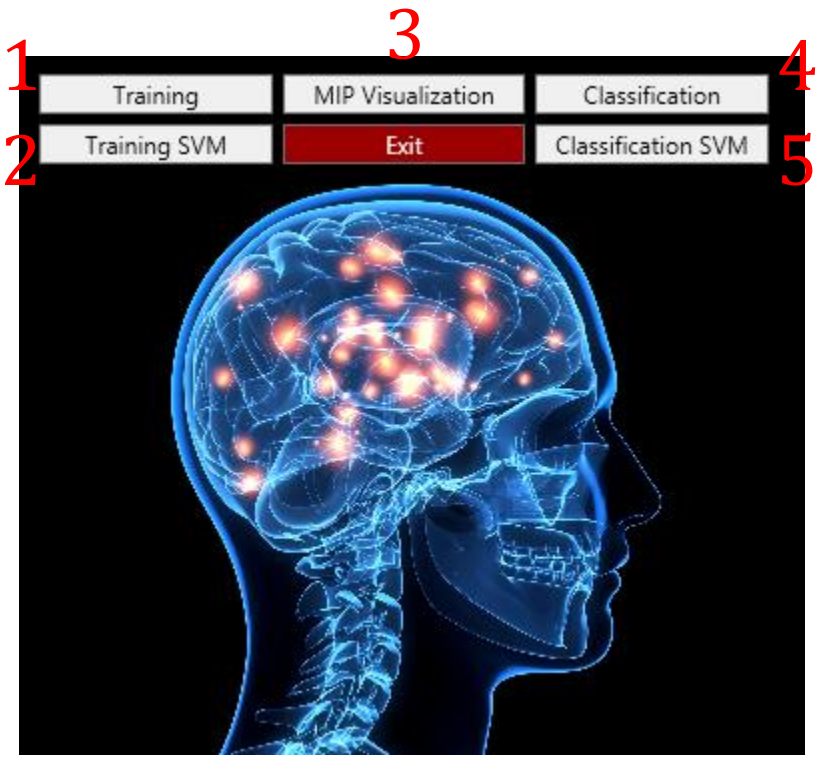
Installation

To install our product, follow these steps:

1. Open attached CD
2. Open Setup directory
3. Run Setup.exe
4. Follow instructions during installation

Main application window

Main application window (Figure 61) contains six buttons.



1 - Opens training window (Figure 62)
2 - Opens training window with SVM (experimental)
3 - Opens visualization window
4 - Opens classification window (Figure 63)
5 - Opens classification window with SVM (experimental)
Exit – Terminates application

Figure 61 - Main application window.

Training window

Training window contains multiple fields and controls. Their meaning is described in Table 5 and Table 6. User is notified during training process. Messages are shown in the big text area at the bottom of training window.

Table 5 – Training window fields.

Field name	Description
# hist bins	Number of histogram bins describing single supervoxel (integer)
# neighb bins	Number of histogram bins calculated from voxels belonging to neighbouring supervoxels (integer)
# inputs	Number of volumes to be loaded from specified directory (integer)
inputs offset	Skips specified number of volumes when loading all volumes from specified directory (integer)
SLIC comp	Compactness of supervoxel used in SLIC algorithm in oversegmentation process (float)
SLIC size	Size of supervoxel used in SLIC algorithm in oversegmentation process (integer)
Architecture	Architecture of created neural network – numbers separated by whitespace. Individual numbers represent number of neurons in hidden layers (integers separated by whitespace)
Save each X epoch	Saves neural network every time it reaches multiple of this number (ongoing serialization) (integer)
Error	When MLP reaches this value, training ends (float)
Save prefix	Prefixes this string to network name when it is being saved
Save directory	Directory where the trained MLP will be saved
Train func	Accepts two strings: rbp – resilient backpropagation lm – Levenberg-Marquardt
Min major class %	Specifies minimal percentage of majority class voxels. If a supervoxel does not have enough majority class voxels, it will not be included into training process (float number lower than 1)
Bg intens thres	Supervoxels with average intensity lower than this value will not be used in training process (integer)

Table 6 - Training window controls.

Control name	Description
Input	Choose directory with input volumes (they will be loaded in alphabetical order)[11]

[11] It is important that inputs and masks are in corresponding alphabetical order, as the first volume is mapped to first mask, second volume to second mask etc.

Masks	Choose directory with input masks (they will be loaded in alphabetical order)
Load volumes	Loads volumes and masks
Clear volumes	Removes volumes and masks from memory
Start	Starts training process
Stop	Stops training process (irreversible)
Pause	Pauses training process. It can be resumed with start
Change	Changes directory where trained MLP will be saved

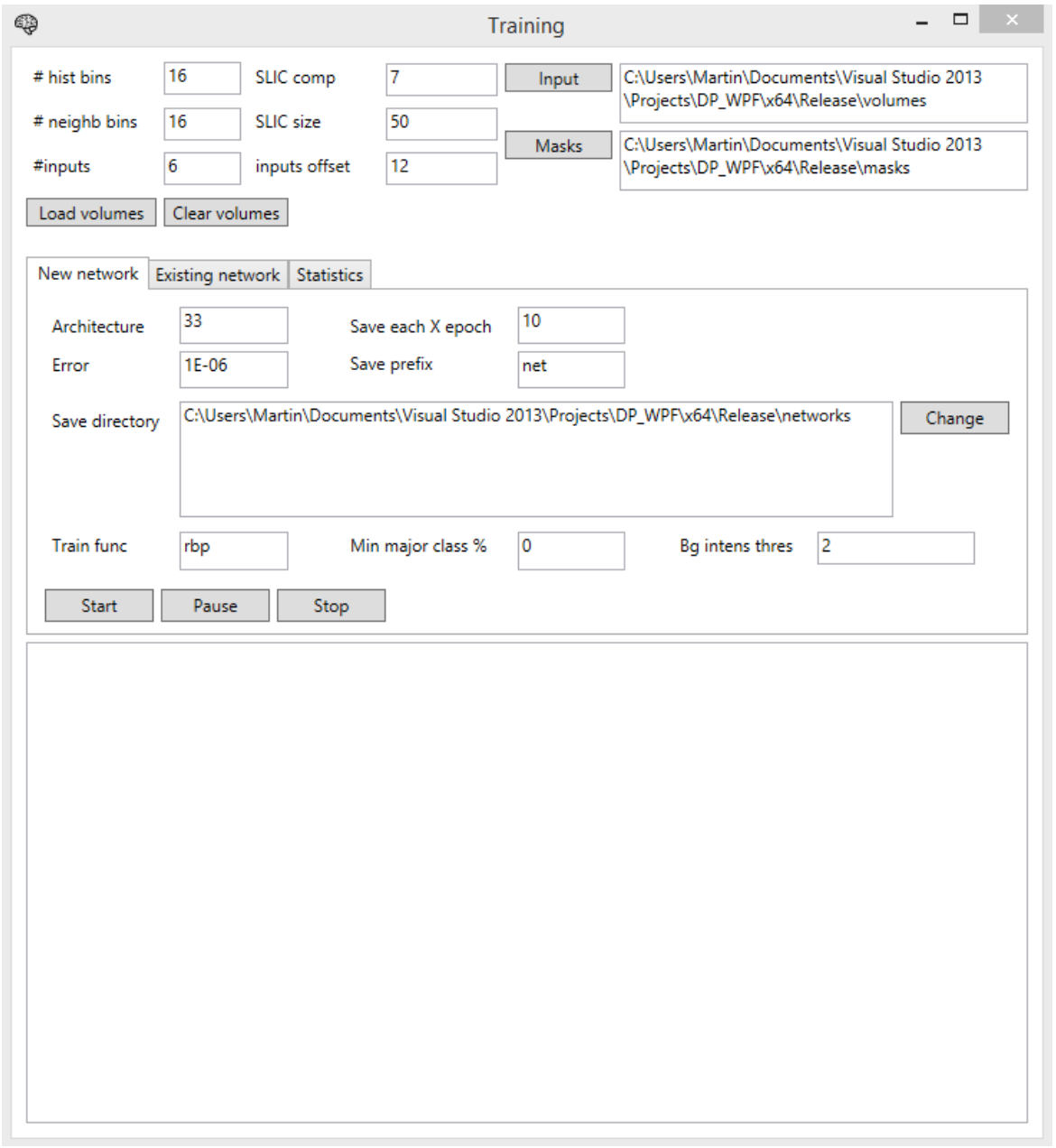


Figure 62 - Training window.

Classification window

Training window contains multiple fields and controls. Their meaning is described in Table 7 and Table 8. User is notified during training process. Messages are shown in the big text area at the bottom of training window.

Table 7 - Classification window fields.

Field name	Description
Skip inputs	Choose directory with input volumes (they will be loaded in alphabetical order)
Num inputs	Choose directory with input masks (they will be loaded in alphabetical order)
Compactness	Compactness of supervoxel used in SLIC algorithm in oversegmentation process (float)
Size	Size of supervoxel used in SLIC algorithm in oversegmentation process (integer)
Bg intens thres	Supervoxels with average intensity lower than this value will be automatically classified as background (integer)
Num hist bins	Number of histogram bins describing single supervoxel (integer)
Num neighb hist bins	Number of histogram bins calculated from voxels belonging to neighbouring supervoxels (integer)
Min major class perc	Specifies minimal percentage of majority class voxels. If a supervoxel does not have enough majority class voxels, it will not be included into training process (float number lower than 1)
Min voxel probability	Not yet used

Table 8 - Classification window controls.

Control name	Description
Input dir	Choose directory with input volumes (they will be loaded in alphabetical order)[12]
Mask dir	Choose directory with input masks (they will be loaded in alphabetical order)
Load volumes	Loads volumes and masks
Mlp path	Select trained MLP from disc
Atlasses path	Not yet used
Start	Starts classification process
Stop	Stops classification process (irreversible)

[12] It is important that inputs and masks are in corresponding alphabetical order, as the first volume is mapped to first mask, second volume to second mask etc.

