Universiteit Leiden
Opleiding Informatica

Unsupervised Segmentation and Texture Visualization of 3D Medical Images using Textural Features

Name: Theodoros Georgiou

Date: 29/07/2016

1st supervisor: Dr. Erwin M. Bakker
2nd supervisor: Dr. Michael Lew

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Acknowledgments

I would like to thank my first thesis supervisor Dr. Erwin M. Bakker from LIACS for all his help with this project. He was the one with the initial idea for this project and without him I would not have started it. He has always answered my questions and helped me keep a strict direction for the project. He arranged frequent meetings, checked my progress and provided consultation on how I should proceed. He also took much time to provide me with reliable datasets that were key to getting the project to an end. I would like to thank him for all his effort on helping me and the project.

I would also like to thank my second thesis supervisor Dr. Michael Lew from LIACS, for his assistance through the evaluation of my thesis.

Contents

Abstract

In this paper we implement and compare textural analysis methods, for 3D medical image segmentation and texture visualization. We implement and evaluate an unsupervised system that makes use of these features in order to automatically segment 3D medical images. There are many algorithms that try to extract features that represent the texture of an image. Moreover due to technological advancement, the use of 3D imaging for medical purposes is wide spread. Thus there is a need for robust automated methods that segment those images. Due to the different nature of different imaging techniques (MRI, Ultra Sound, CT et cetera) the resulted images have different characteristics and produce different textures. Thus for every application and image acquiring technique, different features work. In this paper, we use features that describe 3D texture and feature selection techniques in order to create a general method to segment 3D medical images. The algorithms developed in this project are evaluated on MRI and Ultra Sound images. Moreover, we show the importance of the produced clustering in the visualization of the different textures in an image.

1 Introduction

Accurate image segmentation is a very important step in medical image analysis [1], since it is usually required for computer-aided diagnosis. For example it is very important for detecting tumors, edema and necrotic tissues [2]; quantification of white matter lesions is important for drug treatment estimation in Multiple Sclerosis [3]; volumetry of gray matter, white matter and cerebro-spinal fluid is used for characterization of morphological differences between subjects with schizophrenia and epilepsy. Magnetic Resonance imaging (MRI), is a very powerful diagnostic imaging technique, since it has high contrast resolution for different tissues [2].

For applications, such as prostate cancer segmentation, computer-aided diagnosis and therapy planning, breast cancer segmentation and more, Ultrasound images are used. Although Ultrasound images suffer from speckle noise, and low-contrast organ tissues, they are widely used, for several reasons. The hardware needed to obtain Ultrasound images is much smaller in size than other methods (MRI and CT scanners) and the cost to acquire it, as well as obtain images, is much lower. Thus Ultrasound is a much more accessible technique. Moreover, since it is not ionizing, it is not invasive to patients like other methods.

There are two types of image segmentation, manual segmentation and automated segmentation. In manual segmentation a trained expert classifies each voxel of the image. This procedure is very time consuming, and prone to errors [1]. Thus researchers turn their focus to automated methods. Supervised segmentation methods are in need of datasets with good ground truth in order to be able to segment new images and as a result, they rely on manual segmentation. Since only trained experts are able manually segment these images accurately, building such a data set is very time consuming. In recent years there has been an effort of researchers to build big datasets with well defined ground truth so that methods can be universally compared [64, 65]. Unfortunatelly, these datasets have restricted access, since they can only be used for the purposes of specific chalenges for which they are developed.

In this paper, an unsupervised method that tries to segment 3D medical images is proposed and tested on two MRI datasets and an Ultrasound dataset. One MRI dataset is (BrainWeb [58]) is composed by simulated images, and one by real MR Images (IBSR [63]). These two datasets are commonly used for white matter, gray matter and cerebro-spinal fluid segmentation. The BrainWeb is used because since the images are simulated, they have excellent

ground truth. The second dataset has provided manual segmentation. Although many researchers point out that the ground truth is not very accurate [1, 7], it is still being used for method comparison and evaluation, since real datasets with ground truth provided are very difficult to acquire. The Ultrasound data set is constructed for the purpose of this project. It contains images with agar-agar with flower texture in different concentrations. Since agar-agar scans simulate the texture of different tissues accurately [67], being able to segment those images shows the capabilities of the method to differentiate different tissues on a real scan.

1.1 MRI

1.1.1 Methods, Applications and Challenges

The are a few challenges that MRI segmentation methods have to overcome. These include, but are not limited to, the inhomogeneity of MR Images (or bias field), which is a low frequency noise, which slowly changes the average gray level intensity of tissues within the image. Moreover, many methods suffer from different kind of deformations of the brain of different subjects. This affects mostly methods that take into account the spatial position of a voxel to derive tissue probabilities, based on historical data. One more difficulty that such methods have to overcome is spatial movement of the subject, which results in misaligned consecutive 'slices' of two dimensional images.

There are many studies that try to do automatic MRI segmentation. Mainly they fall into five categories [1], namely:

1) Manual Methods

2) Intensity Based Methods

3) Atlas-Based Methods

4) Surface Based Methods

5) Hybrid Methods

Manual Segmentation is the oldest type of segmentation methods, where trained professionals try to classify each voxel in the image depending on their expertise, often aided by imaging tools. Usually manual segmentation is done in a 'slice-by-slice' manner [1]. Since this is very time consuming, researches try to create automated methods to do accurate segmentation of 2D and 3D MR Images. Although there is a lot of inter and intra-observer variability on the produced segmentation [3], manual segmentation is still considered as 'golden standard' and it is used in order to validate and compare automated methods [1].

There are various types of Intensity based methods. These methods, are very popular due to their computational efficiency and their accurate segmentation. The intensity of a voxel on an MR Image depends mostly on the tissue type, as well as developmental processes, the amount of progress of diseases and some artifacts that are introduced by the scanner [5]. When the level of noise in an image is low enough, the intensity histogram of different tissue types in the brain is discriminative enough for a good segmentation. Some intensity based method types are thresholding, region growing, classification and clustering [1]. These methods usually try to segment a brain image in three categories, white matter (WM), gray matter (GM) and cerebro-spinal fluid (CSF). This is for two main reasons; the structure of the head is very complex and there is a great overlap of the intensities between different types of tissues [1].

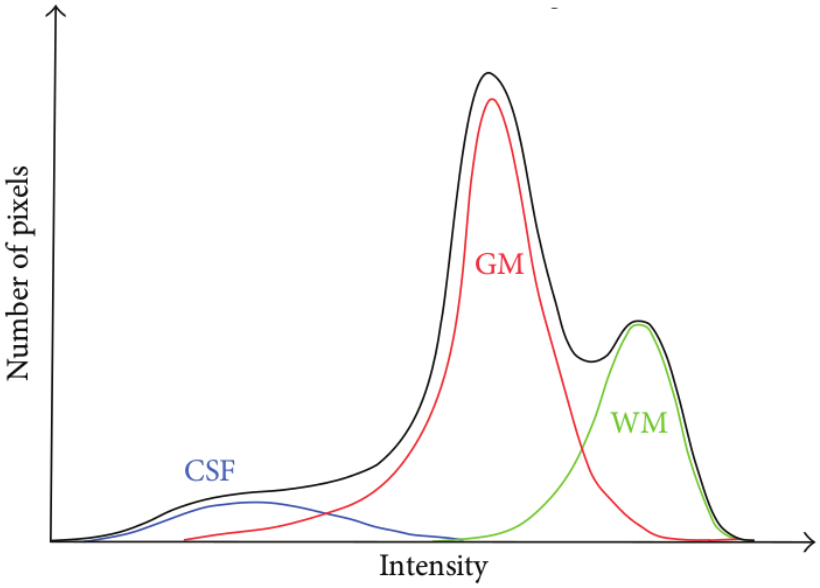

Figure 1: Intensity histogram of White Matter (WM), Gray Matter(GM) and Cerebrospinal fluid (CSF) on T1-Weighted MR Image of adult human brain. Image source [1].

Moreover, segmentation of these three types is one of the most important segmentation for various applications. Thus many researchers focus on accurate segmentation of these three categories, since even that is quite difficult task [1, 3, 6, 7, 8]. An example of the intensity histogram of WM, GM and CSF in an MR Image can be seen on figure 1, provided by [1].

Thresholding is one of the most simple segmentation methods. It uses intensity thresholds in order to classify the voxels to the desired tissues [1]. Due to its simplicity, thresholding is very sensitive to noise and intensity inhomogeneities [1]. Due to the fact that it is very fast, it is usually used in order to separate the brain tissues from background as well as a rough initialization for other methods, such as C-means clustering [1, 9]. Region growing, starts the process from a seed point, and by checking if the intensity of the neighboring voxels is similar enough (depending on predefined uniformity or homogeneity criterion) it either adds them to the region or not. The process continues until there are no more voxels to add to the region. Region growing has been used in order to segment several organs, for example it has been used to segment brain vessels [1, 10] or brain tumor [1, 11].

There are many classification techniques for segmenting the human brain. These techniques, based on prior knowledge of the intensities of each tissue as well as their position in space, try to classify the voxels of new images. For example Warfield et al. [12], used k-NN classifier, together with template matching to segment several MR images of several parts of the body. Other classification methods use the Bayes theorem incorporated in the expectation maximization (EM) framework [1, 6, 7]. Ashburner et al. used the aforementioned technique and incorporated prior probabilities from an atlas to classify the voxels. Fischl et al. used probabilities from a training set for class intensities, in combination with probabilities from an atlas and a 'relaxed' Markov Random Field (MRF) [8]. Leemput et al. also used the EM framework, in which they incorporated prior probabilities from an atlas and from a MRF [3].

Many unsupervised techniques have also been proposed. For instance Zhang et al. used the EM framework in which they incorporated a hidden Markov random field (HMRF) [7]. A very popular clustering algorithm is Fuzzy C Means (FCM). Ahmed et al. [13] proposed a variation

of the objective function of FCM where the degree of cluster membership of the immediate neighbors of a voxel is also taken into account. Such a variation make sense for clustering applications such as image segmentation, where clusters are expected to contain neighboring pixels/voxels. Chen and Zhang [14] proposed a modification of the aforementioned algorithm in order to make it more computationally efficient. They also proposed kerneled versions of the algorithm. In similar work, Shen et al. [2] introduced a variation of FCM were the degree of membership of neighboring voxels is incorporated in the distance measurement. The degree of influence is then optimized with an artificial neural network.

Atlas based methods use prior knowledge of the anatomy of the human brain in combination with segmented historical data in the form of a probabilistic map. The map represents prior probabilities for each voxel to be of a tissue type. Atlas based methods depend a lot on the manually segmented images. Moreover due to the large variability and possible deformations on different subjects, they become very unstable when segmenting images from non healthy subjects. Still, a lot of clustering and classification methods use probabilistic atlases in order to increase the accuracy of their own system. This can be done in several ways, for example the probabilities produced by atlases can be introduced in the density function of an EM clustering algorithm. In other cases the segmentation produced by a probabilistic atlas is used as a initialization technique in clustering algorithms.

Some surface based methods are active contours and surfaces [15] and multiphase active contours [1, 16, 17]. Active contours use parametric curves and surfaces to outline the boundaries of a region. Finally hybrid methods, are methods that combine several of the aforementioned techniques for better accuracy, usually for specific applications. For example Masutani et al. [18] used morphological information of local shape with model based region growing in order to segment cerebral blood vessels.

Until now, the only features mentioned are the gray level intensity as well as the spatial position of voxels, used in atlas based methods. Together with these two other features have been used as well. For example Ahmed et al. [13] also used the mean image in order add neighborhood attraction to FCM. Deformable models [1, 19] use intensity gradients as "external forces". Xue et al. [20] use the median and mean images , as well as a wavelet based filter, in order to reduce noise in the image.

In this project, textural features are going to be used for unsupervised segmentation of 3D brain MR Images. Such features have already been used for tumor classification[21]. By using such features, we hope to implement a robust system, able to segment brain tissues not only of healthy brains but also in case of deformation, for example in the presence of tumor. Although we are going to use them in an unsupervised method (clustering) they can also be used with most of the aforementioned methods, i.e. region growing, atlas based methods, surface based methods, classification methods, et cetera.

Before most of the aforementioned approaches, as well as ours, can be applied preprocessing of an MR Image needs to be done.

1.2 MRI Preprocessing

As mentioned above, MR Images suffer from various types of noise. Thus some preprocessing steps need to be done in order for automated segmentation methods to be able to produce high quality results. The most common preprocessing steps are image registration, brain extraction and bias field correction.

1.2.1 Image Registration

Image registration is the process of aligning medical images, so that same features are aligned [1]. Alignment is used for example in atlas based images were the features of different subjects need to be spatially consistent in order to produce as accurate as possible probabilities. Alignment is also used for motion correction within the same subject. For the scope of this project, alignment between different subjects is not needed. Motion correction though is very important because if two consecutive 2D slices are not aligned, 3D textural features are not reliable, since they represent the intensity relationship between neighboring voxels. The most common approach for spatial alignment is rigid transformation [1]. In this project, we are not going to implement a transformation tool, but the images with which the system is going to be tested, are already aligned.