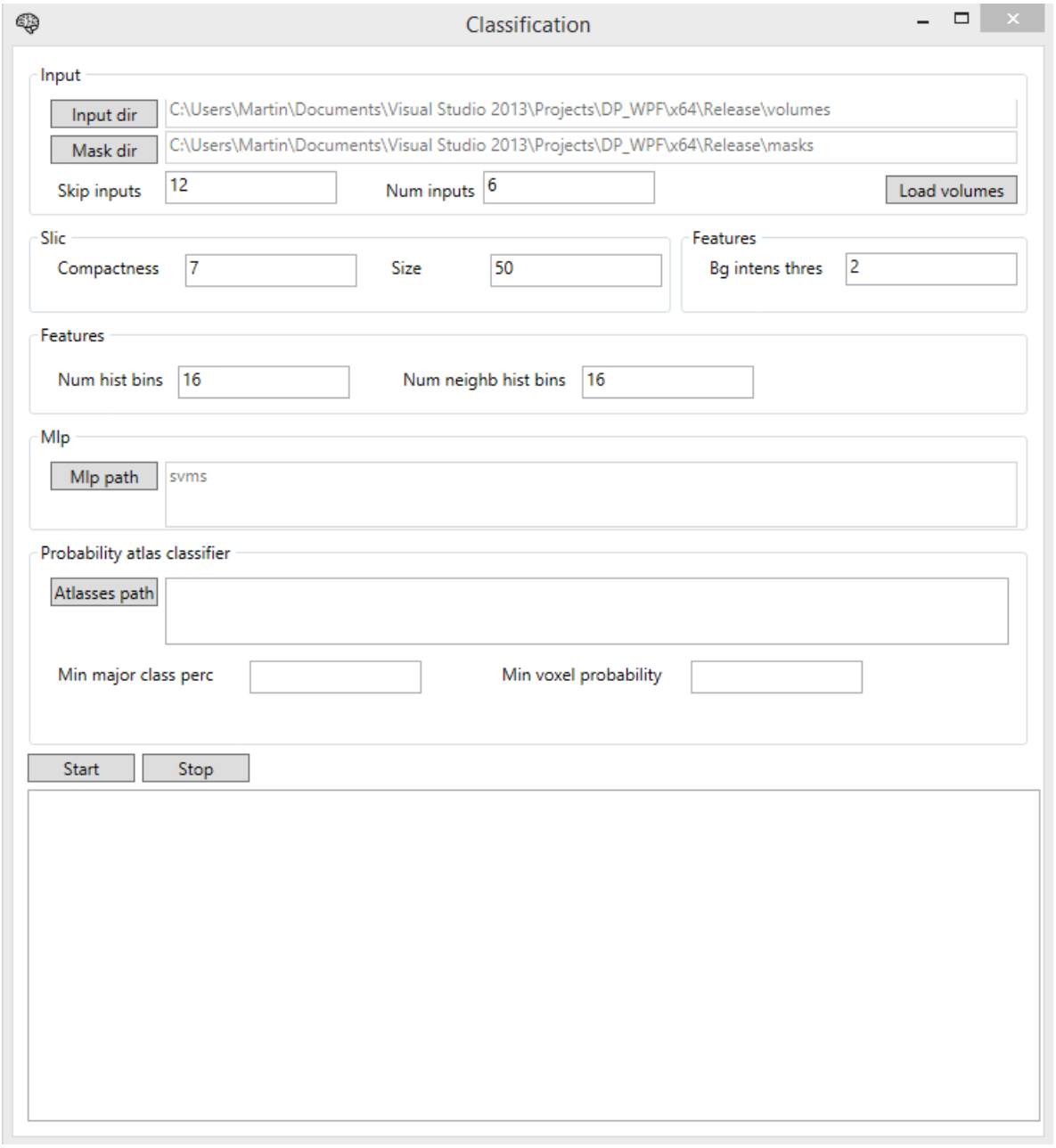


Figure 63 - Classification window.

Visualization window

In visualization window the user can load Nifti volume and visualize it using Maximum intensity projection method. User can also load segmentation mask for this volume and use it to visualize only certain tissues.

User can interact with application either in Overview window (Figure 65 left) using mouse or by setting concrete values in Visualization window (Figure 64). Interaction in Overview window is described in Table 9. Fields and controls in Visualization window are described in Table 10 and Table 11.

Table 9 - Interaction in Overview window.

Interaction	Description
Zoom	Rotate with mouse wheel
Rotation in X and Y axes	Hold left mouse key and move with mouse
Rotation in Z axis	Hold right mouse key and move with mouse
Data highlighting	Hold left ALT key and click on some voxel. If left CTRL is pressed, max value is chosen, otherwise min

Table 10 - Visualization window fields.

Field name	Description
Intensity range	Only voxels having intensity in this range will be shown (integer)
Show every nth voxel	Only every nth voxel will be shown (can boost performance) (integer)
Highlight threshold	Voxels having intensity in this range will be highlighted in Overview window and shown in Detail window (integer)
Scale	Zooms volume (float)
Voxel size	Size of single voxel (float)
Intens. add	This value will be added to intensity of every rendered voxel (integer)
Rotation X/Y/Z	Rotation in axes X, Y and Z (float)
Translation X/Y/Z	Translation in axes X, Y and Z (float)
Label to show	If "Show mask" checkbox is checked and volume mask is loaded, only voxels belonging to this class will be rendered (integer)
Show mask	If checked, only voxels belonging to particular class will be rendered
Rotation speed	Sensitivity of rotation in Overview window
Translation speed	Sensitivity of translation in Overview window

Table 11 - Visualization window controls.

Control name	Description
Pick NIFTI	Loads Nifti volume
Pick mask	Loads mask for volume that was loaded before
Visualize	Starts visualization

Information about loaded volume (and segmentation mask) is shown in Info group. The most important information is number of rendered voxels, dimensions of volume and segmentation mask and classes contained in segmentation mask.

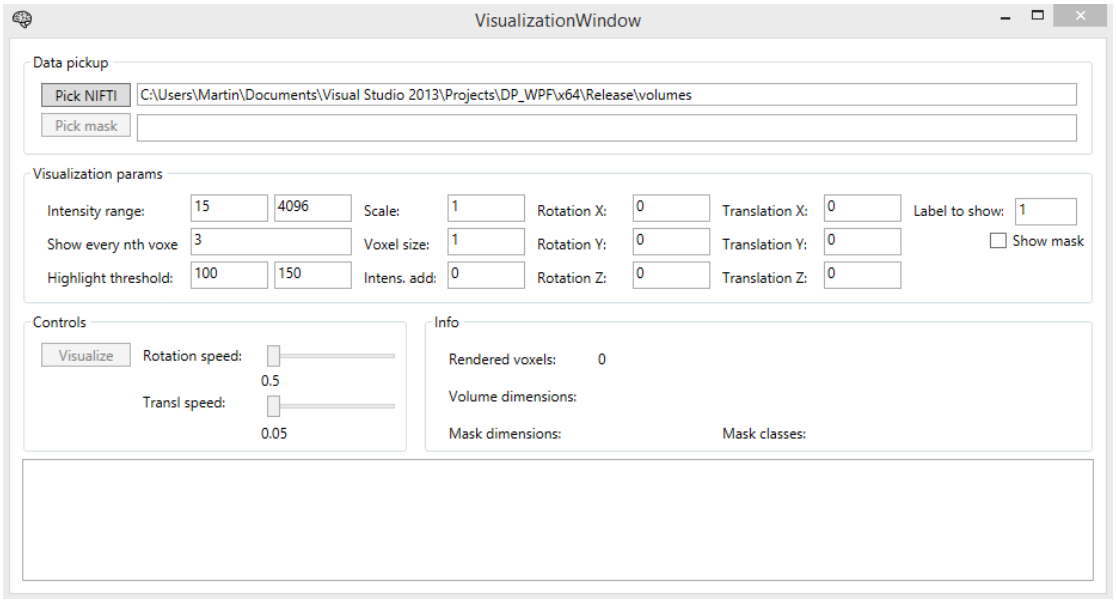


Figure 64 - Visualization window.

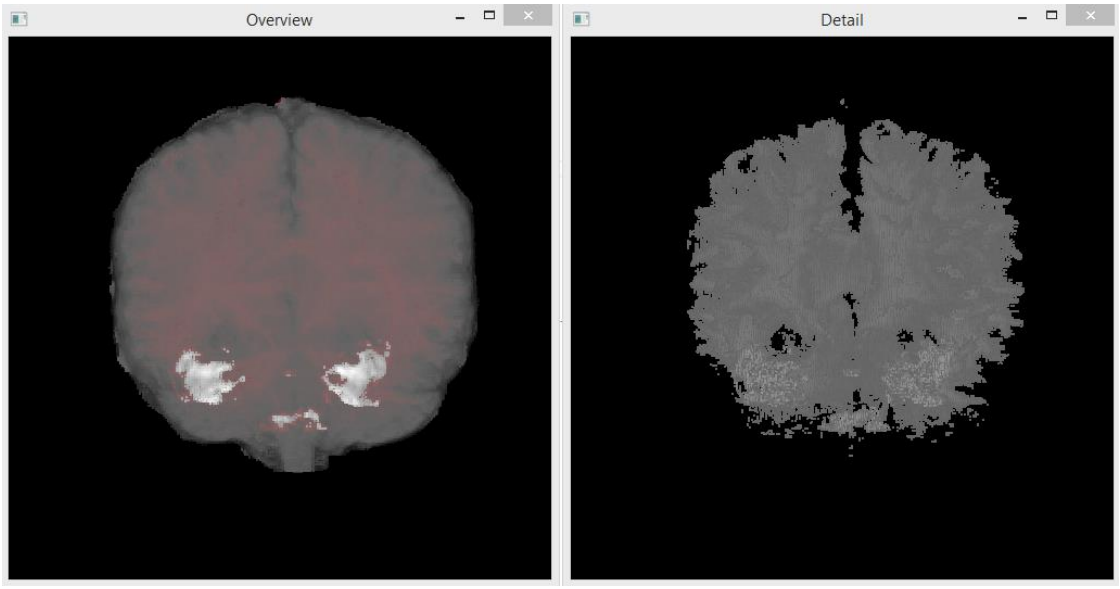


Figure 65 - Overview and Detail windows.

D. Paper accepted to IIT.SRC 2016

Segmentation of anatomical organs in medical data using supervoxels and classification

Martin TAMAJKA*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
martin.tamajka@gmail.com

Abstract. In most cases medical experts, such as radiologists, only have a certain amount of time they can spend examining patient's data. Computer aided diagnosis is a powerful instrument to accelerate this process and eliminate possible human failure. In this paper, we propose a novel approach to human organs segmentation. We focus on segmentation of grey matter (GM), white matter (WM) and cerebrospinal fluid (CSF) from brain MR volume. Our method consists of six steps and is based on oversegmentation of 3D volume to supervoxels using SLIC algorithm [1]. Next, we train MLP and SVM classifiers to classify supervoxels. Classification makes use of various features including intensity distribution, texture, shape and spatial arrangement.

Introduction

In general medical imaging grows in importance. It is more likely to cure a lethal disease if it is detected in early stage of its progression. Unfortunately there are not enough experienced radiologists which would be able to perform screening of whole population. Authors in [2] claim that radiologist needs approximately 6.83 minutes for reading single MRI. If we wanted to examine every person in Slovakia once in two years we would need 107 radiologists working 8 hours a day every single day in a year only on screening. In some countries radiologists are already assisted by computers – computer either pre-processes medical data and makes them easier to read for radiologist or highlights some possible regions of interest.

In this paper, we propose a method for segmentation of specific organ from medical 3D data that requires minimal user interaction. Although our method is going to be tested and evaluated on one specific organ – human brain, we believe it will be easy to use it to segment other organs as well.

Segmentation

According to [5], segmentation is a process of dividing digital image into regions (called segments), that share similar properties. Therefore, segment is a grouping of pixels (in 2D) or voxels (in 3D).

One of the most common and basic properties defining similarity among pixels is their intensity or grey level. Similar to intensity of pixel are its brightness and colour. More complex characteristics are based on neighbouring pixels. In order to describe texture,

---

* Master degree study programme in field: Information Systems
Supervisor: Dr. Vanda Benešová, Institute of Applied Informatics, Faculty of Informatics and Information Technologies STU in Bratislava

some authors use intensity histogram. In some cases, mutual distance of pixels in the digital image is important.

Oversegmentation and supervoxels

The most basic categorization of superpixel and supervoxel methods is:

- Graph based algorithms
- Gradient-ascend based algorithms

Point in 3D space is called voxel. When defining supervoxel, we proceed from Achanta et al. definition of superpixel qualities. Superpixel is a group of related pixels, that [3]:

- Adhere image boundaries
- When used in preprocessing, they are fast to compute, memory efficient and simple to use
- When used in segmentation, they improve speed and quality

In medical imaging, every patient's scan consists of ca. millions of points in 3D space, each having its intensity. In [4] authors process data obtained by electron microscope and segment mitochondria. Interesting is that the authors do not use voxels, but supervoxels, decreasing computational complexity by several orders of magnitude.

Segmentation in medical imaging

Segmentation in medical imaging has its particularities. As stated before, data obtained during single examination consists of millions to billions voxels which results in high computational complexity. Medical images (or volumes) have also some specific characteristics in comparison with usual images:

- Higher dynamic range
  - Common images in greyscale have 8-bit representation, medical images usually 12-bit
- Specific kinds of noise
  - E.g. partial volume effect

Especially at the boundaries, single voxel contains a mixture of tissue classes [5], becoming difficult to classify correctly (such a voxel does not even have to belong to single class).

State-of-the-art medical segmentation methods

In medical imaging, segmentation has several functions - *extraction of tissues of interest*, *preprocessing step* and *removal of non-interesting information.*

In [4] Achanta et al. used oversegmentation to segment mitochondria from EM image stack. The main contribution of oversegmentation of volume into supervoxels was rapid decrease of computational complexity.

Our reference method [6] also makes use of supervoxels in task of segmentation of brain into WM, GM and CSF. Authors describe individual supervoxels using information about intensity, shape and texture, creating 228-dimensional vector. As authors evaluate their methods (ITDS and SITDS) using the same dataset as we do, it is possible to compare our results directly. Moreover, authors also provide comparison of their method with other state-of-the-art methods, concluding their approach reaches the best results (Table 12).

Table 12 - Comparison of ITDS and SITDS with other state-of-the-art brain segmentation methods [6]

Datasets	IBSR				BrainWeb			
Methods	CSF	GM	WM	time(s)	CSF	GM	WM	time(s)
kMeans	0.51±0.06	0.75±0.06	0.78±0.04	8	0.86±0.03	0.84±0.03	0.82±0.04	12
MI	0.52±0.08	0.79±0.04	0.80±0.03	19	0.87±0.02	0.86±0.02	0.85±0.02	23
MRF	0.53±0.06	0.76±0.03	0.87±0.03	521	0.89±0.02	0.90±0.01	0.91±0.01	636

ITDS	0.60±0.05	0.81±0.03	0.86±0.02	26	0.92±0.01	0.92±0.01	0.93±0.01	32
WPNN	0.63±0.03	0.83±0.02	0.87±0.03	92	0.93±0.02	0.93±0.01	0.91±0.02	151
SITDS	0.67±0.03	0.86±0.01	0.89±0.02	29	0.94±0.01	0.95±0.01	0.94±0.01	35

Proposed method

In this paper we propose a method for organ segmentation from medical images. We focus on segmentation of brain tissue from MR image stack and its classification into four classes: {WM, GM, CSF, BG}.

Because our method mostly relies on classification, it is divided into two main phases – *Training phase* and *Classification phase*. Both phases start with the same five steps: *Data loading and conversion*, Preprocessing, *Oversegmentation*, *Identification of neighbourhoods* and *Features extraction*. Then Training phase continues with *Training of multiple classifiers* and Classification phase continues with *Classification*.

Preprocessing

There are two main motivations for preprocessing – accentuation of differences between tissues and removal of non-brain tissue.

As far as accentuation of differences between tissues we propose to use *power law transformation and brightness adjustment*. In our experiments we discovered that intensity distributions of voxels of individual tissues provide us with non-overlapping regions. As it can be seen in Figure 68, intensities of WM voxels are in general higher than those of GM.

Next preprocessing technique is removal of non-brain tissue. We decided to use Brain extraction tool (BET) [7]. There are two main requirements that we lied on this step:

1. Method must remove significant number of non-brain voxels from processed volume (eq. (1))
2. Method must not remove more than 0.25% of voxels belonging to brain (eq. (2))

$$success = 1 - \frac{|NONBRAIN - REMOVED|}{|NONBRAIN|} \quad (1)$$

$$0.0025 \leq 1 - \frac{|REMOVED \cap BRAIN|}{|BRAIN|} \quad (2)$$

Based on results in Table 13, we decided to use BET with intensity threshold equal to 0.3. Result of applying BET on brain MR volume can be seen in Figure 66.

Table 13 - Influence of parameters on BET performance (observer on first five IBSR volumes)

Intensity threshold	Equation (1) (%)	Equation (2) (%)
0.3	*2.1154*	*0.1344*
0.4	*1.3971*	*0.4118*
0.5	*0.6730*	*1.0494*
0.6	*0.4160*	*2.4955*

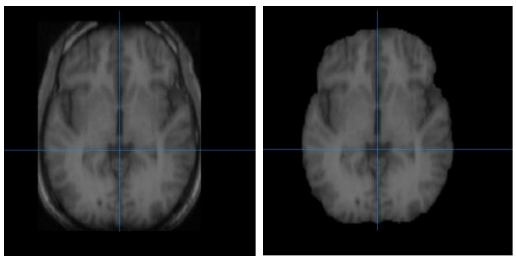


Figure 66 – Application of BET on brain MR

Oversegmentation using supervoxels

To oversegment image into supervoxels, we decided to use SLIC algorithm proposed in [1]. Supervoxels, represented by statistical data of included voxels, in its nature resist to noise better than single voxels. Our approach allows to model supervoxel in context of 3D neighbourhoods.

Success rate of oversegmentation for single supervoxel was evaluated using equation (3):

$$success = \frac{1}{N} \sum_{s \in Supervoxels} \frac{|major(s)|}{|s|} \quad (3)$$

N denotes number of supervoxels and major(s) denotes number of voxels from major class of voxels in supervoxel. We measured success rate of oversegmentation for various combinations of *compactness* and *size* parameters of SLIC algorithm. Based on measured results (Figure 67) we decided to set compactness equal to 4 and size equal to 500.