1.2.2 Brain Extraction

During this procedure, the voxels that belong to brain tissue are extracted from the image. This step is important for most segmentation methods, because other tissue types like bone, fat and muscles have intensities that overlap with the intensities of the brain tissues, i.e. white matter (WM), gray matter (GM) and cerebro-spinal fluid (CSF). Due to the fact that all the aforementioned methods, except atlas based methods, depend solely on the intensity values of the voxels to segment the image, the existence of these tissues confuse them. In our case, although we try to segment the image before brain extraction, our experiments show that the extra tissue types also confuse our method.

The most common method for brain extraction is the use of a template image [1, 22]. Unfortunately using a template image suffers a lot in cases of deformation and of different developing stage of the brain. Stephen Smith developed an alternative method, brain extraction tool (BET) [23, 24]. This method first finds the center of gravity of the brain and then inflates a sphere, until it finds the brain boundary. This method is publicly available within the software package FSL [23, 24].

1.2.3 Bias Field Correction

The bias field, or intensity inhomogeneity is a within-image artifact created by MR Image acquisition. Its effect is a slow and smooth spatial change in the signal intensity within tissue of the same physical properties. It is mainly caused by the inhomogeneity of the magnetic field, inhomogeneity of the radio frequency pulse, the interaction between magnetic field and human body, as well as nonuniform sensitivity of the receiver coils [1, 25, 26]. This effect can be in the order of 10%-30%. This field can cause significant amount of misclassification in MR Image segmentation, especially with methods depended on gray level intensities. An example of bias field can be seen in figure 2, produced by [1].

Most state-of-the-art bias correction methods assume a multiplicative model of the field. When the intensities are log-transformed, this field becomes additive [1, 13, 6, 7, 3, 25, 26]. Many methods, incorporated the bias field estimation in the segmentation algorithm as a post-processing step[13, 6, 7, 3]. On the other hand, Lewis and Fox [25] proposed a method that works as a preprocessing step. In their work a differential bias field is estimated between consecutive 2D slices and based on that the slices are corrected. Sled et al. [26] also proposed a nonparametric approach that corrects the bias field independent of the tissue classes, and as a result, it can also be used as a preprocessing step.

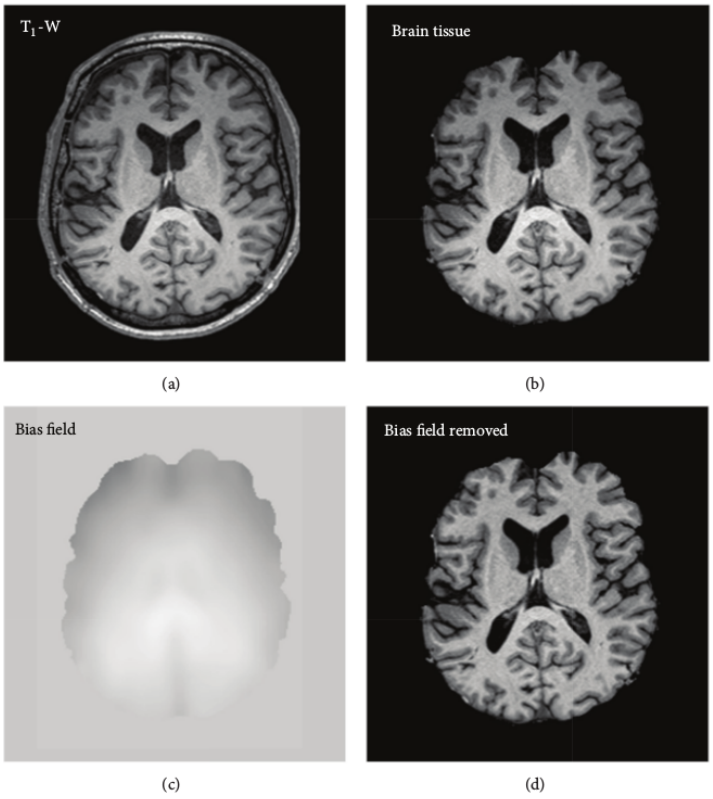

Figure 2: (a) original T1-weighted MR Image (b) brain tissue, after removing nonbrain structures (c bias field (d) brain after bias field correction). Image source [1].

1.3 Ultrasound

As mentioned above, Ultrasound Images are widely used for medical applications, for two main reasons. They are not invasive to the patients and the time needed to acquire Ultrasound images is much less than other imaging techniques. Although they have these strong benefits, they suffer a lot from speckle noise. The growth of Ultrasound Images, moves the focus of researchers towards segmenting them.

Many methods have been proposed that try to segment Ultrasound Images. H. Akbari et al. [27] proposed a system to segment prostate from Ultrasound Images. Their system applies 2D biorthogonal wavelets in three different planes, sagittal, coronal and transverse. From these it extracts textural features with which they train Kernel Support Vector Machines. There are a lot of KSVM's used, each responsible for a different plane and sub-region of the image. Finally they register the image with the database images and get a probabilistic model for the prostate. Each SVM classifies the voxels as prostate or non-prostate voxels. Each voxel has labels from the three SVM's and the probabilistic model. They combine these with a weighted function to get the final prediction. The weights for each plane and the probabilistic model are set through optimization. In similar work, J. Yang and B Fei [28] use Gabor filters to extract features for the voxels and use these features to train a KSVM. Y. Zhan and D. Shen [29] also developed a similar method to the two aforementioned. A deformable model approximates the boundaries of the prostate. Then Gabor filters are applied in order to extract features from the boundaries of the model. Several K-SVMs, each responsible for a different patch of the boundary, are used in order to classify tissues as prostate or non-prostate. J. Olivier and L. Paulhac [30] developed

an interactive process to perform skin segmentation. Their system computes textural features that are easily understandable by non-experts, i.e. granularity, contrast and roughness. The user is responsible for identifying the important features for the specific image. Then K-Means algorithm is applied to produce the segmentation. J.D. Quartaro [31] proposed a system which uses active contours to get the shape of the prostate. His system was tested on simulated images as well as phantom images and real Ultrasound images. J. Anquez et al. [32] developed a system that makes use of prior class probabilities based on gray-level intensity coupled with deformable models in order to segment a fetus from the uterus.

As with MR Images, Ultrasound images can go through preprocessing. For example, images that get statistical models from a database need to register the new image first. J.D. Quartaro [31], also tried out a lot of de-noising preprocessing steps, but concluded that the only one able to boost his system performance is intensity normalization, meaning that the values of the gray-level intensity of the voxels are normalized in a way that the whole range of intensities (0 - 255) is used.

2 Textural Features

The exploration of the capabilities of 3 dimensional textural features for the purpose of segmentation of MR Images is the main goal of this project. Textural features try to describe the intensity relationship between neighboring pixels/voxels. There are many methods that try and describe this relationship [33, 34]. Some examples are Local Binary Pattern (LBP) [35], Gray Level Co-occurrence Matrix (GLCM) [36], Run Length Matrix (RLM) [37], Markov Random Fields (MRF) [38] and many more [33, 34].

In previous work [33], we researched the capabilities of such methods in the purpose of unsupervised three dimensional image segmentation. During that project, we tested the following methods:

- First Order Statistical features (FOS) [34]
- Gray Level Co-occurrence Matrix (GLCM) [36]
- Gray Level Aura Matrix (GLAM) [39]
- Run Length Matrix (RLM) [37]

These methods produce features that are able to discriminate different textural properties in very noisy data sets [33]. The downside is that these methods can produce a very large number of features, each able to differentiate different textural properties. Thus in a completely unsupervised problem, one has to extract as many features as possible, which consists the algorithm inefficient. The same methods are going to be tested in this project, so that we can see whether they can accurately segment 3D MRI brain scans and 3D Ultrasound Images.

First order statistical features are quite self explanatory. Given a neighborhood of a voxel, they are the mean gray level intensity of the neighborhood, the standard deviation of it, the first order gradient, its mean and standard deviation. Gray level Co-occurrence matrix is a matrix that represents the probability of a voxel with gray level intensity 'A' having a neighbor with intensity 'B' in a specific direction. From this matrix several features can be computed [36, 33]. From the previous work we concluded to the following subset.

- Second Moment provides the sum of squares of all the elements in the GLCM

$$Sec.Moment = \sum_i \sum_j p(i,j)^2 \tag{1}$$

- Contrast measures the intensity contrast between neighboring pixels/voxels.

$$Contrast = \sum_i \sum_j |i-j|^2 p(i,j) \tag{2}$$

- Homogeneity or Inverse Difference moment measures the closeness of the distribution of elements of the GLCM to the GLCM diagonal

$$Homogeneity = \sum_i \sum_j \frac{1}{1+|i-j|^2} p(i,j) \tag{3}$$

- Entropy measures the 'randomness' in the image (neighborhood)

$$Entropy = -\sum_i \sum_j p(i,j) \log\left(p(i,j)\right) \tag{4}$$

- Dissimilarity is a measure of distance between pairs of pixels/voxels within the neighborhood

$$Dissimilarity = \sum_i \sum_j |i-j| p(i,j) \tag{5}$$

- Maximum probability measures the maximum likelihood of a pixel/voxel relationship (intensity wise)

$$Max.Prob = max.p(i,j) forall(i,j) \tag{6}$$

- Cluster shade and cluster prominence characterize the tendency of clustering of the pixels/voxels in the neighborhood.

$$ClusterShade = \sum_i \sum_j (i+j-\mu_x-\mu_y)^3 p(i,j) \tag{7}$$

- Cluster Prominence

$$ClusterProm. = \sum_i \sum_j (i+j-\mu_x-\mu_y)^4 p(i,j) \tag{8}$$

Run length matrix represents the number of consecutive runs, as well as their length, one can do in a neighborhood around a voxel with constant intensity. Like with GLCM a number of features can also be computed from this matrix [37, 33]. Finally Gray Level Aura matrix is a generalization of GLCM [39], were the matrix does not represent probabilities in a specific direction, rather than all possible immediate neighbors. The same features with GLCM can be extracted from this matrix. The down side of GLAM is that in case of directional texture, meaning that the properties that can differentiate different elements lie in a specific direction, features extracted from it might not be able to capture the difference. On the other hand, since it does not compute different features for each direction, the computations needed are less, and the features produced smaller in number, which makes the segmentation algorithm to converge faster.

When dealing with such a number of features, from which only some are considered as 'good' for a specified image, one has to use feature selection algorithms in order to discard irrelevant to clustering features, as well as redundant features for computational efficiency.

3 Feature Selection

Feature selection is a very important step for any data mining problem with big amount of data. The reason is that a lot of the information given is not relevant to the task in hand. In our case, as stated above, there are only a few features, for each image, able to discriminate the different textures appearing in the image. On top of that, objects in different images, may be discriminated better with different features. Thus, we can not know beforehand which features have valuable information. As a result a good unsupervised feature selection technique is needed, in order to discard irrelevant to the task features. For supervised problems, i.e. there is a training set with class information to train the prediction algorithm, most methods use the class information given and discard features that are not able to discriminate the data which belong to different classes [40]. In unsupervised learning, since no class information is given, the task is more complicated.

Many methods have been developed in order to discard features on an unsupervised manner. Generally these methods can be categorized as filter approaches, wrapper approaches or embedded approaches [41, 42]. Wrapper approaches wrap around a clustering algorithm and try to select features in order to increase the quality of clustering. In other words, they select a subset of features, cluster with a clustering algorithm the data and get feedback on the quality of the features by measuring the quality of clustering. These kind of methods are highly computationally expensive, since for every subset of features that has to be tested they cluster the data. For example, Dy and Brodley [43] used a wrapper method called FSSEM (Feature Subset Selection with EM), where the selection algorithm is wrapped around the Expectation - Maximization (EM) algorithm. They tried two different clustering evaluation metrics, namely scatter separability and maximum likelihood while for searching the feature subset space they used sequential forward search. Dash and Liu [44] introduced a search algorithm, namely RANK, used for exploring the feature subset space, and used scattering criterion as quality metric. The whole system is wrapped around k-means clustering algorithm.

Filter approaches try to tackle the problem in a different manner. The feature selection step, is considered as a preprocessing step and does not get feedback from the final clustering. Many researchers have developed filter methods. Mitra et al. [45] developed an algorithm to discard redundant features. Using k-NN algorithm they try to find the most similar features and discard them. Dash et al. [46] developed a method to measure approximately the entropy of a feature subset and use that as quality metric for the subset. Boutsidis, Drineas and Mahoney [47] developed a randomized method using top-k right singular vectors to choose features for the k-means algorithm. Basak, De and Pal [48] designed an A-NN to compute Fuzzy Feature Evaluation Index(FFEI) and with back propagation optimize the weights of the features in order to minimize FFEI. Velayutham and Thangavel [49] developed a method (Quick Reduct) based on rough set theory. Li, Lu and Wu [41] proposed a two layer approach, in order to get the advantages of two different methods, i.e. one method is responsible for discarding redundant features and an other to discard irrelevant to clustering features. Their idea was also used by [50], where the final implementation consists of three layers of filters.