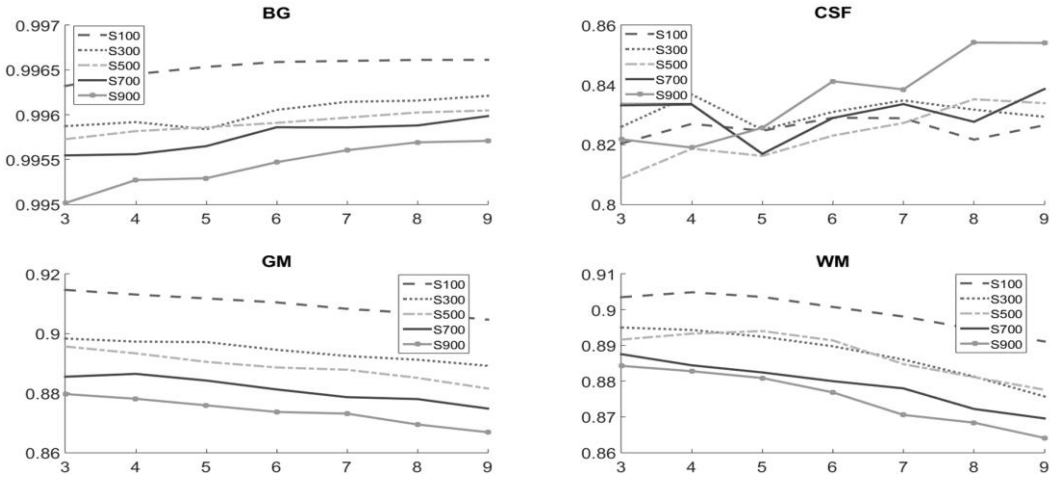


Figure 67 - Success rate of oversegmentation as a function of supervoxel compactness at fixed size (all classes)

Features extraction

Information in a supervoxel is an aggregation of information caught by contained voxels. In proposed method we describe supervoxels with following features:

- Normalized intensity histogram of voxels in supervoxel
- Normalized intensity histogram of all voxels in neighbouring supervoxels
- Normalized Euclidean distance of supervoxel centroid from the centre of the brain

Histograms of intensities describe texture and intensity distribution of supervoxel and its neighbours. According to Figure 68 it is possible to distinct between supervoxels using intensity distribution.

Location of supervoxels belonging to different tissues is typically invariant. Therefore we decided to describe supervoxels by distance of their centroids from the centre of the brain.

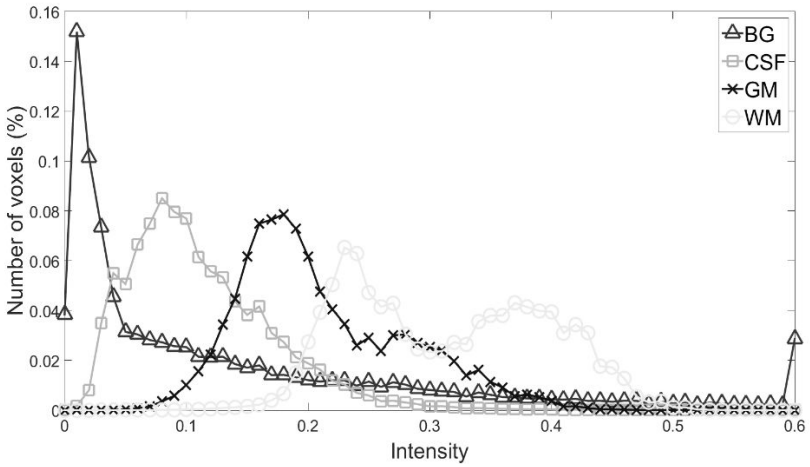


Figure 68 - Intensity distribution of WM, GM, CSF and BG in T1 MRI

Training and classification

Classification of brain tissues is not a trivial task. Therefore we decided to combine two classifiers-MLP and SVM in manner described by following pseudocode:

```
classes = ARRAY.CREATE
FOR each fv in supervoxels.features
    response = MLP.classify(fv)
    IF number of level 1 excitations in response <> 1
          PUSH(classes, combine(response, SVM.classify(fv)))
    ELSE IF number of level 2 excitations in response <> 1
          PUSH(classes, combine(response, SVM.classify(fv)))
    ELSE
          PUSH(classes, get_class(response))
    END IF
END
```

In MLP, we recognize two levels of excitation:

- **Level 1**: excitation of an output neuron is greater than 0.40
- **Level 2 (significant excitation)**: excitation of an output neuron is greater than 0.70

Success rate of MLP and SVM classification can be seen in Table 14.

Current results

Because we did not implement complete pipeline as it is defined in proposed method, we were not able to directly measure success rate of overall segmentation. This will be done in the earliest future. On the other hand, we have enough data to estimate the performance we would reach if we used MLP we trained. We also thoroughly evaluated average percentage of voxels belonging to major class in individual supervoxels. Having this information, we can combine it with success rate of classification of supervoxels of individual classes. Results can be seen in Table 14. Estimated segmentation success rate is equal to product of MLP success rate (third column) and average percentage of major class in supervoxel (second column).

Table 14 - Estimated success rates of overall segmentation for individual classes

Class	Major class voxels in superv. (%)	MLP success (%)	SVM success (%)	Estim. segm. success (%)
Background	*99.95*	*99.1*	*99.95*	98.70
White matter	*78.26*	*77.7*	*78.26*	69.54
Grey matter	*86.24*	*88.7*	*86.24*	79.65
Cerebrospinal fluid	*35.61*	*85.7*	*35.61*	70.10

Conclusion and future work

To date we proposed complex and novel method for segmentation of brain tissues. We implemented most of its steps, combining implementation in C++ and MATLAB.

We achieved promising results especially as far as segmentation of CSF is concerned. Our estimated success rate is very close to our reference method. Of course we are going to increase segmentation success rate in many different ways, including increased number of volumes used in training, evaluating of different types of MLP, investigating new features based on supervoxel shape, considering other oversegmentation algorithms. After all partial steps are completed we are going to complete overall segmentation process.

We are going to continue with evaluation. Because we find visualization at least as important as mathematical evaluation of success rate of segmentation, we are going to proceed with it, too.

Acknowledgement: This work was partially supported by the Scientific Grant Agency of the Slovak Republic, under the contract No. VEGA 1/0625/14.

References

[1] Achanta, R., Shaji, A. et al.: SLIC Superpixels. EPFL Technical Report no. 149300, June 2010.

[2] *Radiologists' reading times using PACS and using films: one practice's experience.* Fleishon, Howard B and Bhargavan, Mythreyi and Meghea, Cristian. 4, s.l. : Elsevier, 2006, Academic radiology, Vol. 13, pp. 453-460.

[3] *SLIC Superpixels Compared to State-of-the-Art Superpixel Methods.* Achanta, Radhakrishna and Shaji, Appu and Smith, Kevin and Lucchi, Aurelien and Fua, Pascal and Susstrunk, Sabine. Washington, DC : IEEE Computer Society, November 2012, IEEE Trans. Pattern Anal. Mach. Intell., Vol. 34, pp. 2274-2282. 0162-8828.

[4] *Supervoxel-based segmentation of mitochondria in em image stacks with learned shape features.* K. Smith, R. Achanta, G. Knott et al. s.l. : IEEE, 2012, Medical Imaging, IEEE Transactions, Vol. 31, pp. 474-486. 02780062.

[5] *Automated medical image segmentation techniques.* Sharma N, Aggarwal LM**.** s.l. : Medknow Publications, 2010, Journal of Medical Physics / Association of Medical Physicists of India, Vol. 35, pp. 3-14.

[6] *Discriminative Clustering and Feature Selection for Brain MRI Segmentation.* Youyong Kong, Yue Deng, Qionghai Dai. 5, s.l. : IEEE, 2015, Signal Processing Letters, Vol. 22, pp. 573-577. 1070-9908.

[7] *Fast robust automated brain extraction.* 3, s.l. : Wiley Online Library, 2002, Human brain mapping, Vol. 17, pp. 143-155.

E. Paper accepted to IWSSIP 2016

Automatic Brain Segmentation Method based on Supervoxels

Martin Tamajka[1], Wanda Benesova[2]
Faculty of Informatics and Information technologies, Slovak University of Technology in Bratislava Ilkovicova 2, 842 16 Bratislava 4, Slovakia
[1]*martin.tamajka@gmail.com*, [2]*benesova@fiit.stuba.sk*

Abstract—In this work, we present a fully automatic brain segmentation method based on supervoxels (ABSOS). We propose novel features used for classification, that are based on distance and angle in different planes between supervoxel and brain centre. These novel features are combined with other prominent features.
The presented method is based on machine learning and incorporates also a skull stripping (cranium removing) in the preprocessing step. Neural network - multilayer perceptron (MLP) was trained for the classification process. In this paper we also present thorough analysis, which supports choice of rather small supervoxels, preferring homogeneity over compactness, and value of intensity threshold parameter used in preprocessing for skull stripping. In order to decrease computational complexity and increase segmentation performance we incorporate prior knowledge of typical background intensities acquired in analysis of subjects.

Keywords—Supervoxel, brain mri segmentation, IBSR, positional feature, supervoxel classification, WM, GM, CSF.

INTRODUCTION

In radiology and neurology, segmentation of brain tissues into cerebrospinal fluid (CSF), grey matter (GM) and white matter (WM) is an important part of clinical diagnostics as it allows to extract and examine only tissues of interest. Brain segmentation is often used as a preprocessing step in medical image processing pipeline.

Input images acquired by magnetic resonance (MR) devices suffer from specific kinds of noise (e.g. periodical noise or partial volume) or other distortion. Moreover, inhomogeneity in magnetic field produced by MR causes intensity inhomogeneity where the intensity level of a single tissue class varies gradually over the extent of the image [1].

In recent years, techniques based on oversegmentation using superpixels or supervoxels have grown in importance. Supervoxels are segments in 3D space which are expected to create homogeneous regions of a given size and, despite this, the supervoxel edges should follow the natural intensity gradients in data. Supervoxels can be characterized by different kinds of statistic-based features including those based on intensity, texture, shape and position in MR volume.

In this paper, we propose a method for segmentation of brain tissue from MR image stack and its classification into four classes: {WM, GM, CSF, BG}. The proposed method, ABSOS, is a fully automatic and based on supervoxels. Supervoxels are classified using the MPL neural network and they are described by set of features *fsv* = {*Normalised intensity histogram, Normalised intensity histogram of voxels in neighbouring supervoxels, Normalized Euclidean distance from brain centre, Angles between supervoxel centroid and brain centre*}.

A. Segmentation in 3D medical imaging

Segmentation of medical volumes can be performed in two different ways: either directly in three-dimensional space or slice-by-slice, assigning a class label to each pixel in a slice.

Used similarity metrics could be based on intensity, brightness or colour and also could be extended by more complex metrics which are based on neighbouring voxels. In [2] authors use average intensity of neighbouring pixels, differences of maximum brightness values and differences of five minimum brightness values. In some cases, mutual distance of voxels is important, especially if voxels from the same class are positioned in some specific location (e.g. location of thalamus relatively to the centre of brain).

B. Oversegmentation and supervoxels

In medical imaging, a common volume consists of millions of voxels. Oversegmentation using supervoxel reduces the redundance in the data and decreases the computational complexity by several orders of magnitude. Another benefit is that supervoxels can be easily described by a set of statistical features [3].

In addition to qualities of supervoxels defined in [4] we also find homogeneity of a supervoxel important. In other words, intra-supervoxel variance should be minimized.

RELATED WORK

Almost twenty years ago authors in [5] proposed an adaptive algorithm using knowledge of tissue intensities and EM for MRI data segmentation. Since then, an appreciable amount of brain segmentation methods has been proposed. Methods are based on different segmentation techniques, such as thresholding [6], region-growing [7], edge based techniques, atlases [8] or active contour [9]. With increase of computational performance more advanced and computationally expensive techniques could be used, such as Markov Random Field (MRF) or Self-Organizing Map (SOM). From another perspective, segmentation methods and techniques can be based on statistics, prior knowledge (e.g. intensity distribution) or on their combination [10].

In [3], Lucchi et al. segmented mitochondria in electron microscopy image stacks. The authors first over-segmented 3D data and then merge these groupings (supervoxels) according to their similarity. Intensity histograms of voxels and neighbouring supervoxels were chosen as similarity criteria.

Authors in [11] used intensity histogram in combination with features that described texture and shape of a supervoxel, creating a feature vector with length 228.

Combination of prior information of relative overlap between tissue intensity distributions in MRI, spatial information and probabilistic atlas maps is the base of [10]. Authors observed that the overlap between tissue pairs is relatively stable among MR volumes and that the overlap extent differs among tissue pairs. This prior knowledge in combination with adaptive tissue priors initialized by probabilistic atlases is used in Bayesian decision theory framework. Authors in [2] segmented T2 MR images into CSF, GM, WM. Authors based their method on

1-D SOM and Adaptive Resonance Theory (ART). In this method brightness difference of brain tissues is widely used. Authors claim that brightness of pixels is the most informative property. As a feature vector not the brightness of individual pixel is used, but characteristics of whole neighbourhood of pixel, called block. Four features were proposed: brightness, average brightness in block, difference of maximum and difference of minimum.

In [9] authors incorporate region-based active contour/surface model for MR brain segmentation. Method balances between global and local intensity information. If the segment contour is close to boundary, local intensity term grows in importance. When the contour reaches boundary, it stops there. Method is evaluated on standard Brain Web[12] dataset. Authors claim promising results - Jaccard similarity coefficients for CSF, GM and WM are 0.77, 0.79 and 0.87.

Especially in last few years, many segmentation methods using oversegmentation to superpixels or supervoxels have been presented [3], [11], [13]. In May 2015 Kong et al. [11] published "SITDS", a supervised method for brain segmentation from MRI. Supervoxels obtained using SLIC were used as the unit of segmentation. "SITDS" incorporates discriminative clustering, which handles intra-class variability in order to maximize the margin among clusters and a set of initial labels. The goal of the method is to assign a label to every supervoxel. Algorithm is initialized by k-means. Next, labels are iteratively re-assigned, maximizing mutual information between labels and supervoxels.

METHOD OVERVIEW

In this paper we propose ABSOS, a method for segmentation of brain tissue from MR image stack and its classification into four classes: {WM, GM, CSF, BG}. The method is divided into two main phases – Training phase and Classification phase. Both phases start with the same five steps: Data loading and conversion, Preprocessing, Oversegmentation, Identification of neighbourhoods and Features extraction. Then Training phase continues with Training of MLP and Classification phase with Classification.

TABLE I: Intensity threshold (IT) influence on BET.

IT	Missing brain vox. (%)	Remaining non-brain vox. (%)
0.3	0.5254	2.4595
0.4	1.5555	1.4548
0.5	3.9954	0.5473
0.6	7.9358	0.2109

A. Data description

There are two dataset that are commonly used for this evaluation of performance of brain segmentation methods, IBSR[14] (real world examinations) and Brain Web [12] (synthetized). In this paper we use IBSR dataset for evaluation.

PREPROCESSING

As volumes in IBSR-18 vary in dynamic range, all data are normalized into interval [0,1] using quantile normalization. In order to avoid the usage of noisy values in normalization process, we consider all values above $Q_{0.99999}$ equal 1.

Main goals of preprocessing in our method are to increase success rate of supervoxel classification and to decrease computational complexity of training and classification. We incorporate prior

knowledge that supervoxels that have average intensity below 2 (before normalization) belong to background. Removal of skull, eyes and other non-brain tissues can dramatically decrease time needed for training and classification, too. In [6] authors proposed Brain extraction tool (BET) for this purpose]. We use a BET plugin[13] in Multiimage Analysis GUI application (Mango)[14]. This implementation has two main parameters – Intensity threshold and Threshold gradient. There are two main requirements that we lay on this step:

- Remove significant number of non-brain voxels from processed volume (maximize eq. (1))
- Must not remove more than 0.75% of voxels belonging to brain (eq. (2))

If we remove every voxel from volume, we will maximize the equation (1), which would obviously lead to volume consisting only of one segment. Therefore, we set the second requirement, which satisfaction guarantees that after this preprocessing step almost all brain voxels stay in the preprocessed volume.

$$s = 1 - \frac{|NONBRAIN - REMOVED|}{|NONBRAIN|} \quad (1)$$

$$0075 \leq \frac{|REMOVED \cap BRAIN|}{|BRAIN|} \quad (2)$$

We thoroughly evaluated influence of Intensity threshold (Threshold gradient is set to default value). Based on evaluation results (Table I) we decided to use BET with intensity threshold equal 0.3.

[13] http://rii.uthscsa.edu/mango/plugin jbet.html

[14] MangoHomepage:http://rii.uthscsa.edu/mango/

OVERSEGMENTATION USING SUPERVOXELS

To oversegment image into supervoxels, we decided to use SLIC algorithm proposed in [15] and compared SLIC to other state-of-the-art superpixel and supervoxel algorithms in [4].

Supervoxels, represented by statistical data of included voxels, in its nature resist to noise better than single voxels. In [9] authors also use neighbouring pixels to describe single pixel, but they do it in 2D space. Our approach models supervoxel in context of 3D neighbourhoods.

In ideal case a supervoxel should contain only voxels belonging to the same tissue. Success rate of supervoxelization in terms of homogeneity of supervoxels can be expressed by following equation:

$$success = \frac{1}{N} \sum_{s \in 1}^{N} \frac{major(s)}{size(s)} \quad (3)$$

where *N* is number of supervoxels, *size*(*s*) is size of the supervoxel s, *major*(*s*) is number of voxels that belong to the most occurring class in a supervoxel s. We have also evaluated eq. (3) corresponding to the classes, because the number of supervoxels belonging to background was much higher than the number of supervoxels belonging to other classes {WM, GM, CSF}.

SLIC allows to set desired size and compactness of supervoxels. The outcomes of our analysis were used for the parameters setting. For compactness equal to 6 at fixed supervoxel size reaches the success rate defined in eq. (3) its maximum. We adopted this value and

used it in oversegmentation procedure. Although very small supervoxels maximize 3, we decided to use supervoxel size 120 in order to balance oversegmentation success rate and amount of information in supervoxel. In our method supervoxel S1 and S2 is considered a neighbour of S2 if at least one voxel from S1 is in 8neighbourhood with at least one voxel from S2 in a single slice of volume.

FEATURES EXTRACTION

In case of supervoxel we can characterize its neighbours much more descriptive than in case of a voxel. We propose to describe supervoxel with following features:

- Normalized intensity histogram of voxels in supervoxel (24 bins)
- Normalized intensity histogram of all voxels in neighbouring supervoxels (24 bins)
- Normalized Euclidean distance of supervoxel centroid from the centre of the brain
- Angle between supervoxel centroid and brain centre in XY, XZ and YZ plane

If a supervoxel is surrounded by supervoxels with intensity histograms typical to some class (e.g. GM) it is a good chance that the supervoxel will also belong to the same class. On the other hand if, intensity histogram of neighbours contains intensities from all classes it is clear that the classified supervoxel lies on the boundary of some tissue.

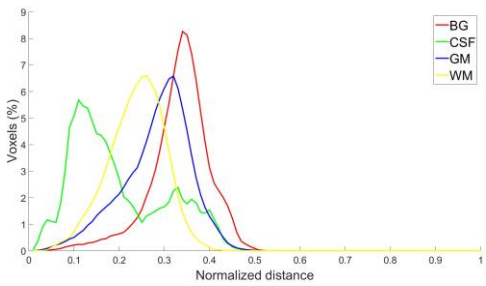


Figure 1: Distribution of normalized Euclidean distances of supervoxels of individual classes from brain centre. Supervoxels having mean intensity equal to 0 are ignored.

Normalized Euclidean distance of supervoxel centroid from the centre of the brain, is based on morphology of brain. Outer boundary of GM and CSF is typically the most distant from the centre of the brain. On the other hand, some structures belonging to CSF are closer to centre. Distributions of normalized distances between centroids of supervoxels and brain centre can be seen in Fig. 1. Distance is calculated as:

$$dist = \frac{\sqrt{(x_s - x_b)^2 + (y_s - y_b)^2 - (z_s - z_b)^2}}{norm} \quad (4)$$

norm denotes normalization term equal to:

$$norm = \sqrt{(\frac{max(x)}{2})^2 + (\frac{max(y)}{2})^2 + (\frac{max(z)}{2})^2} \quad (5)$$

x_s, y_s and z_s are coordinates of supervoxel, x_b, y_b and z_b are coordinates of brain centre, *norm* is normalization term and *max*(*x*/*y*/*z*) is maximal coordinate in MR image in a particular direction.