In this project the two layer filter approach is going to be adapted to our needs. This is because all the aforementioned methods have at least one short coming, i.e. too computationally expensive for big datasets, discard only redundant features, discard only irrelevant features [41]. Some of the above filter methods and variations of them will be tested in order to find the best set up for the needs of this project. The rest of this section is organized as following. First the methods tested are going to be presented and then the variations of them will be

described.

3.1 RANK, SRANK

As mentioned above Dash and Liu [44] proposed a feature subset space search algorithm, namely RANK. The main idea is to rank and sort all the features, based on a quality for clustering metric. Then the ranked list of features can be used for faster search of the feature subset space, i.e. instead of choosing features blindly there is a preference on the features that have good ranking.

The metric they used is the loss of entropy when the feature in question is removed from the dataset. The entropy for a dataset with N points is given by the following formula [44]:

$$E = -\sum_{i=1}^{N}\sum_{j=1}^{N}(S_{i,j} * \log S_{i,j} + (1 - S_{i,j}) * \log (1 - S_{i,j})), \tag{9}$$

where $S_{i,j}$ is the similarity of instances i and j and its given by the following formula:

$$S_{i,j} = e^{-a*D_{i,j}}, \tag{10}$$

where a a parameter and $D_{i,j}$ the normalized distance between instances i and j.

Algorithm 1 RANK

```
for every feature F_i do
    P_i = CalcEnt(F_i)
end for
Return P
```

Since RANK needs to go through all pairs of instances, multiple times (one for every feature) for big datasets it is very computationally expensive. In order to tackle this, SRANK was introduced. SRANK takes p random samples of n data points and calls RANK on each sample. The final ranking of a feature is the summation of the individual rankings.

Algorithm 2 SRANK

```
for all features F_i, Overall Rank (OR_i) = 0
for l = 1 to p do
    take sample L_l
    P_l = RANK(L_l)
    for every feature k do
        OR_k = OR_k + P_{l,k}
    end for
end for
Return OR
```

Although [44] wrapped RANK around k-means clustering, it was incorporated as a feature subset space search method to the second filter of the two layer filter approach [41], responsible for discarding irrelevant to clustering features.

3.2 Mitra's et al. algorithm

Mitra et al. [45] developed an algorithm, based on the k-NN algorithm in order to discard redundant features. The k-NN algorithm has as distance measurement the similarity of the features. As a similarity metric they used Maximal Information Compression Index (MICI). For two random variables x and y, MICI is given by the following formula:

$$\lambda_2 = \frac{1}{2}(var(x) + var(y) - \sqrt{(var(x) + var(y))^2 - 4var(x)var(y)(1 - \rho(x,y)^2)}), \quad (11)$$

where $var(x)$ and $var(y)$ are the variances of variables x and y and $\rho(x,y)$ is the correlation coefficient between the random variables x and y.

The algorithm initially computes the nearest neighbors for all features. The feature for which the distance from the kth neighbor is minimum they discard the k nearest neighbors. Then they define similarity threshold (ϵ) equal to the distance of the aforementioned feature to its kth neighbor. Continue with the rest of the features, picking them with the same manner, and discarding only the features with similarity smaller than ϵ.

3.3 Neuro-Fuzzy Approach

Basak, De and Pal try to incorporate ANN with fuzzy set theory [48] in order to not only find the best subset of features, but also weight them for optimal results. They achieve that by introducing a fuzzy feature evaluation index (FFEI).

$$E = \frac{2}{s(s-1)} \sum_{p} \sum_{q \neq p} \frac{1}{2}[\mu_{pq}^T(1 - \mu_{pq}^O) + \mu_{pq}^O(1 - \mu_{pq}^T)], \quad (12)$$

where μ_{pq}^O is the degree that both pth and qth data points belong to the same cluster in the original data space and μ_{pq}^T in the evaluated subset. The degree of membership, μ_{pq} is given by the following formula:

$$\mu_{pq} = \begin{cases} 1 - \frac{d_{pq}}{D}, \text{ if } d_{pq} \leq D \\ 0, otherwise \end{cases}, \quad (13)$$

where D is proportional to the maximum distance in the dataset.

FFEI decreases as the degree of membership increases/decreases, meaning that the boundaries of clusters become better defined. In order to find the optimum feature subset, they designed an Artificial Neural Network that computes FFEI from the dataset by weighting the features and using back propagation to optimize them. Since for very large datasets, the computation of FFEI becomes very expensive, random sampling is applied.

3.4 Two Layer Filter Approach

Li, Lu and Wu [41], argued that all existing methods have at least one of the following shortcomings. Only remove redundant features, only eliminate irrelevant features, low performance on high dimensional data set, expensive computation cost for high dimensional data or sensitive to noisy data. In order to overcome this issue, they proposed a two layer filter approach. Use one method that is very efficient at discarding redundant features and one at irrelevant to clustering features. The filter that is less computationally expensive will be used first. In order to eliminate redundant features, they used the method proposed by Mitra et al. [45]. For

discarding irrelevant to clustering features, they proposed a method that not only does that, but also weights the remaining features for optimized results.

The method consists of three main components: feature subset space search, feature subset evaluation and subset selection. For searching the feature subset space, they used RANK (Algorithm 1) and for big datasets SRANK (Algorithm 2). Instead of the entropy, they use a slightly different metric, called ranking index and is given by the formula:

$$H = \sum_{i=1}^{N} \sum_{j=1}^{N} (S_{i,j} * e^{S_{i,j}} + (1 - S_{i,j}) * e^{(1-S_{i,j})}), \tag{14}$$

Again $S_{i,j}$ is the similarity between instances i and j and is given by equation 2.

Once all features are ranked, a weighting scheme is applied, based on the rankings. The weight of each feature is equal to its contribution on the overall H-value difference. More specifically, the overall H-value difference is defined as the summation of the difference of the H-value for each feature to the one ranked lowest. In algorithm 3, m is the total number of features, and assumed that they are sorted in decreasing order based on the value of H. The Difference value of the last feature (DH_m) is set to one, so it has a minimum value different than zero.

Algorithm 3 CalcWeight

Overall Difference of H (ODH) = 0
Difference of H for feature m(DH_m) = 1
for k = 1 to m - 1 **do**
    $DH_k = H_k - H_m$
    $ODH+ = DH_k$
end for
$ODH+ = DH_m$
for k = 1 to m **do**
    $w_k = DH_k/ODH$
end for

To select the feature subset, they use forward selection. For evaluation of a feature subset, FFEI (Equations 4, 5) is used. After defining a threshold ϕ for minimum decrease of FFEI, an iterative process is executed where features are added to the chosen subset until the difference of FFEI before and after the addition of a feature is less than ϕ.

3.5 Proposed variations of Mitra's et al. Algorithm

Mitra's et al. algorithm works very well and has very low computational cost. Whilst this is true, it has one drawback, which for our case is quite essential. This algorithm takes as a parameter the number of nearest neighbors k, which does not reflect the similarity threshold for all datasets. Meaning, for every dataset the similarity threshold will be different for the same k. In a situation where there are no redundant features Mitra's et al. algorithm will still discard a number of them. There is an equivalent behavior for the opposite situation. If most of the features are redundant, there is a high probability that many of them will not be discarded. In order to avoid this behavior, two variations of this algorithm, are proposed, both with the same principle, instead of defining k, the similarity threshold ϵ is defined instead.

The first variation works as following. Find for every feature, the number of features closer than ϵ. Starting with the feature that has the most, discard all the features with similarity less than ϵ.

Algorithm 4 1st Variation

Define ϵ
for every features k **do**
    Find all features $l_i \neq k$, with $S_{lk} < \epsilon$
    Add l_i to L_k
    $C_k = count(l_i)$
end for
sort(C, L)
for every feature k **do**
    Discard $\forall l_i \in L_k$
end for

In algorithm 5, L_k is a vector of features containing all features l_i that have similarity with feature k S_{lk} smaller than the threshold ϵ.

The second variation is very similar to the first. The difference is the decision mechanism for choosing the order of features. Instead of looking which feature has the most neighbors, we start from the feature with the smallest entropy. The entropy is calculated with an approximate method developed by Dash et al. [46].

Algorithm 5 1st Variation

Define ϵ
for every features k **do**
    Calculate Approximate Entropy of k
end for
Sort features based on Approximate Entropy
for every feature k **do**
    Discard all features $l_i \neq k$, with $S_{lk} < \epsilon$
end for

For both variations, also the similarity measurement is slightly modified. MICI, is a very good similarity measurement for features, since it is invariant to rotation, it is symmetric, and it is equal to zero when the two features are linearly related [45]. Unfortunately the upper bound is $0.5 * (var(Feature_1) + var(Feature_2))$. This means that it is not invariant to scaling. If two features have high variances the MICI will be larger than in the case of features with small variances. In order to avoid this factor, we calculate the following similarity measurement:

$$Similarity = \frac{2 * MICI(Feature_1, Feature_2)}{var(Feature_1) + var(Feature_2)} \tag{15}$$

Now the similarity takes values from zero to one. The behavior of these two measurements is presented with the help of a simple example. Given the features in table 1, table 2 shows the calculated MICI and our similarity measurement. Clearly for clustering purposes, features 1 and 2 should be similar.

	Point α	Point β	Point γ
$feature_1$	0,2	0,3	0,9
$feature_2$	0,8	0,7	0,2
$feature_3$	0,2	0,8	0,3

Table 1: The values of three data points for features 1,2 & 3.

	features 1,2	features 1,3
MICI	0,000325	0,118354
Our Similarity	0,001822811	0,719608105

Table 2: The values of MICI and our similarity's measurement for pairs of features 1,2 and 1,3.

Both similarity measurements are able to discriminate the similar features from the not similar. The difference is that due to the dependence of MICI on the sum of the variances of the features, the value of MICI for features 1 and 3 is quite small. Thus it will make it very hard to come up with a threshold that is universal for all possible pairs of features. Thus our similarity measurement is going to be used.

3.6 Simplification of Irrelevance Filter

For this project, a simplified variation of the irrelevancy filter described in section 5.4 is going to be used. Due to the large time complexity of the computation of FFEI, the last part of the algorithm is going to be changed. Instead of adding features until the FFEI does not drop more than the predefined threshold, a weight threshold is going to be used. Meaning that after the features are weighted, if the weight that is assigned to a feature is less than the predefined threshold, the feature is going to be discarded. Moreover, through the experiments, a limitation of the weighting scheme is identified. The last feature (the one with the worst ranking) has a very low weight, almost zero. In order to avoid that factor a slightly different weighting scheme is proposed. Instead of setting the DH value of the last feature to one, a new virtual feature is introduced with score equal to last minus the average difference of rankings of the features. The weights given are the difference of each feature from the virtual feature divided by the difference of the best feature with the virtual feature. It should be noted that the average distance is not computed for all combinations of features, rather than the consecutive pairs after sorted by their rankings.

Algorithm 6 CalcWeightNew

1: calculate average difference AVG
2: Ranking of virtual feature $\mathrm{V}(H_v) = H_m - AVG$
3: Calculate Maximum difference $DH_{max} = H_0 - H_v$
4: **for** k = 1 to m **do**
5: $\quad w_k = (H_k - H_v)/DH_{max}$
6: **end for**

With this weighting scheme there are still some downsides. For example the smallest weight given is always $\frac{1}{N_{featuers}}$.

4 Clustering

There are many algorithms that can be used for clustering, such as K-means, Expectation Maximization (EM), Fuzzy C-means (FCM) and many more. For the purpose of MRI segmentation the most popular are EM and FCM [1, 6, 7, 3, 4, 20, 14]. Zhang et al. [7] proposed a method that incorporates probabilities drawn from a Hidden Markov Random Field (HMRF)

and probabilistic distribution for the bias field in the EM's objective function, for the purpose of segmentation of WM, GM and CSF in healthy human brain. The HMRF model serves the purpose of representing neighborhood attraction. Ashburner and Friston [6] added to the EM's objective function a model for the bias field, but also incorporated prior probabilities from an atlas, making the method supervised. Xue et al. [20] used a wavelet based filter to de-noise the image. Then with intensity thresholding initialize the clusters which are then optimized with the help of FCM. In order to get rid of the 'salt and pepper' noise of the clustering, the image goes through median and weighted average filters, with the help of class labels. Finally with the new image and as initial centers the centers from FCM, FCM is triggered again to produce the final result. Ahmed et al. [13] modified the objective function of FCM in order to take into account the class labels of the neighboring voxels and Chen and Zhang [14] modified the latest method in a way that it is more computationally efficient and included kernel distance measure. It is important to mention that all the aforementioned methods use only the intensity values of the voxels in order to cluster an image and also that they focus on clustering WM, GM and CSF, from healthy human brains using 3D MR Images.

In this project K-Means++ [51] is going to be used for most of the experiments, due to its known robustness and computational efficiency. FCM, as well as Chen and Zhang's variation, are going to be tested to see whether they can boost the performance of the system.