Many supervoxels belonging to different classes have similar distance from brain centre. Therefore, we added three angles between the centroid of supervoxel and brain centre (in XY, XZ and YZ plane). In combination with the distance from brain centre, position of supervoxel is described much more precisely and uniquely.

CLASSIFICATION

Each supervoxel is assigned to either BG, CSF, GM or WM. In training phase, we train multilayer perceptron (MLP) with two hidden layers, sigmoidal activation function and Levenberg–Marquardt training function. A significant number of supervoxels consists of voxels from even more than two different classes. In training process, we do not include supervoxels having less than 87% voxels from single class and in overall classification we do not include supervoxels having mean intensity equal less than 2 (a priori background).

VIII. Results

For training, classification and evaluation purposes, we took supervoxels from all subjects in IBSR-18 and split them to training set and testing set (80:20). Proposed method is compared with the current state-of-the-art method SITDS [11] using the same evaluation metrics – DSC. The authors compared SITDS using the IBSR-18 with other state-of-the-art methods and reported the best results. We thoroughly evaluated performance of proposed method for every tissue individually (Table II) and conclude that our results are clearly comparable to those of current state-of-the-art methods.

(a) Ground truth (b) ABSOS segmentation (c) Low MLP excitation

Figure 2: Segmentation result. (c) highlights supervoxels that excited MLP in rate lower than 0.9.

TABLE II: Performance comparison of ABSOS measured using Dice similarity coefficient (DSC).

Tissue	CSF	GM	WM
DSC	0.67	0.86	0.85

Performance of proposed method can be increased even more. In classification evaluation we observed that misclassified supervoxels tend to have bigger standard deviation, lower percentage of major class voxels and lower MLP excitation rate (Table III). Therefore, we can identify missclassified supervoxels, split them and classify individually. We assume that such supervoxels will be more homogeneous.

TABLE III: Correctly classified supervoxels have greater MLP excitation and major class percentage. Contrary, intensity standard deviation is lower among them.

	Intens. stand. dev.	Major class perc.	MLP excit. rate
Correct	8.1692	92.67%	0.9883
Missclass	10.0361	72.50%	0.8964

We also trained second MLP using all supervoxels from subjects 3-15 of IBSR-18. Subsequently, we segmented subject 3 from IBSR-18. Results can be seen in Fig. 2.

CONCLUSION

In this paper we propose a fully automatic method for segmentation of brain from MR images, ABSOS. Supervoxels are classified into four classes {BG, CSF, GM, WM} and they are described by set of features *fsv* = {*Normalised intensity histogram, Normalised intensity histogram of voxels in neighbouring supervoxels, Normalized Euclidean distance from brain centre, Angles between supervoxel centroid and brain centre*}. As shown in Table II, our results are promising in context of current state-of-the-art methods and proposed method (as is) is going to be subject of further research.

Supervoxels with higher oversegmentation error (which tend to be misclassified, too) have higher standard deviation and lower MLP excitation rate. We are going to use this information to identify potentially oversegmented and/or misclassified supervoxels and either split them into smaller supervoxels (which will be classified individually) or use some other segmentation technique, e.g. majority voting using nonrigidly registered atlases. Next option is to use and evaluate performance of another oversegmentation algorithm.

ACKNOWLEDGMENT

Acknowledgement: This work was supported by the Grant VEGA 1/0625/14 and Siemens Healthcare Bratislava.

REFERENCES

[1] N. Sharma, L. M. Aggarwal *et al.*, "Automated medical image segmentation techniques," *Journal of medical physics*, vol. 35, no. 1, p. 3, 2010.

[2] K. Sato, S. Kadowaki, H. Madokoro, M. Ito, and A. Inugami, "Unsupervised segmentation for mr brain images," in *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, ser. ISABEL '11. New York, NY, USA: ACM, 2011, pp. 44:1–44:5. [Online]. Available: http://doi.acm.org/10.1145/2093698.2093742

[3] A. Lucchi, K. Smith, R. Achanta, G. Knott, and P. Fua, "Supervoxelbased segmentation of mitochondria in em image stacks with learned shape features," *IEEE Transactions on Medical Imaging*, vol. 31, no. 2, pp. 474–486, Feb 2012.

[4] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov 2012.

[5] W. Wells, W. Grimson, R. Kikinis, and F. Jolesz, "Adaptive segmentation of mri data," *IEEE Transactions on Medical Imaging*, vol. 15, no. 4, pp. 429–442, Aug 1996.

[6] S. M. Smith, "Fast robust automated brain extraction," *Human brain mapping*, vol. 17, no. 3, pp. 143–155, 2002.

[7] A. M. Alyassin and G. B. Avinash, "Semiautomatic bone removal technique from ct angiography data," vol. 4322, 2001, pp. 1273–1283. [Online]. Available: http://dx.doi.org/10.1117/12.431005

[8] X. Artaechevarria, A. Munoz-Barrutia, and C. O. de Solorzano, "Combination strategies in multi-atlas image segmentation: Application to brain mr data," *IEEE Transactions on Medical Imaging*, vol. 28, no. 8, pp. 1266–1277, Aug 2009.

[9] L. Wang, C. Li, Q. Sun, D. Xia, and C.-Y. Kao, "Brain mr image segmentation using local and global intensity fitting active contours/surfaces," in *Medical Image Computing and Computer-Assisted Intervention– MICCAI 2008*. Springer, 2008, pp. 384–392.

[10] N. Verma, G. S. Muralidhar, A. C. Bovik, M. C. Cowperthwaite, M. G. Burnett, and M. K. Markey, "Three-dimensional brain magnetic resonance imaging segmentation via knowledge-driven decision theory," *Journal of Medical Imaging*, vol. 1, no. 3, p. 034001, 2014. [Online]. Available: http://dx.doi.org/10.1117/1.JMI.1.3.034001

[11] Y. Kong, Y. Deng, and Q. Dai, "Discriminative clustering and feature selection for brain mri segmentation," *IEEE Signal Processing Letters*, vol. 22, no. 5, pp. 573–577, May 2015.

[12] C. A. Cocosco, V. Kollokian, R. K.-S. Kwan, G. B. Pike, and A. C. Evans, "Brainweb: Online interface to a 3d mri simulated brain database," *NeuroImage*, vol. 5, p. 425, 1997.

[13] N. Verma, M. C. Cowperthwaite, and M. K. Markey, "Superpixels in brain mr image analysis," in *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, July 2013, pp. 1077–1080.

[14] T. Rohlfing, "Image similarity and tissue overlaps as surrogates for image registration accuracy: Widely used but unreliable," *IEEE Transactions on Medical Imaging*, vol. 31, no. 2, pp. 153–163, Feb 2012.

[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels," Tech. Rep., 2010.

F. Resume in Slovak language

Cieľ práce

Cieľom tejto práce je navrhnúť, implementovať a overiť metódu na automatickú segmentáciu anatomických orgánov v medicínskych dátach. My sme sa zamerali na segmentáciu mozgu v MR dátach do štyroch tried – pozadie (BG), mozgovomiechový mok (CSF), šedá hmota (GM) a biela hmota (WM).

Analýza problému a domény

Doména

Pri špecifických druhoch medicínskych zákrokov je nutné vopred získať informáciu o vnútornom stave pacienta. Ideálne je pritom použitie neinvazívnych metód, ako sú CT, MR, PET, RTG alebo ultrazvuk, prípadne ich kombinácií. V tejto práci sme sa zamerali na dáta pochádzajúce z magnetickej rezonancie (MR).

Dáta z magnetickej rezonancie majú formu trojdimenzionálneho hustého objemu, kde každý bod má okrem troch súradníc taktiež intenzitu. Nakoľko sme chceli úspešnosť našej metódy porovnať so súčasnými modernými riešeniami prostredníctvom tých istých dát, použili sme štandardnú dátovú množinu IBSR [33].

Cieľov metód spracuvávajúcich medicínske dáta je viacero. Prvým je zefektívniť prácu špecialistov, ktorí vďaka týmto metódam potrebujú menej času na vyšetrenie jedného pacienta. Okrem toho existujú metódy, ktoré majú znížiť riziko, že expert prehliadne nejaký dôležitý aspekt pri vyšetrení. Príkladom sú metódy z rodiny CAD (computer aided diagnosis).

Teoretický základ

Pod pojmom **segmentácia** sa rozumie rozdelenie digitálneho obrazu (v tomto prípade trojrozmerného objemu) na regióny, v rámci ktorých voxely zdieľajú podobné vlastnosti.

Existuje viacero vlastností, na základe ktorých je možné zadefinovať podobnosť medzi voxelmi. Prvou z nich je intenzita, ktorá je zároveň základom jednej z najjednoduchších segmentačných techník – prahovania. Podobnou vlastnosťou je farba. Na rozdiel od intenzity môže byť farba voxelu definovaná v rôznych tzv. farebných priestoroch. Nami využívaný supervoxelizačný algoritmus SLIC využíva farebný priestor CIELAB, ktorý je považovaný za najbližší ľudskému vnímaniu farebných rozdielov.

Komplexnejšie charakteristiky sú založené na susediacich voxeloch. Týmito môže byť priemerná intenzita susediacich voxelov, rozdiel medzi najsvetlejším a najtmavším voxelom či histogram intenzít.

Podobnosť voxelov nemusí byť určovaná len na základe ich intenzity alebo farby. Mierne modifikovaný zhlukovací algoritmus K-Means použitý v algoritme SLIC využíva na meranie podobnosti medzi voxelmi ich vzájomnú euklidovskú vzdialenosť. Vzdialenosť je pritom možné zadefinovať rôzne, či už ako kosínusovú, manhattanskú alebo inú.

Posledná veľká skupina charakteristík je založená na opise tvaru objektu.

Existujú viaceré segmentačné metódy. Najjednoduchšími sú prahovanie a metóda rastúcich regiónov. Okrem toho existujú metódy založené na hranách, klasifikácií alebo apriórnej vedomosti. Prahovanie je založené na rozdelení obrázku na popredie a pozadie, pričom voxely majúce intenzitu (farbu) väčšiu ako stanovený prah patria do popredia a zvyšne do pozadia. Sú možné rôzne modifikácie prahovania. Metóda rastúcich regiónov vyžaduje inicializáciu počiatočným semiačkom, od ktorého v každom smere región rastie až kým nie je splnená ukončovacia podmienka. Tou môže byť maximálna veľkosť regiónu, hrana alebo iná podmienka. Klasifikačné metódy opisujú voxel sadou príznakov, na základe ktorých je následne voxel priradený do niektorej z tried. Tieto príznaky sú typicky odvodené na základe okolia voxelu.

Pod pojmom **presegmentovanie** sa rozumie rozdelenie obrazu do viacerých regiónov než ako je v ňom objektov reálneho sveta. V rámci tejto práce využívame presegmentovanie obrazu vo forme **supervoxelov**, na ktoré je tento obraz rozdelený. Supervoxely by pritom mali spĺňať nasledovné charakteristiky [8]:

- Mali by priliehať k hranám.
- Keď sú použité v rámci predspracovania, mali by byť jednoduché na výpočet, pamäťovo nenáročné a jednoduché na použitie.
- Keď sú použité pri segmentácií, mali by zvýšiť jej kvalitu a rýchlosť.

Podľa nás je taktiež dôležité, aby supervoxely boli čo najviac homogénne. Existujú viaceré supervoxelizačné algoritmy. My sme sa rozhodli použiť SLIC, keďže sa jednoducho používa a okrem toho produkuje veľmi dobré výsledky.

Naša metóda je okrem presegmentovania založená na **klasifikácií**. Analyzovali sme viacero klasifikátorov a rozhodli sme sa využiť doprednú **neurónovú sieť**. Neurónová sieť pozostáva z neurónov, ktoré sú usporiadané vo vrstvách. Neuróny medzi jednotlivými vrstvami sú navzájom pospájané váhovanými spojeniami. Prostredníctvom týchto spojení sa zo vstupných neurónov prenáša excitácia na výstupné.

Segmentácia medicínskych dát má svoje špecifiká. Tými hlavnými sú vyšší dynamický rozsah takýchto dát (12 bitov) a špecifické druhy šumu. Typickým šumom v MR dátach je tzv. efekt čiastočného objemu (angl. partial volume effect), kedy sa v jednom voxeli agreguje informácia z viacerých rôznych tkanív. Efekt čiastočného

objemu sa vyskytuje najmä na rozhraniach medzi tkanivami. Okrem toho sa aj v nami použitých dátach vyskytuje periodický šum, efekt rozvlnenia (angl. ringing effect) a iné.

Súčasné riešenia

Presegmentovanie veľkoobjemových dát bolo použité už vo viacerých vedeckých prácach. V [11] Achanta využil supervoxely pri segmentácií mitochondrií v dátach získaných elektrónovým mikroskopom. Autori práce tvrdia, že sa im podarilo znížiť výpočtové nároky segmentácie o niekoľko rádov. Zároveň je nutné podotknúť, že autori, podobne ako my, použili histogram intenzít supervoxelu na jeho opis.

Autori v [27] (našej referenčnej metóde) segmentovali mozog do tried pozadie, mozgovomiechový mok, šedá hmota a biela hmota. Použili pritom tie isté dáta ako my a preto bolo možné s touto metódou priamo porovnať úspešnosť našej metódy. V práci sú predstavené dve metódy založené na presegmentovaní za použitia supervoxelov, pričom prvá z nich (ITDS) nevyžaduje učiteľa a druhá (SITDS) áno. Lepšie výsledky, ktoré prekonali aj výsledky iných moderných metód, dosiahla SITDS.

Okrem segmentácie založenej na klasifikácií či zhlukovaní boli navrhnuté aj metódy využívajúce apriórnu znalosť. Príkladom je distribúcia intenzít jednotlivých tkanív či tvar regiónov ohraničujúcich tieto tkanivá.

Návrh metódy a opis riešenia

Vstupné dáta

Existuje viacero štandardných formátov na uchovávanie a spracovanie medicínskych dát. Najpoužívanejším a zároveň najrobustnejším je DICOM, ktorý okrem samotného formátu dát opisuje aj infraštruktúru a protokoly ich prenosu. Starším formátom je Analyze. Formát Minc je používaný zväčša lokálne.

Formátom, ktorému je prispôsobená a s ktorou pracuje naša metóda, je **Nifti**. Okrem samotných dát Nifti uchováva aj metadáta opisujúce kontext v ktorom boli dáta získané a tiež orientáciu týchto dát. Vo všeobecnosti sa v medicínskej praxi používa viacero orientácií tých istých dát, pričom odborníci z rôznych oblastí používajú rôzne orientácie. V našom prípade pred samotným spracovaním dát tieto najprv transformujeme do orientácie RSA (right-superior-anterior). V zásade je jedno, v akej orientácií sa budú spracovávané dáta nachádzať, dôležité je len to, aby boli všetky v rovnakej.

IBSR dataset obsahuje dáta z vyšetrenia osemnástich subjektov T1 MR zariadením s rozlíšením 256x256x128. Každý subjekt obsahuje okrem samotného MR objemu taktiež trojrozmernú segmentačnú masku mozgu a aj jednotlivých tkanív. Táto maska bola použitá pri tréningu a taktiež pri vyhodnotení ako zlatý štandard.

Prehľad metódy

Naša metóda pozostáva z dvoch fáz – *Tréning* a *Klasifikácia*. Obe začínajú tými istými piatimi krokmi: *Načítanie a konverzia dát*, *Predspracovanie*, *Presegmentácia* (supervoxelizácia), *Identifikácia susedstiev* a *Extrakcia príznakov*. Následne tréning pokračuje s *Trénovaním klasifikátora* a klasifikácia s *Klasifikáciou*.

Tréning

V prvom kroku je nutné **načítať dáta**. Ako už bolo spomenuté, pracujeme s Nifti dátami, ktoré pred spracovaním orientujeme do RSA orientácie.

Predspracovanie pozostáva z dvoch krokov. Rôzne subjekty mávajú rôzny dynamický rozsah, keďže boli zaznamenané na rôznych MR zariadeniach. Z tohto dôvodu je nutné **normalizovať intenzity** jednotlivých voxelov do intervalu *[0; 1]*. Intenzity normalizujeme kvantilovou normalizáciou, pričom hodnoty väčšie ako $Q_{0.99999}$ kladieme rovné 1. Ďalším krokom je odstránenie voxelov, ktoré netvoria mozgové tkanivo. Za týmto účelom využívame metódu BET (Brain Extraction Tool). Cieľom v tomto kroku bolo pre nás odstrániť čo možno najviac voxelov netvoriacich mozgové tkanivo a zachovať viac ako 99.25% voxelov, ktoré ho tvoria. Parametrom metódy BET, ktorého vplyv na uvedené dve podmienky sme skúmali, bol **prah intenzity** (Intensity Threshold). Po dôkladnej analýze **sme hodnotu 0.3 vyhodnotili ako najvhodnejšiu** a následne sme ju použili pri extrakcií mozgu. Platí pritom, že so stúpajúcou hodnotou prahu intenzity sa darí odstraňovať viac voxelov nepatriacich mozgu, no zároveň je odstraňovaných viac voxelov, ktoré mozgu patria.

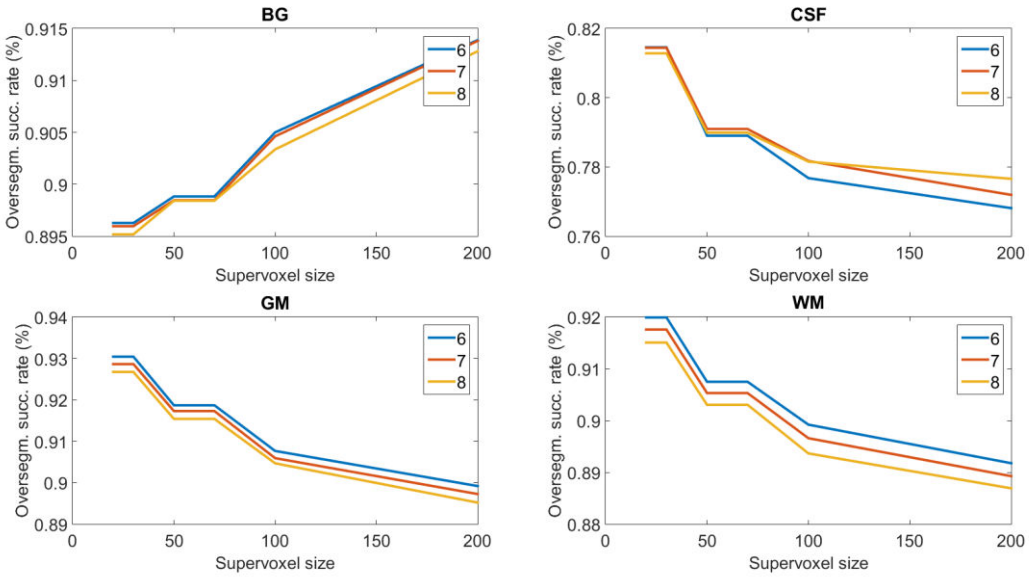
Pri **presegmentovaní** objemu **do supervoxelov** sme hneď na začiatku stanovili mieru úspešnosti takejto segmentácie, vzhľadom na ktorú sme sa snažili nájsť ideálne hodnoty parametrov *veľkosť* a *kompaktnosť* algoritmu SLIC. Mieru úspešnosti definujeme ako

$$success = \frac{1}{N} \sum_{s \in Supervoxels} \frac{major(s)}{size(s)}$$

Význam použitých symbolov je nasledovný:
N – Počet supervoxelov
s – Jeden supervoxel
$size(s)$ – Počet voxelov obsiahnutých v konkrétnom supervoxeli
$major(s)$ – Počet voxelov triedy s najväčším výskytom v jednom supervoxeli

Okrem celkovej úspešnosti presegmentácie sme vyhodnotili aj úspešnosť pre jednotlivé triedy, nakoľko početnosť supervoxelov patriacich do jednotlivých tried sa medzi triedami výrazne odlišuje. Najmenej početnými boli tie patriace do triedy mozgovomiechový mok, najpočetnejšími tie patriace do pozadia. Vo všeobecnosti bolo

možné sledovať trend, že so stúpajúcou veľkosťou a kompaktnosťou supervoxelu miera úspešnosti presegmentácie klesala. Preto by sa mohlo zdať, že ideálne by bolo zvoliť veľmi malé a veľmi nekompaktné supervoxely. Na tomto mieste však treba vzhliadnuť na fakt, že malé supervoxely poskytujú menšie množstvo údajov, ktoré by ich umožnili opísať. Zároveň prílišná tvarová rôznorodosť supervoxelov môže nepriaznivo vplývať na kvalitu opisu lokalizácie supervoxelu vzhľadom na centrum mozgu. Z tohto **dôvodu sme sa rozhodli použiť supervoxeli s veľkosťou približne 120 voxelov a kompaktnosťou s hodnotou 6**. Vzťah medzi veľkosťou, kompaktnosťou a úspešnosťou pri fixnej kompaktnosti a variabilnej veľkosti je možné vidieť na Obrázok 1.