4.1 K-Means++

K-Means++ is a variation of the original K-Means. The sole difference of the two methods is the initialization technique of the centroids. In the case of K-Means, K (number of clusters) data points are picked at random to serve as the centroids, with a uniform probability for all points. In the case of K-Means++ although the centroids are also picked at random from the data set, the probability of a point being chosen is proportional to its distance from the closest, already picked, centroid. As a result, the initial centroids are more likely to be better distributed along the data, resulting in faster convergence and higher probability of avoiding local minimum.

4.2 Fuzzy C-Means (FCM)

The FCM algorithm is a generalization of K-Means [52, 1], based on fuzzy set theory [53]. It is a generalization because it allows points to belong in several clusters, based on a membership function. The objective function of FCM is:

$$J_m = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m * ||x_j - v_i||^2, \tag{16}$$

where N is the number of data points to be clustered, C is the number of clusters, u_{ij} is the membership function of point x_j for cluster i and m is a parameter that controls the level of fuzziness of the resulting clustering. If m is set to one, the algorithm is the same with K-Means clustering. Given the constraints:

$$u_{ij} \in [0, 1],\ \sum_{i=1}^{C} u_{ij} = 1, \forall j,\ 0 < \sum_{j=1}^{N} u_{ij} < N,\ \forall i \tag{17}$$

and setting to zero the partial derivatives of J_m with respect to u_{ij} and v_i, one will get two conditions for minimizing J_m:

$$u_{ij} = \left[\sum_{k=1}^{C} \left(\frac{D_{ij}}{D_{kj}} \right)^{\frac{1}{m-1}} \right]^{-1}, \tag{18}$$

$$v_i = \frac{\sum_{j=1}^{N} u_{ij}^m * x_j}{\sum_{j=1}^{N} u_{ij}^m} \tag{19}$$

The algorithm, given initial centers, calculates the membership function u_{ij}. Given that membership function, it recomputes the centers. FCM iterates between these two steps until the maximum relocation of a center is smaller than a user specified threshold ϵ. Finally in order to get the crisp version of the clusters, each point is assigned to the cluster to which its membership (u_{ij}) is maximum.

4.3 FCM modification, including neighborhood attraction

As mentioned above, Ahmed et al. proposed a variation of FCM so that the classes of neighboring voxels can affect the membership function. This make sense in image segmentation, because the clusters within an image are expected to contain neighboring pixels/voxels. In order to do so, they introduced an extra term in the objective function J_m:

$$J_m = \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m * ||x_j - v_i||^2 + \frac{\alpha}{N_R} \sum_{i=1}^{C} \sum_{j=1}^{N} u_{ij}^m \left(\sum_{x_r \in N_j} ||x_r - v_i||^2 \right), \tag{20}$$

where N_j is the set of neighbors of voxel j, N_R is the number of neighbors and α is the parameter that controls the importance of the neighboring attraction. By computing the partial derivatives of J_m we get the new formulas for updating the centers and membership functions:

$$u_{ij} = \left[\sum_{k=1}^{C} \left(\frac{D_{ij} + \frac{\alpha}{N_R} \left(\sum_{x_r \in N_j} ||x_r - v_i||^2 \right)}{D_{kj} + \frac{\alpha}{N_R} \left(\sum_{x_r \in N_j} ||x_r - v_i||^2 \right)} \right)^{\frac{1}{m-1}} \right]^{-1}, \tag{21}$$

$$v_i = \frac{\sum_{j=1}^{N} u_{ij}^m * \left(x_j + \frac{\alpha}{N_R} \sum_{x_r \in N_j} x_r \right)}{(1+\alpha) \sum_{j=1}^{N} u_{ij}^m} \tag{22}$$

Like with the original FCM, this algorithm iterates between updating the centers and computing the membership function. The process stops with the same criterion as before.

5 Experiments

5.1 Experimental Setup

In order to test our system and the different methods that can be used within it, the experiments are divided in three sections. First an exhaustive search of the different methods and their parameters is going to be done in order to get insight of the performance of different

methods on different images. Based on the results the second stage is designed in which the redundant filters are going to be tested. Finally, filters that discard irrelevant features are going to be tested. From these experiments a final system is going to be defined. In order to get a clear image of the approach, the work flow of the proposed system is presented in figure 3.

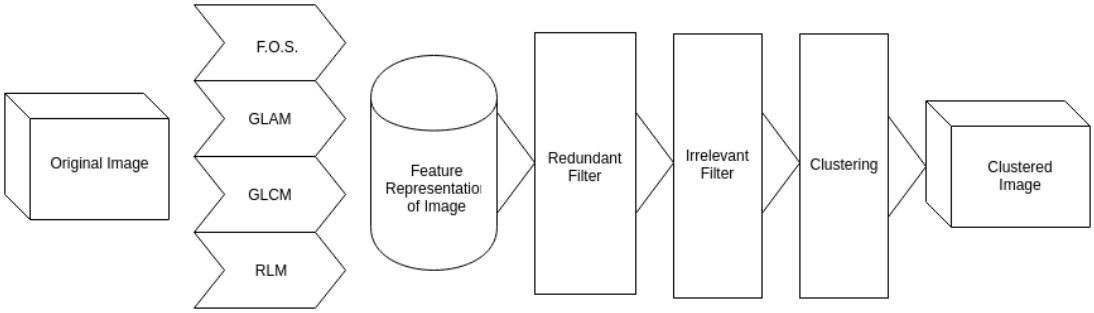

Figure 3: Proposed system work flow.

5.2 Validation

In order to validate our results, an accuracy measurement is needed. Manning et al. [54] introduce four metrics for clustering, namely Purity, Normalized mutual information, Rand Index and F measure. According to them Purity and Normalized mutual information suffer from the same drawback; they do not penalize high values of the number of clusters produced. Meaning that if a class from the data set is split to two clusters, but there are no points from different classes in these clusters, purity and normalized mutual information will output the maximum value (1). Random Index on the other hand, will equally penalize false positive as well as false negative examples. It accomplishes that by measuring the amount of pairs of data points that are clustered correctly or wrong to the same or different cluster. Random Index is given by the following formula:

$$R.I. = \frac{TP + TN}{TP + TN + FP + FN}, \tag{23}$$

where TP refers to the true positives, which are the pairs of data points that were clustered in the same cluster and belong to the same class, TN refers to the true negative, which are the pairs of data points that are clustered in different clusters and belong to different classes. FP refers to the false positive and FN refers to the false negative. The F - measure is a variation of Random Index, which allows different weighting of false negatives and false positives. The F - measure can be computed using the following formulas:

$$P = \frac{TP}{TP + FP},\ R = \frac{TP}{TP + FN},\ F_\beta = \frac{(\beta^2 + 1) * P * R}{\beta^2 * P + R} \tag{24}$$

Other evaluation measurements that are used in medical bibliography are the Dice index [55, 1, 56, 57], the Jackard Similarity [1], sensitivity and specificity [56, 57]. The Dice index is given by the following formula:

$$\rho_i = \frac{2|A_i \cap B_i|}{|A_i| + |B_i|} \tag{25}$$

These similarity measurements require classification of the volumes. Sensitivity measures the true positive rate whilst specificity the true negative rate. Due to the unsupervised manner

of our approach, these measurements are not very sufficient as we need to also incorporate a way to classify the resulted volumes. Thus, the Rand Index and F measurement seem more appropriate. Since we do not want any bias in favor of false negatives or false positives, the Rand Index measure is going to be used.

5.3 MRI

5.3.1 Data Set

In order to test the proposed system and optimize it, we had to obtain a dataset with as accurate ground truth as possible. Given the difficulty and time needed to label a 3D image manually, such a dataset is not easy to find. There are four data sets that came to our attention, namely BrainWeb [58, 59, 60, 61, 62], IBSR [63], BraTS [64] and MRBrains [65].

BrainWeb is a collection of MRI simulated 3D images for healthy brains as well as brain with multiple sclerosis (MS). The simulator is a complex system. Starting from a digital phantom, it performs modeling based on the Bloch equations in order to build the simulated image [59]. The dataset is composed by two phantoms, one of healthy brain and one of a brain with MS. For each phantom there are 18 different simulated images. Each image has a different level of Gaussian noise and intensity non-uniformity (bias field), i.e. the level of Gaussian noise can be 0, 1, 3, 5, 7 or 9 percent while the intensity non-uniformity level 0, 20 or 40 percent. Since the data set is composed by simulated images, it has excellent ground truth. As such, it is widely used by researchers in order to validate their methods, although simulated images might not have the same artifacts, noise distribution and intensity histograms as the real images [1, 6, 56, 57]. Example T1 weighted and T2 weighted simulated images, as well as their ground truth for CSF, WM and GM, can be seen in Figure 4.

The Internet Brain Segmentation Repository (IBSR) is provided by the Center for Morphometric Analysis at Massachusetts General Hospital [63]. It is composed by two data sets. The first contains twenty T1 weighted 3D MRI scans, whose manual segmentations are provided. All MRI scans have 1mm x 1mm x 3mm resolution. The second contains 18 scans whose resolution is (.84mm - 1mm) x (.84mm - 1mm) x 1.5mm. For these images manual segmentation is also provided. The set contains scans from juveniles to a seventy year old person. The images provided have gone through bias field correction by the CMA 'autoseg' routines. Unfortunately many papers point out that the ground truth is not very accurate [1, 7]. Although this is the case, due to lack of other datasets with well specified ground truth for segmentation, it is widely used for method comparison and validation.

MRBrains and BraTS are composed by real and simulated images of brains with tumor. Manual segmentation is provided, for the tumor and edema and in some cases also other volumes such as white matter, gray matter, scull, muscles et cetera. Unfortunately they have restricted access. In order be able to use them the administrator has to grand access. MRBrains can only be used for the purpose of the MRBrain challenge, thus we were not able to obtain these two datasets.

5.3.2 Textural Method Comparison

In order to compare the methods that are explored in this paper, as well as understanding how parameters such as step size (GLCM and RLM), neighborhood size and neighboring element size (GLAM) affect the results some initial experiments are done. For this initial experimentation, six images from the BrainWeb Repository are used. More specifically three

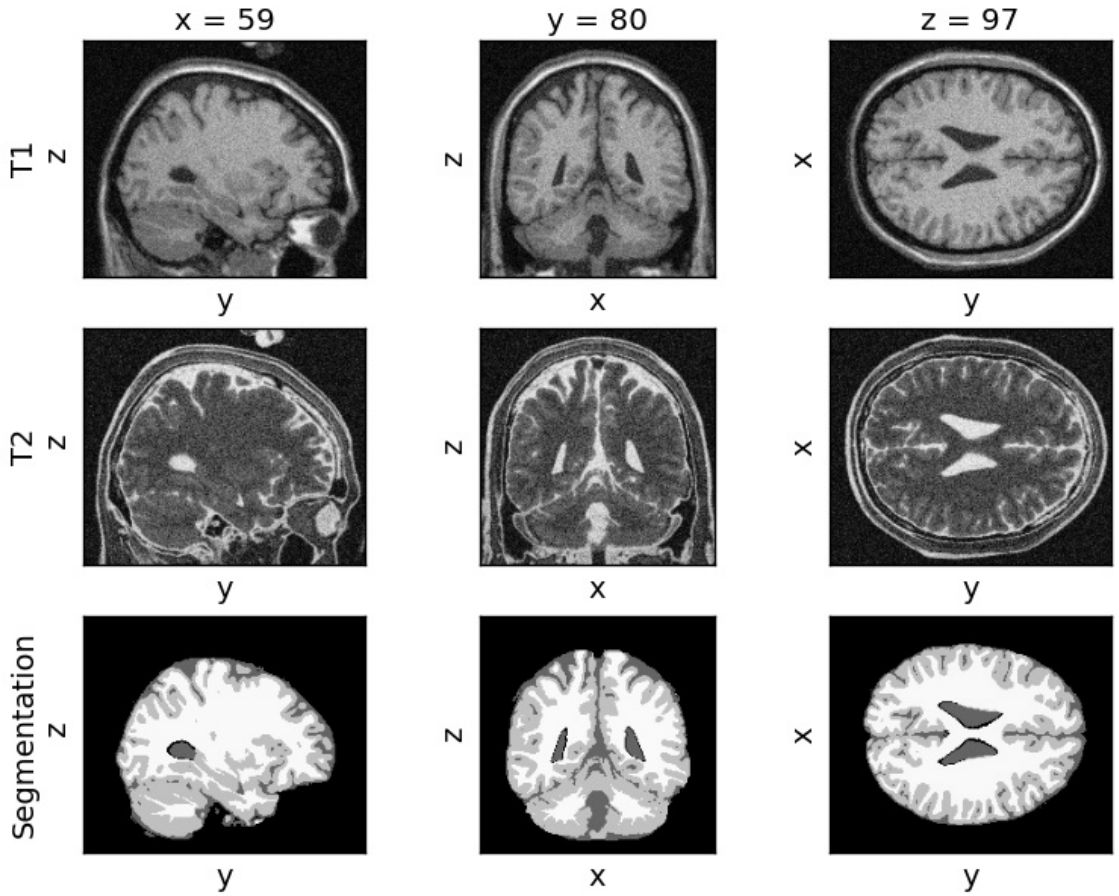

Figure 4: Example T1-weighted and T2-weighted images from BrainWeb with their ground truth. Both images have inhomogeneity level 20% and 9% noise.