Obrázok 1 - Úspešnosť presegmentácie ako funkcia veľkosti supervoxelu.

Po presegmentovaní je nutné **identifikovať susedstvá medzi supervoxelmi**. Dva supervoxely S1 a S2 považujeme za susediace vtedy a len vtedy, keď S1 obsahuje voxel, ktorý je vo vzájomnej 8-susednosti s aspoň jedným voxelom zo supervoxelu S2 v niektorom reze objemu.

Pre účely klasifikácie je nutné **opísať jednotlivé supervoxely sadou príznakov**. Ako už bolo spomenuté, väčšie supervoxely sú viac samoopisné ako tie malé. V tejto práci charakterizujeme supervoxely nasledujúcou sadou príznakov:

- Normalizovaný histogram intenzít všetkých voxelov v supervoxeli (24 uniformných intervalov).
- Normalizovaný histogram intenzít všetkých voxelov všetkých susediacich supervoxelov (24 uniformných intervalov).
- Normalizovaná euklidovská vzdialenosť centroidu supervoxelu od centra mozgu.
- Uhol medzi centroidom supervoxelu a centrom mozgu v rovinách XY, XZ a YZ.

Predpoklad, že histogramy intenzít budú schopné aspoň do určitej miery rozlíšiť medzi supervoxelmi rôznych tried, vychádza z faktu, že distribúcie intenzít voxelov jednotlivých tried sa navzájom odlišujú.

Opis supervoxelov prostredníctvom normalizovanej euklidovskej vzdialenosti ich centier a centra mozgu vychádza z morfológie mozgu. Vonkajšie okraje šedej hmoty a bielej hmoty sú typicky v najväčšej vzdialenosti od centra mozgu. Naopak, niektoré štruktúry mozgovomiechového mozgu sú k centru mozgu najbližšie. Nemalo by zmysel extrahovať absolútne vzdialenosti, nakoľko rozlíšenia jednotlivých MR zariadení sú rôzne. Preto túto euklidovskú vzdialenosť normalizujeme.

Viaceré supervoxely majú takmer rovnakú vzdialenosť od centra mozgu. Preto sme pridali opis polohy prostredníctvom uhlov medzi centrom mozgu a centroidom supervoxelu v rovinách XY, XZ a YZ. Vďaka tomu je ich poloha charakterizovaná viac-menej jednoznačne.

Supervoxely, charakterizované uvedenými príznakmi, následne slúžia na **natrénovanie klasifikátora**. V tejto práci využívame doprednú neurónovú sieť (viacvrstvový perceptron) s dvomi skrytými vrstvami, sigmoidálnou aktivačnou funkciou a Levenberg–Marquardt trénovaciou funkciou. Do procesu trénovania nezahŕňame dve skupiny supervoxelov:

1. Supervoxely, ktoré mali pred normalizáciou priemernú intenzitu menšiu ako 2. Tieto sú automaticky považované za pozadie.
2. Supervoxely, kde početnosť voxelov majoritnej triedy je menšia ako 87%. Tieto supervoxely zväčša nie je možné presne klasifikovať a preto nemá zmysel ich zahŕňať do trénovacej množiny.

Takouto filtráciou trénovacích dát sme chceli dosiahnuť zrýchlenie trénovania a tiež zvýšenie presnosti klasifikácie.

Vo fáze klasifikácie je prvých päť krokov zhodných s fázou trénigu. Jednotlivé supervoxely sú klasifikované na základe ich príznakov. Všetkým voxelom v klasifikovanom supervoxeli je priradené návestie (angl. label) zhodné s výsledkom klasifikácie supervoxelu. Na tomto mieste môžu vzniknúť dve chyby:

1. Celý supervoxel bude chybne klasifikovaný, t.j. nebude priradený do triedy, z ktorej pochádza majoritná časť obsiahnutých voxelov.
2. Supervoxel bude správne klasifikovaný, no obsahuje voxely z viacerých tried. V takomto prípade bude niektorým voxelom nutne priradené nesprávne návestie.

Základ pre vznik druhej chyby bol pritom položený už pri presegmentácií, kedy boli voxely z viacerých tried zaradené do jedného supervoxelu.

Podobne ako pri tréningu, ani pri klasifikácií sa nezaoberáme supervoxelmi, ktoré majú pred normalizáciou priemernú intenzitu menšiu alebo rovnú 2. Tieto sú automaticky považované za pozadie.

Pri vyhodnocovaní sme porovnali štandardnú odchýlku, percentuálne zastúpenie majoritnej triedy a mieru excitácie perceptronu pri správne a nesprávne klasifikovaných supervoxeloch. Je možné vidieť (Table 4), že správne klasifikované supervoxely majú výrazne vyššiu mieru excitácie a percentuálne zastúpenie majoritnej triedy. Okrem toho majú správne klasifikované supervoxely nižšiu štandardnú odchýlku.

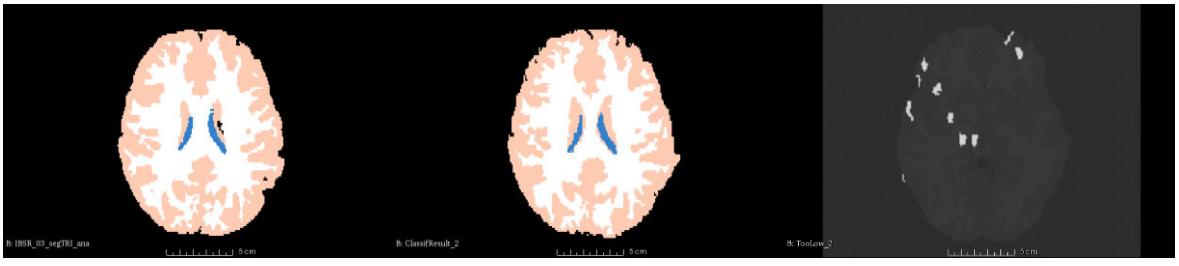
Realizácia riešenia

Riešenie sme realizovali v jazykoch C++ a C#, pričom časť prototypovania a vyhodnocovania výsledkov sme vykonali v prostredí MATLAB. Celé riešenie je z fyzického hľadiska rozdelené do viacerých projektov a z konceptuálneho do viacerých modulov. Projektmi sú DP_CLR, DP_CS, DP_Visualization, DP_WPF_prototype, SLICSuperpixels a modulmi *Klasifikácia*, *Konverzia*, *Načítavanie*, *Spracovanie*, *Serializácia*, *Štatistika*, *Vizualizácia* a *GUI*. Okrem nich sme vyčlenili samostatné pomocné moduly *Pomocné programy* a *Pomocníci*.

Dosiahnuté výsledky

Natrénovali sme viacvrstvový perceptron s dvomi skrytými vrstvami (52 a 8 neurónov). Pri vyhodnocovaní sme rozdelili dátovú množinu v pomere 80:20 (trénovacia:testovacia) na úrovni supervoxelov, t.j. každý piaty supervoxel sme zaradili do testovacej množiny. Úspešnosť segmentácie sme merali **Diceovým koeficientom podobnosti**. Dosiahli sme výsledky jasne porovnateľné so súčasnou modernou metódou SITDS. Konkrétne hodnoty: **mozgovomiechový mok 0.67, šedá hmota 0.86, biela hmota 0.85**.

Okrem toho sme natrénovali klasifikátor na subjektoch 4-15 z dátovej množiny IBSR-18 a následne sme segmentovali subjekt 3. Výsledky je možné vidieť na Obrázok 2.



(a) Zlatý štandard (b) Navrhovaná metóda (c) Nízka excitácia MLP

Obrázok 2 - Výsledok segmentácie. Na (c) sú vyznačené supervoxely, pri ktorých excitácia v MLP nedosiahla hodnotu 0.9.

Zhodnotenie a budúca práca

Podarilo sa nám navrhnúť, implementovať a overiť metódu na automatickú segmentáciu mozgu z MR dát za použitia supervoxelov. Dosiahli sme výsledky porovnateľné so súčasnými modernými metódami.

V budúcnosti by sme sa chceli zamerať na zvýšenie úspešnosti segmentácie. Chceme pri tom využiť fakt, že neurónová sieť dosahuje nižšiu úroveň excitácie pri nesprávne klasifikovaných supervoxeloch, a taktiež že supervoxely s vyššou štandardnou odchýlkou v rámci intenzity sú častejšie nesprávne klasifikované. Okrem toho chceme vyskúšať aplikovať iný presegmentačný algoritmus, čím by sme chceli znížiť chybu v tomto kroku.