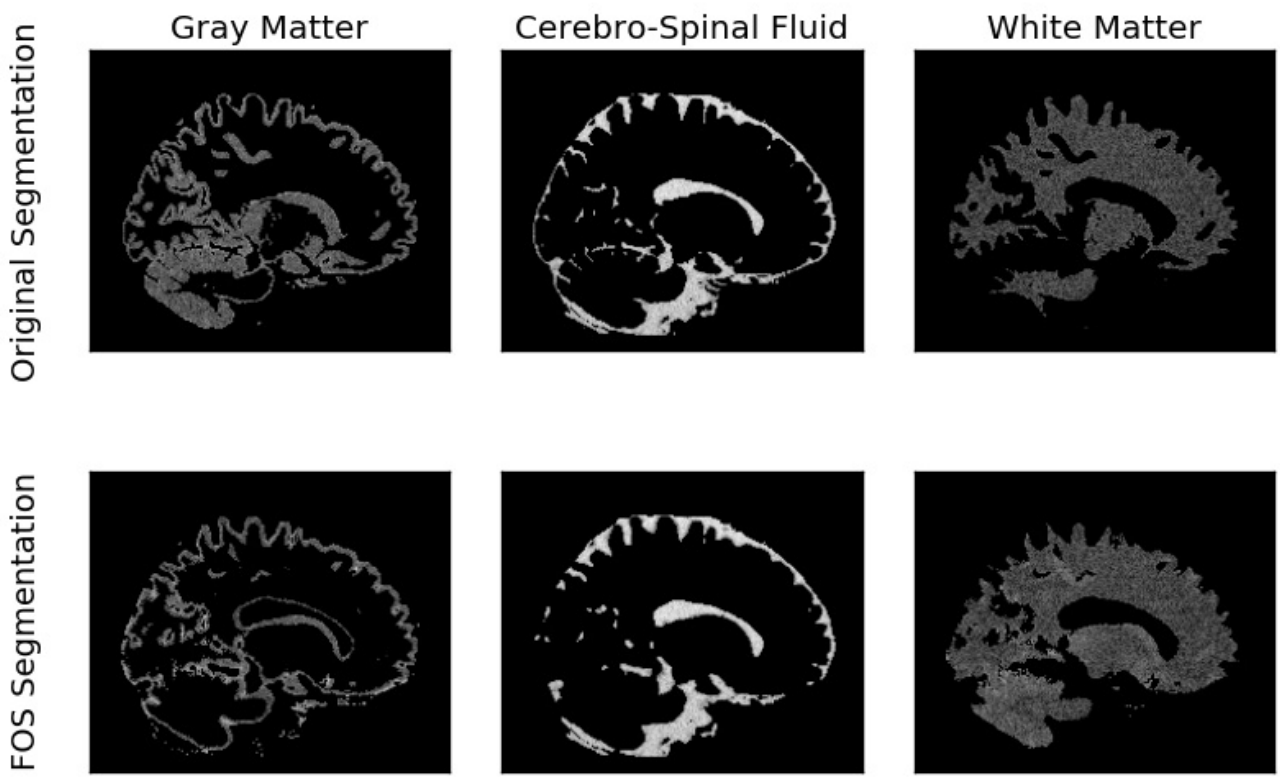

Figure 5: Segmentation produced by FOS on T2 Weighted image, Noise = 9%, Bias Field = 20%.
Neighborhood Radius = 4

T1 - weighted and three T2 - weighted, with different combinations of noise and inhomogeneity level. The combinations are nine percent noise with twenty and forty percent inhomogeneity level and five percent noise with forty percent inhomogeneity level. Every method is used alone, only with feature rescaling, so that no feature is weighted more that the others due to different scaling. The clustering is done by K-Means++.

First order statistical features (FOS) are tested for neighborhood size from two to five. They are not tested for neighborhood radius one, because in order to compute the absolute gradient value there is the need of at least neighborhood radius 1. In order to measure its mean and standard deviation the minimum possible radius becomes two. The upper bound was chosen due to the results. As seen in figure 6, experimentation for larger values of neighborhood radius seems unnecessary.

It is clear that FOS performs better on T1 weighted images, as the difference in accuracy reaches even twenty percent. Moreover when dealing with T1 weighted images the bias field significantly affects the results whilst Gaussian noise does not. This effect does not appear with the same magnitude when dealing with T2 weighted images. In that case, the general behavior looks quite different. That is because the difference in intensity level between gray

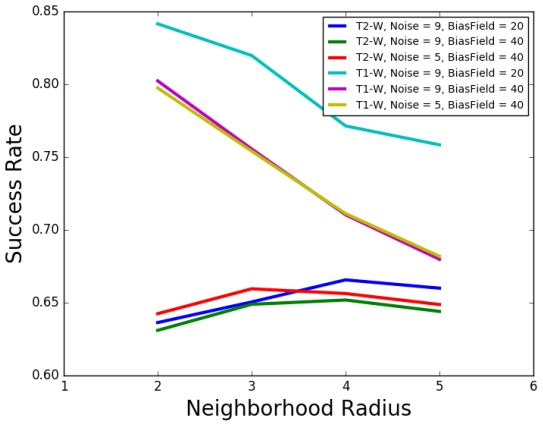

Figure 6: FOS success rate for neighborhood size from two to five, for six simulated images from BrainWeb.

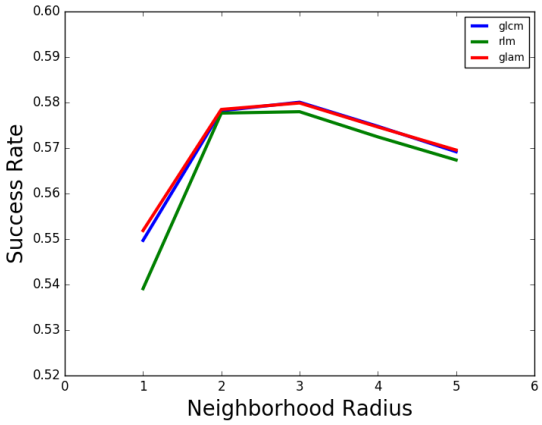

Figure 7: Success rate for neighborhood size from one to five, for RLM, GLCM and GLAM when clustering T1-Weighted image with Noise = 9%, Bias Field = 20%.

mater and white mater is much smaller than in the case of T1 weighted images. As seen in figure 5, although the algorithm is able to cluster CSF quite accurately, it completely confuses white matter and gray matter, while assigning them to the same cluster and having as a third cluster their boundaries with CSF. The confusion of WM and GM is probably the reason of the different behavior. While the neighborhood size increases, the 'boundary' cluster becomes larger. Since most of the voxels that are close to the boundary with CSF belong to GM, when the 'boundary' cluster becomes bigger it starts representing GM. This effect stops for big enough neighborhood sizes, as voxels that belong to white matter start being clustered in the 'boundary' cluster as well. The resulted clustering for T1 weighted image can be seen in figure 8.

The other three methods are tested for neighborhood radius from one to five and step size (in case of RLM and GLCM) or neighboring element size (in case of GLAM) from one to neighborhood radius. Figure 7 shows the accuracy of these methods for different neighborhood sizes. All three methods produce very similar results. For the rest of the images the behavior recorded is also very similar. Since GLAM produces slightly better results, the rest of the plots shown are for GLAM. If an other method shows different behavior it will be specified.

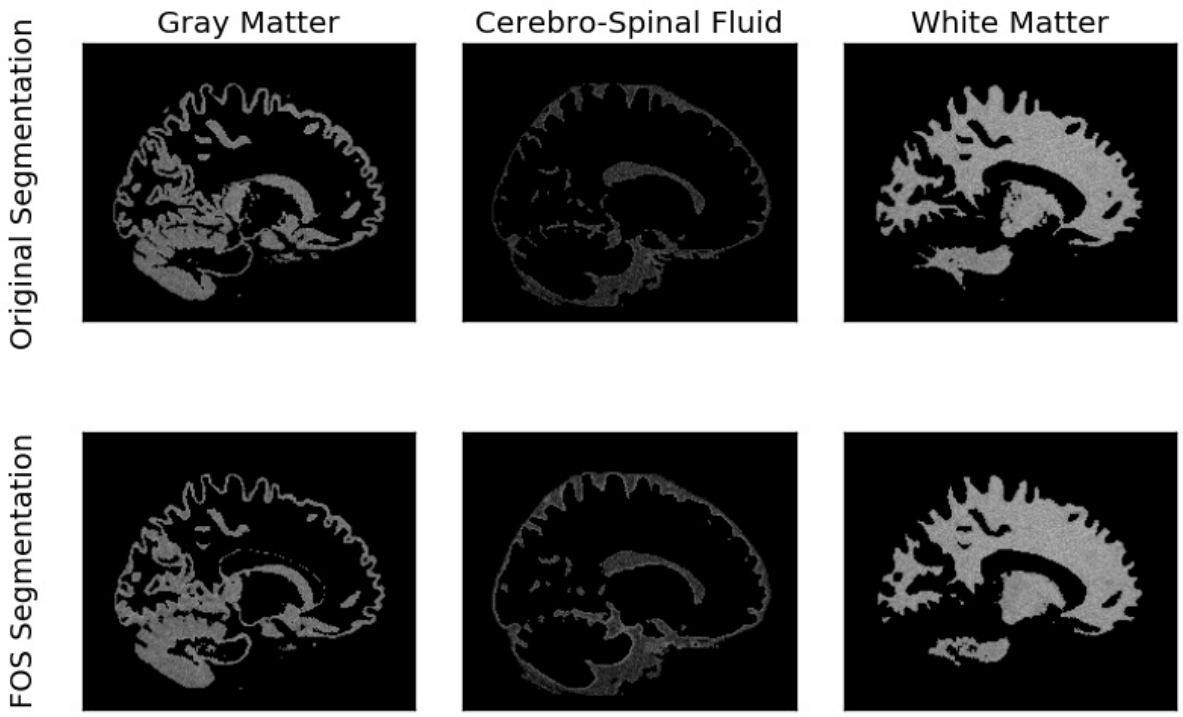

Figure 8: Segmentation produced by FOS on T1 Weighted image, Noise = 9%, Bias Field = 20%, Neighborhood Radius = 2

Figure 9 shows the accuracy of GLAM for all six images. It is clear that its behavior is different than the one of FOS. When dealing with T2 weighted images, performs better than when dealing with T1 weighted. It is important to notice that when the Gaussian noise increases, the performance also increases, whilst when the inhomogeneity level increases, the performance decreases. That is to be expected since all textural methods depend on high frequency noise that depends on the objects, in order to be able and discriminate them. Thus when dealing with T1 weighted images, in which the intensity is more homogeneous than T2, or with low level of Gaussian noise, the only texture that they "see" is around the boundaries between tissue types. Thus they cluster the image to homogeneous regions (not boundary regions) and non-homogeneous region (boundary regions).

Figure 10 shows the accuracy for GLAM, for all six images for different neighboring element sizes. The neighborhood radius is set to five. The size of the neighboring element, does not affect much the performance of the method. In most cases it doesn't affect it at all, whilst in the cases that it does, the difference is almost negligible. For different neighborhood sizes the behavior is the same. Since GLAM has time complexity $O(N_S^3)$, where N_S is neighboring element size, and the performance is not affected by it, small values of it seem more appropriate for an application. An example of GLAM segmentation can be seen in Figure 11.

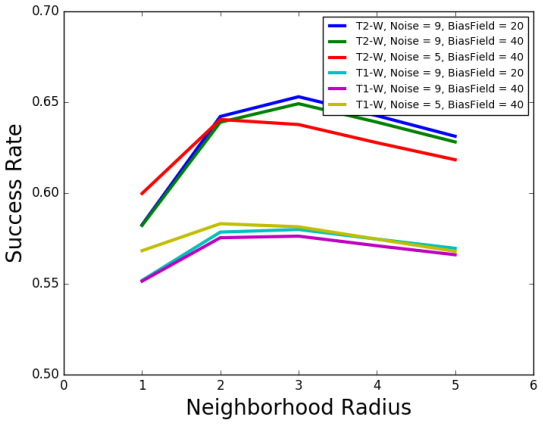

Figure 9: GLAM Success rate for neighborhood size from one to five for all six images.

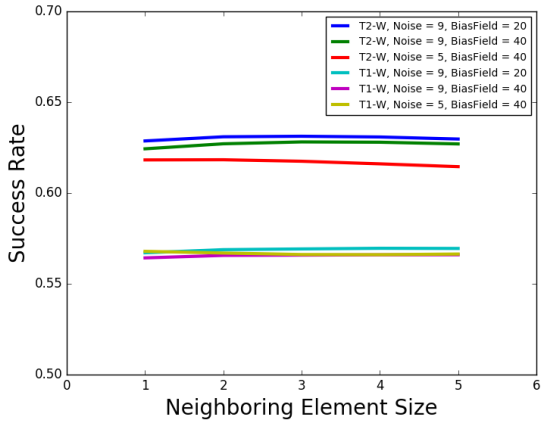

Figure 10: GLAM Success rate for neighboring element size from one to five for all six images. Neighborhood radius is five.

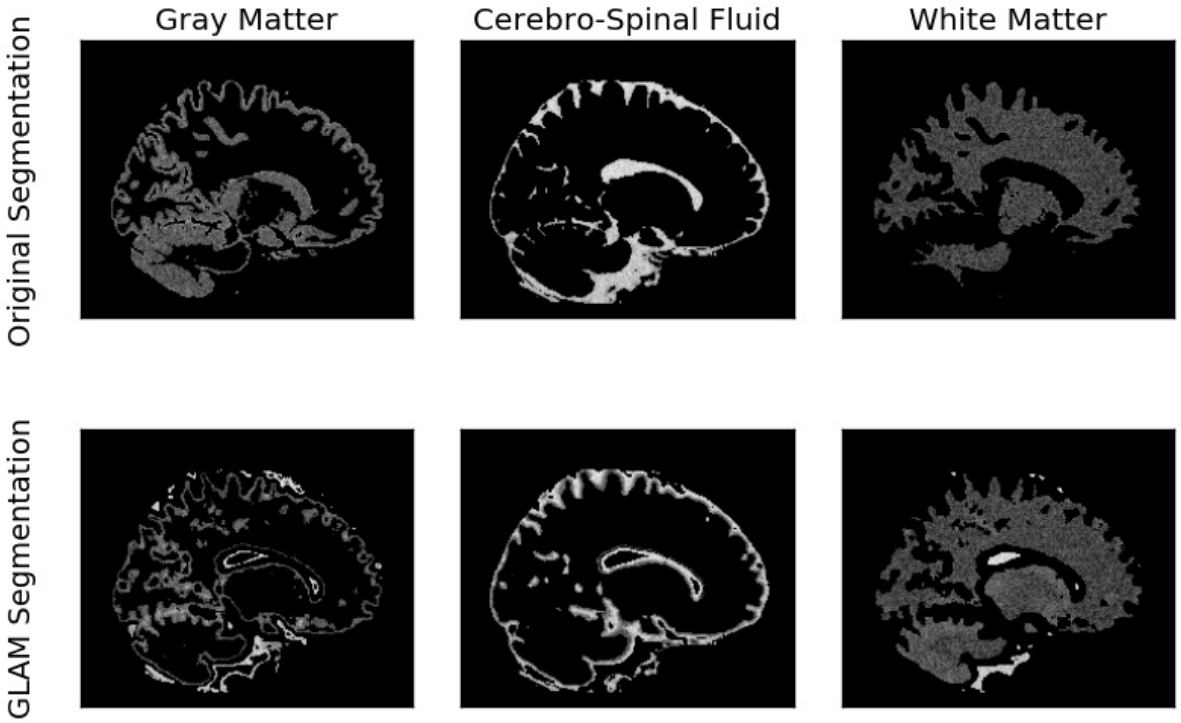

Figure 11: GLAM segmentation for T2-Weighted image, Noise = 9%, Bias Field = 20%. Neighborhood Radius = 3, Neighboring element size = 2.

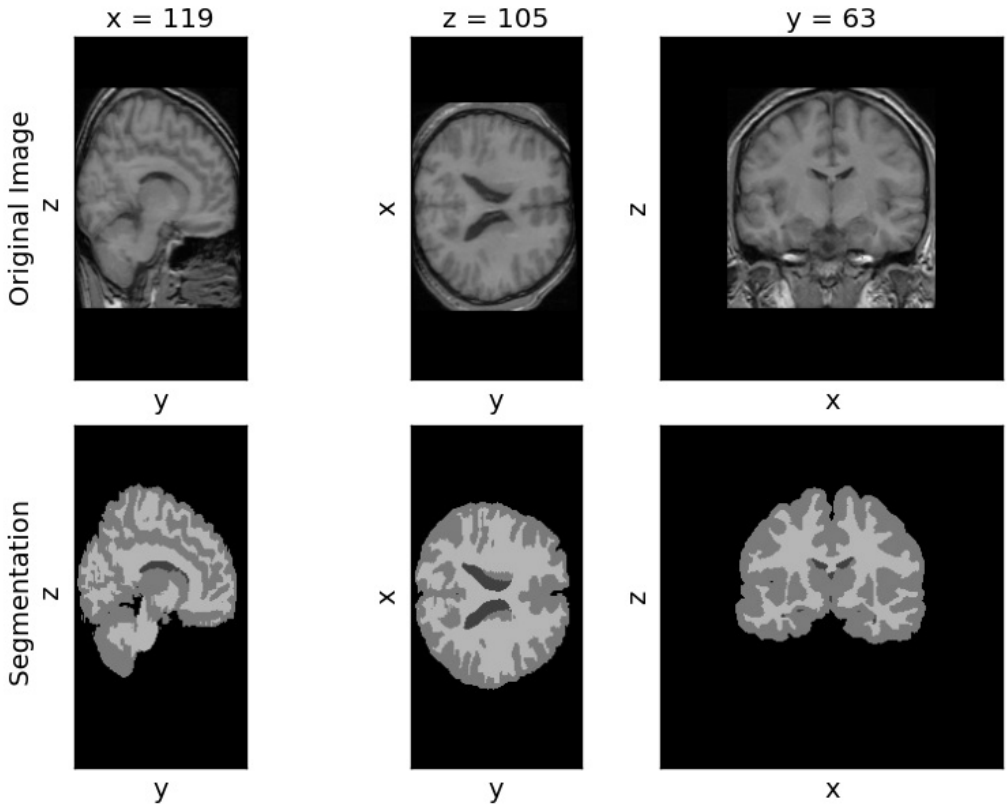

Figure 12: Example real MR Image with Ground truth from IBSR v2.0

Except the simulated images, the above methods are also tested on a database of real MR Images. The images are collected from the Internet Brain Segmentation Repository (IBSR) [63]. Five images from the second version are used. That is due to their higher resolution. Since the methods used use all three dimensions in order to extract features, if the resolution in one of the three dimensions is reasonably lower than the other two, then the methods are "confused". Two images are chosen, from 41 year old people, one from 37 and two from juveniles. Unfortunately the ground truth of this dataset is not very accurate [1, 7]. An example image can be seen on figure 12. Especially in the case of CSF (comparing also with anatomical models) big parts are missing, or misclassified as Gray Matter. For example in the image (and generally the brain structure, see figure 4) there is a layer of CSF around the brain (white matter and gray matter). Part of this layer is not present in the segmentation, whilst an other part classified as Gray Matter. Moreover usually there are small volumes of CSF between gray and white matter which, in this case, are either classified as gray or white matter. As mentioned before, all five images have gone through bias field correction by the CMA 'autoseg' routines.

First order statistical features show the same behavior as with the simulated T1 images, with much lower accuracy (Figures 6, 15). As the neighborhood size increase, the performance decreases. One of the reasons for the lower performance is the poor ground truth of the dataset. An example segmentation can be seen in Figure 13.

GLAM, RLM and GLCM also show the same behavior with T1 simulated images. Due to the homogeneity within the tissues, the features produced by these methods are able to discriminate boundary with non boundary clusters. As the neighborhood size increases, boundary clusters become larger whilst non-boundary clusters become smaller (Figure 14). The change in the measured accuracy is not related to actual recognition of the tissues, rather than the proper

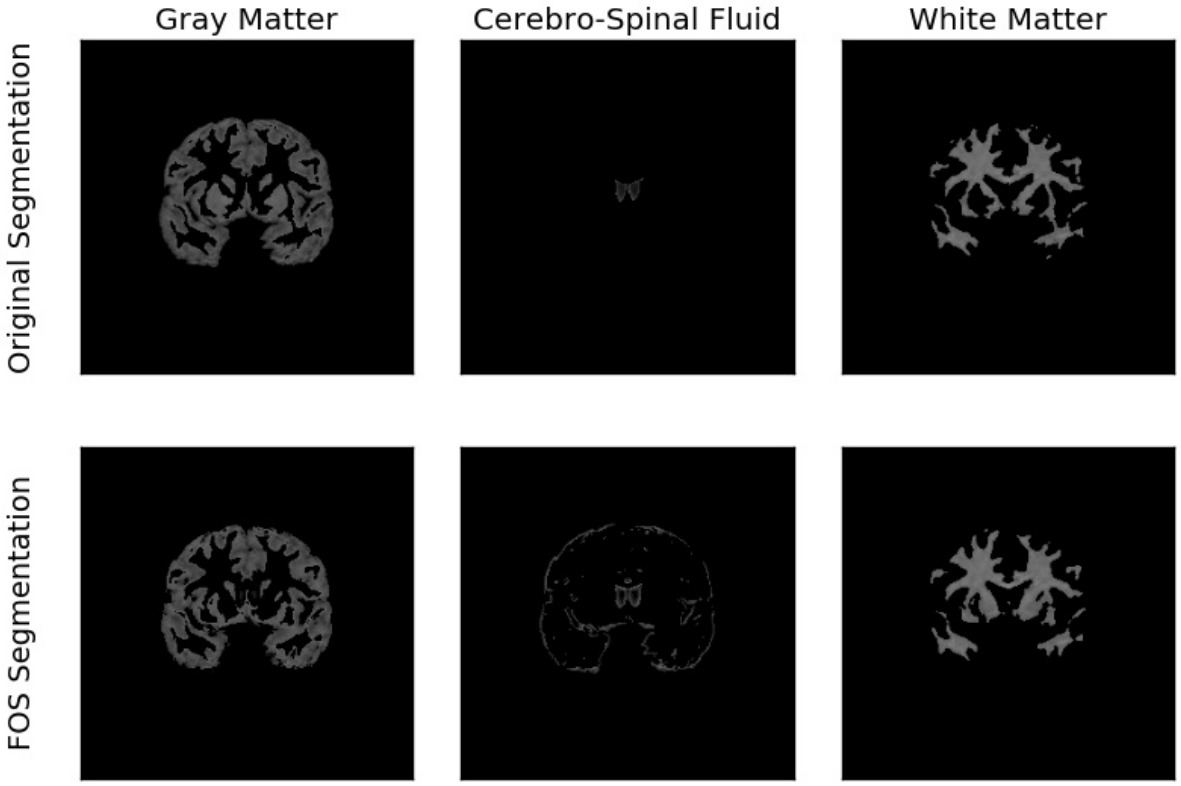

Figure 13: FOS segmentation for T1 weighted real image, Male, Juvenile

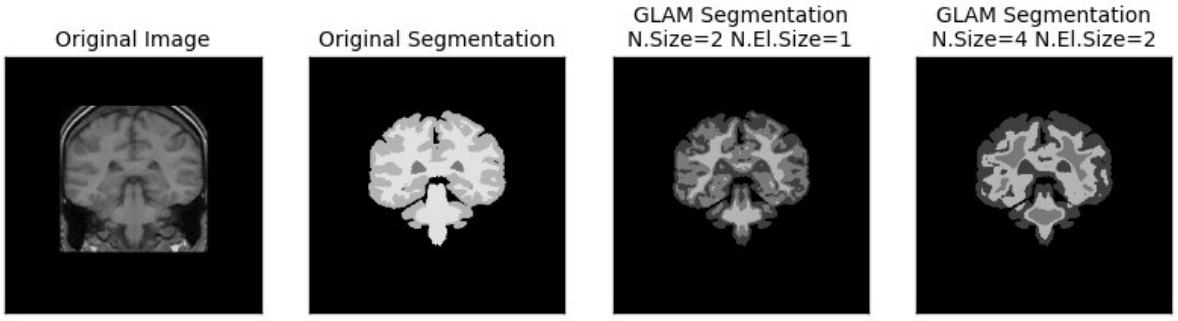

Figure 14: GLAM segmentation for T1 weighted real image, Male, Juvenile.

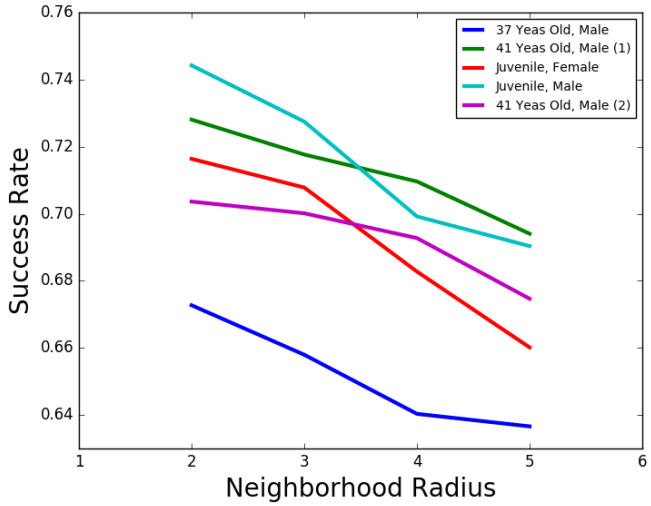

Figure 15: FOS segmentation for T1-Weighted real images

boundary cluster to contain as much GM in combination with minimum White matter.

Through these experiments, it is shown that textural features do not perform well on this task, i.e. segmenting White matter, Gray matter and cerebro-spinal fluid using MR Images. As also mentioned above this is probably due to the similarity in texture of different tissues, and the more discriminative feature being the gray level intensity. Thus features that describe texture can differentiate the boundary regions from homogeneous regions. In this case also the rest of First Order Statistical features should not perform well with these images. In order to test this hypothesis, T1-weighted image is clustered using First Order Statistical features independently. The results can be seen in table 3.

Average Gray-Level Intensity(GLI)	Standard Deviation of GLI	Absolute Gradient Value (AGV)
0.816522	0.527546	0.538247
Average AGV	Standard Deviation of AGV	Combination
0.564737	0.531014	0.802324

Table 3: F.O.S. features accuracy when segmenting T1 weighted image from BrainWeb, Noise = 9%, Bias Field = 40%.

The results shown on table 3 strengthen the hypothesis. From all the above experiments we can conclude that when dealing with healthy brain MR Images, the intensity of the voxels is the most powerful feature. In order to have a visual comparison of the methods with the average intensity, resulted clustering is shown in figure 16. It is only natural that all the existing methods are intensity based whilst sophistication plays part in the segmentation methods and not the features. Since most of the features (if not all) tested in this paper are not helpful for this task, no more experiments are going to be done here, since these datasetscan not provide any more insight on the behavior of the features.

Nevertheless, it is noticeable that the textural features tested can help in the visualization of the different textures in the image. The fact that the performance of the methods with respect to the ground truth is low, is true given that the task in hand is not to identify

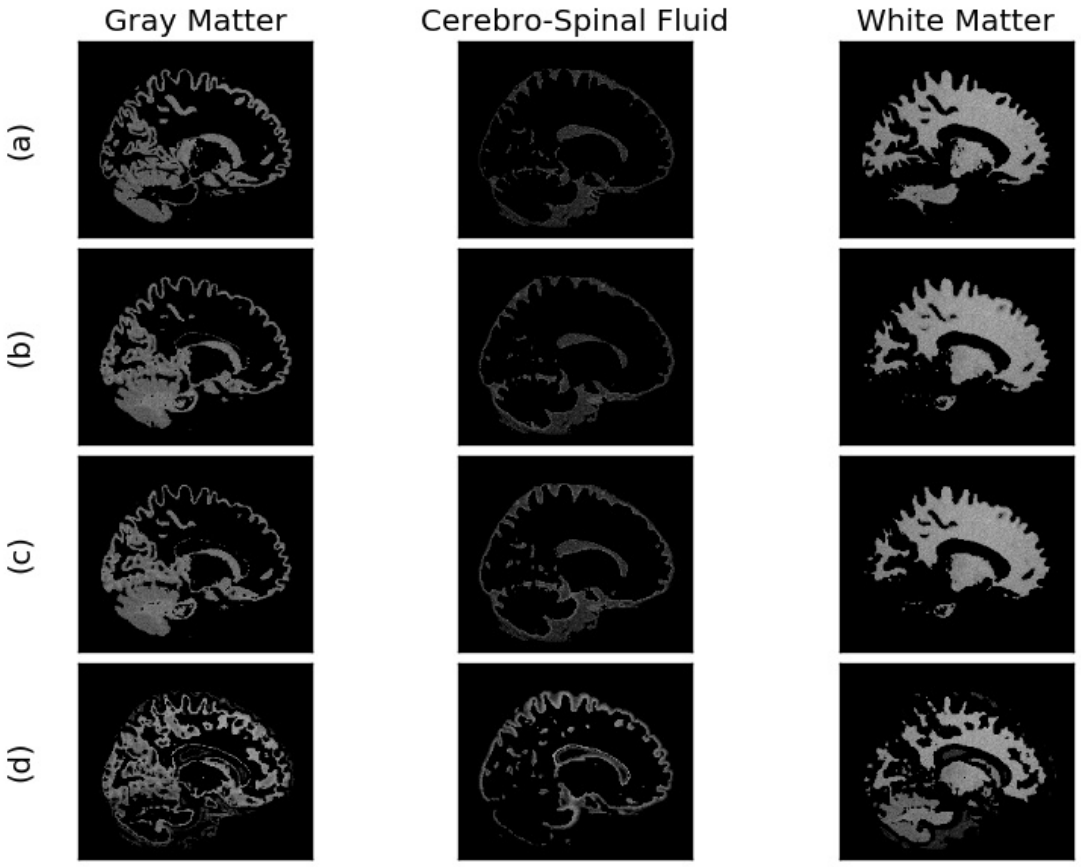

Figure 16: Average gray level intensity, FOS and GLAM segmentations for T1-Weighted image from BrainWeb, Noise = 9%, Bias Field = 40%.
(a) is the Original Segmentation, (b) is the Segmentation of Average Gray Level Intensity, (c) is the Segmentation of FOS features and (d) is the GLAM segmentation.

different textures. The results shown in figures 5, 11, 14 show that the features tested can discriminate the different textures and help visualizing them.

In order to test the feature selection methods Ultrasound Images are going to be used since textural features are more useful for them since they have a lot of speckle noise. Once features selection algorithms are compared, the final system is going to be tested with the MR Images again to see whether the feature selection algorithms will be able to discard and weight features accordingly, so that the final clustering is as close to the one from average intensity as possible.

5.4 UltraSound

5.4.1 Data Sets

As mentioned before, in order to test the system proposed in this project, as well as the 3D textural feature extraction methods, there is a need of 3D datasets with ground truth. Regarding Ultrasound, such a dataset proved to be very hard to obtain. Thus the focus turned to phantom images. There are many proposed methods in creating ultrasound phantom images, using several material, like Gelatine, agar, flower, water and more [66, 67, 68]. Usually one is the main compound and there can be several additives. According to [67], the phantoms

created using Agar-Agar, are very good approximations of real tissues. Thus in this project Agar-Agar phantoms are created with flower as additive. The phantoms were created by Pr. E.M. Bakker, using a Terason T2000+ with 12L5-V Linear Array Transducer from Terason US Scanner [69].

In order to have as general conditions as possible, many blocks with different textures are created. Each block contains $75ml$ of Agar-Agar. Eight different blocks are created, each with different mass of flower in it. The weight of the flower in each case is a multiplicative of $0.2gram$. As a result we have the following blocks.

Block Name	fl1	fl2	fl3	fl4	fl5	fl6	fl7	fl8
Flower Density ($gram/75ml$)	0,2	0,4	0,6	0,8	1,0	1,2	1,4	1,6

Table 4: Flower Concentration of created blocks.

Some example slices from each block can be seen in figure 17.

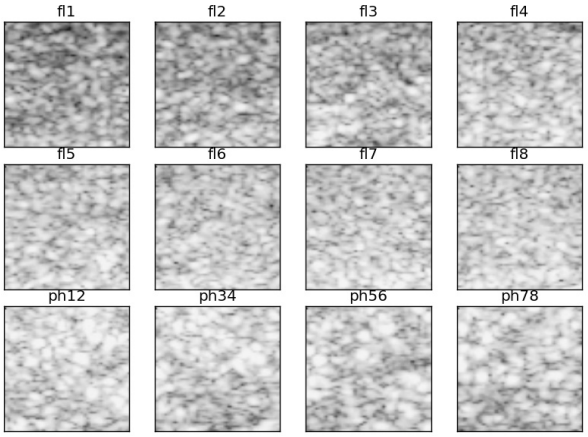

Figure 17: Example 2D slices from each 3D building block.

In addition to these blocks, a Model 539 Multipurpose US Phantom created by ATS laboratories, Inc., USA, is used. The blocks of Agar are put on top of the phantom and a 3D ultrasound scan is produced. In order to have known ground truth, blocks of size 64x64x64 are "cut" from the image and are used as building blocks with which a few sets of images are created. One block for each flower density and four different blocks of the Multipurpose US phantom. With these blocks three sets of images are created. The first has only blocks with Agar-Agar texture, in different combinations. Each image has two different concentrations of agar and each block is used twice. They are placed in an square and the same blocks are on opposite corners of the square. The name of the images comes from their building blocks. In total six images are created with resolution 128x128x64, namely fl1fl3, fl1fl6, fl2fl4, fl2fl7, fl3fl5, fl3fl8. Some example slices from the images as well as their ground truth can be seen in figure 18.

The second set of images is created using combinations of blocks of agar and blocks of the Multipurpose US Phantom. Again four blocks in total, two from the phantom and two from the agar blocks. The two phantom blocks are positioned in opposite corners of a square while on the other two corners are the agar blocks. In this set eight images are constructed, namely ph12fl2, ph12fl5, ph34fl1, ph34fl4, ph56fl3, ph56fl6, ph78fl7 and ph78fl8.

Finally the third set of images consists of more combinations of the agar blocks and the phantom blocks. Six images are constructed in total. Three only with agar blocks, two of

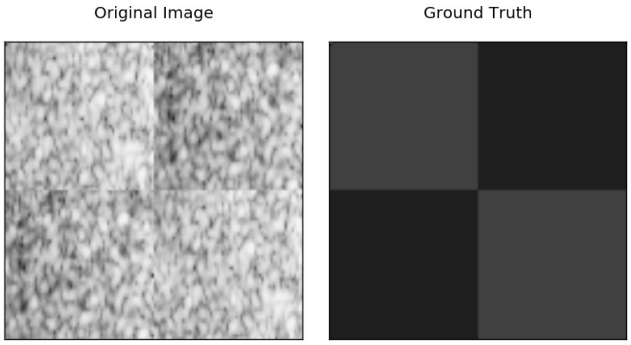

Figure 18: Example 2D slice from fl2fl4.

which have four different blocks and have size 128x128x64 and the third with all eight blocks. Its resolution is 128x128x128. Three more images belong to this set, which have combinations of Multipurpose US Phantom blocks and agar blocks. Again two images have four blocks, two different agar blocks and two different Multipurpose US Phantom blocks. The third has eight blocks, the four Multipurpose US Phantom blocks and four agar blocks. It should be mentioned that the Multipurpose US Phantom blocks have all the same label, i.e. we want to cluster them in the same cluster. The constructed images are fl1357, fl2468, flall, ph1234fl48, ph5678fl26 and phallfl1537.

5.4.2 Textural Method Comparison

The goal of the first part of the experiments, with the Ultrasound dataset, is to compare the methods individually and see how they behave with different parameters. According to previous research, the difference in clustering using different neighboring element sizes of GLAM is negligible. Thus for these experiments it is set to a small value (two), since the time complexity of GLAM is $O(N_S^3)$, where N_S is the neighboring element size. In the case of GLCM and RLM, as the step size increased, the accuracy of the system slightly dropped. Thus the step size for these two is set to one. Moreover, GLAM is a generalization of GLCM, and previous experiments, as well as our previous research show that they have similar behavior. Thus for most of the experiments, where the behavior of the algorithms is tested, only GLAM is going to be used since it has lower time complexity and produces much less features than GLCM, resulting in faster clustering. After choosing the final setup, GLCM is also going to be tested, with the parameters for which GLAM produced the best results. All methods are tested for neighborhood radius from five to nineteen.

Before the results of the experiments are presented, it is important to mention that the accuracy measurement used does not have the same behavior as others. More specifically, when the number of clusters increases, random accuracy increase as well. So in order to avoid any confusion, table 5 shows the accuracies of a random generator, for the set of images constructed in section 5.4.1.

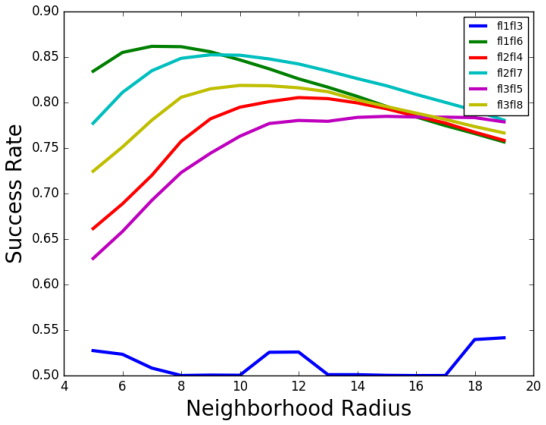

Figure 19: Success rate for neighborhood size from five to nineteen for FOS, for all combinations of flower concentration

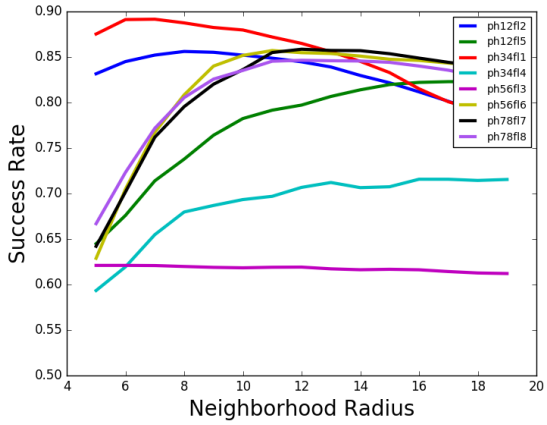

Figure 20: Success rate for neighborhood size from five to nineteen for FOS, for all combinations of flower with phantom blocks.

Image/ set of Images	Random Accuracy
set 1&2	0.5
fl1357, fl2468	0.625001
flall	0.78125
ph1234fl48, ph5678fl26	0.541668
phallfl1537	0.59375

Table 5: Random Generators accuracy for all constructed images.

First, the algorithms are tested on the different combinations of flower concentration in the agar. Figure 19 shows the results for First Order Statistical Features. Almost in all cases, FOS features show the same behavior, with a different maximum. For small neighborhood sizes the accuracy is relatively low, and once the maximum accuracy is reached, it starts dropping again with the increase of the neighborhood size. In two cases the method seems to behave differently. First and most different are the results for the image fl1fl3, i.e. one cluster with flower concentration 1*0,2 and 3*0,2 ($gram/75ml$). Although the method did fail to capture the difference intended, it was able to differentiate the parts were the flower dropped (and as a result higher concentrate), due to gravity, before it froze (right part of each block). The resulted clustering for neighborhood radius nine can be seen in figure 21.

It is interesting to see that, as expected, images with blocks whose difference of the flower concentration is small are more difficult to segment properly, i.e. the algorithm produces the worst results for fl1fl3, fl2fl4 and fl3fl5 whilst for the other three images in which the difference in flower concentration is larger, the accuracy reported is also larger. Besides the small difference in texture, one more reason for this behavior is that when the difference of the flower concentration is small, the difference caused by the movement of the flower to the one side, before it froze, causes bigger difference in the concentration within the blocks than between the blocks. Thus, for these images, the actual performance of the method is better than the one shown in these figures. The strongest example is the one mentioned in the previous paragraph, the results for fl1fl3.

The second set of images is composed by combinations of phantom blocks and flower blocks of different flower concentration. The results for FOS features can be seen in figure 20. Although

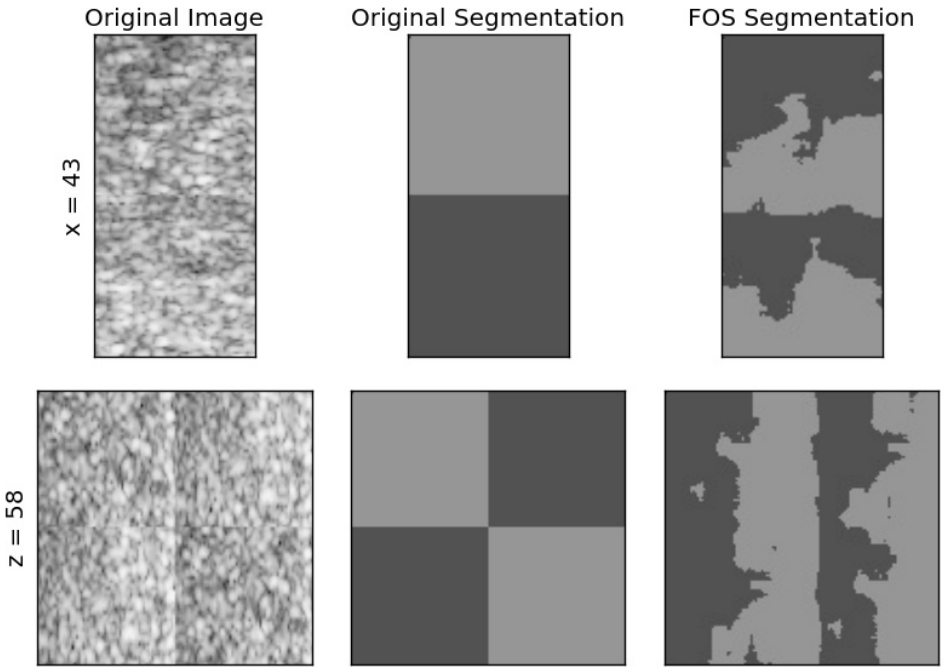

Figure 21: FOS segmentation for fl1fl3 with neighborhood size nine

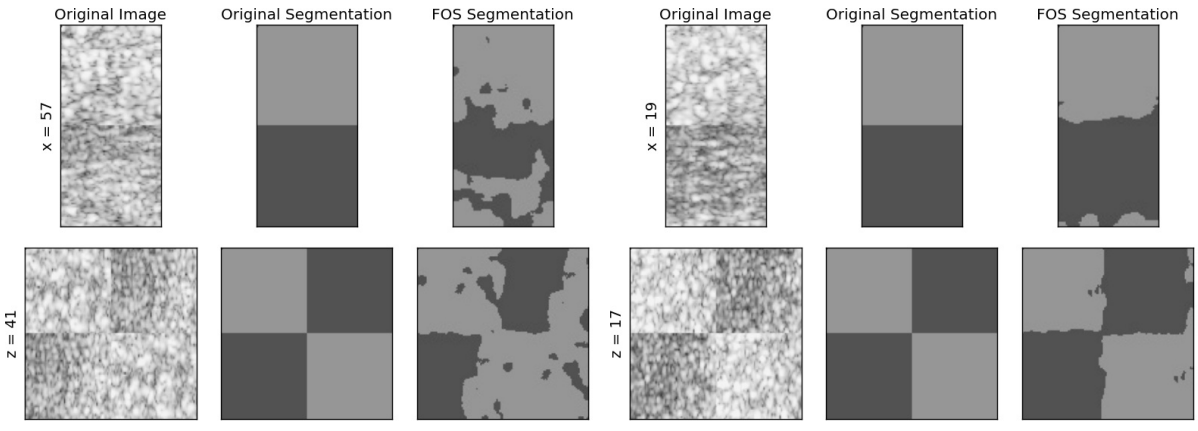

Figure 22: FOS segmentation for ph56fl3 with neighborhood size five

Figure 23: FOS segmentation for ph34fl1 with neighborhood size six

the behavior is more unstable, it looks like it follows the same trend with the first set of images. Again there are two examples which show different behavior, the ph56fl3 and ph34fl4. Even though their performance is a lot worse than the others, the difference in accuracy even exceeds 20%, still FOS features are able to see some difference. Example segmentations can be seen in figures 22, 23. The same effect with the first set of images can be seen with this set as well. The left part of the flower blocks (z-view) have smaller concentration of flower and are clustered with the phantom block. The effect is even visible with the ph34fl1 image (Figure 23) for which the accuracy reaches the highest value, 89,1532%.

It can be noted that, similarly with the MRI segmentation, the clustering of the image produced by the methods can help in visualizing the different textures in the images. In both figures 22 and 23 the clustering of the voxels help to show the difference in texture were the flower "dropped" due to gravity.

In order to compare the difference of the importance of the average intensity with the MR Images, the last experiment done with the MR Images, is repeated. All FOS features are used independently and in combination to cluster an image. Two images are used for this experiment, one that maximized difference in color between the clusters (fl1fl6) and one that minimized it (fl2fl4). The success rates can be seen in tables 6 and 7. Although still the average gray level intensity is the most useful, when combined with the rest of the features the overall accuracy increases a lot. The difference is maximized in the case of fl2fl4, in which the difference in the intensities between the clusters is small and the average value of it seems incapable of discriminating the two clusters.

Average Gray-Level Intensity(GLI)	Standard Deviation of GLI	Absolute Gradient Value (AGV)
0.81143	0.500069	0.503145
Average AGV	Standard Deviation of AGV	Combination
0.697436	0.500484	0.861742

Table 6: F.O.S. features accuracy when segmenting fl1fl6 image.

Average Gray-Level Intensity(GLI)	Standard Deviation of GLI	Absolute Gradient Value (AGV)
0.649274	0.500024	0.501108
Average AGV	Standard Deviation of AGV	Combination
0.681854	0.512458	0.805399

Table 7: F.O.S. features accuracy when segmenting fl2fl4 image.

As mentioned above, GLAM is also tested for neighborhood sizes from five to nineteen, for the same sets of images. The results are shown in figures 24 and 25. When dealing with the first set of images, GLAM also shows a specific behavior similar to FOS features. On the other hand, when dealing with the second set of images, for some of them, there are two local maximum, something that did not occur in any other experiments. It is also interesting to see that the order of difficulty of the images is the same with FOS features, i.e. the order of the images with respect to the accuracies of the segmentations is almost the same for both methods.

The same setup is also used for RLM. The general behavior is similar to the one of GLAM, but there are some exceptions (Figures 26, 27). Especially for images that GLAM does not seem to be able to cluster properly, RLM shows more robustness and the success rate of its

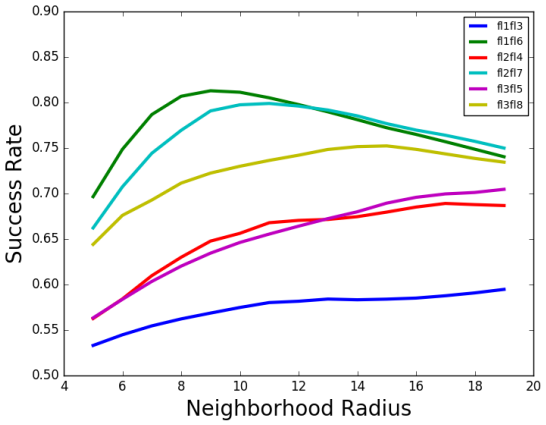

Figure 24: Success rate for GLAM for neighborhood radius from five to nineteen, for all combinations of flower concentration.

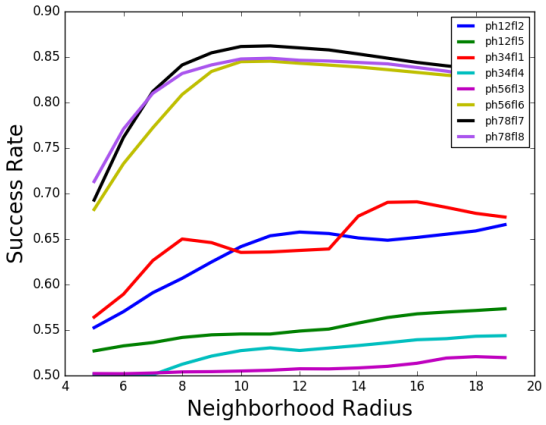

Figure 25: Success rate for neighborhood size from five to nineteen for GLAM, for all combinations of flower with phantom blocks.

produced clustering does not fall as low as the one of GLAM. Still there is one exception. RLM seems completely incapable of segmenting the fl2fl4 image, as we would want it to (Figure 28). On the other hand, there are cases were RLM produced the best clustering of all three methods. These are the ph56fl6 and ph78fl7.

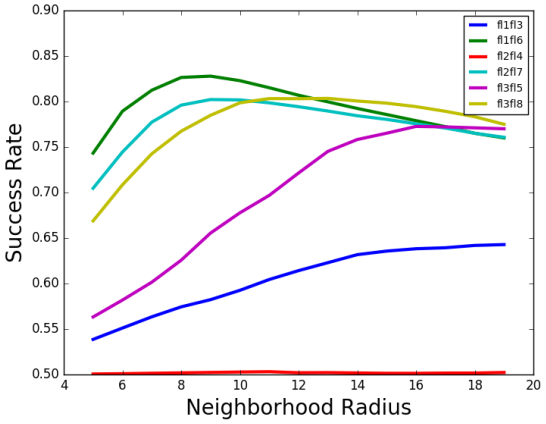

Figure 26: Success rate for RLM for neighborhood radius from five to nineteen, for all combinations of flower concentration.

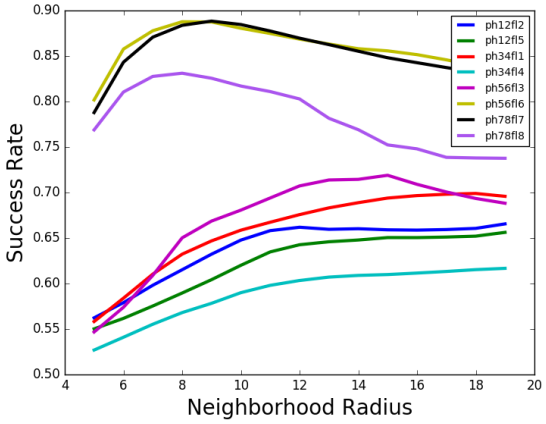

Figure 27: Success rate for neighborhood size from five to nineteen for RLM, for all combinations of flower with phantom blocks.

For the next experiment, GLCM is used for some of the images, with the step size one and neighborhood size the one that GLAM performed best. As mentioned above, the more detailed experiment is avoided due to the large time complexity of GLCM, as well as the similarity in its behavior with GLAM. The images chosen are such so images that GLAM is not segmenting well and images that it does are tested. The images chosen are fl1fl6, fl2fl4, fl1fl3, ph78fl7 ph34fl1 and ph78fl8. Table 8 shows the results for these six images for GLCM, as well as the best configuration of FOS, GLAM and RLM.

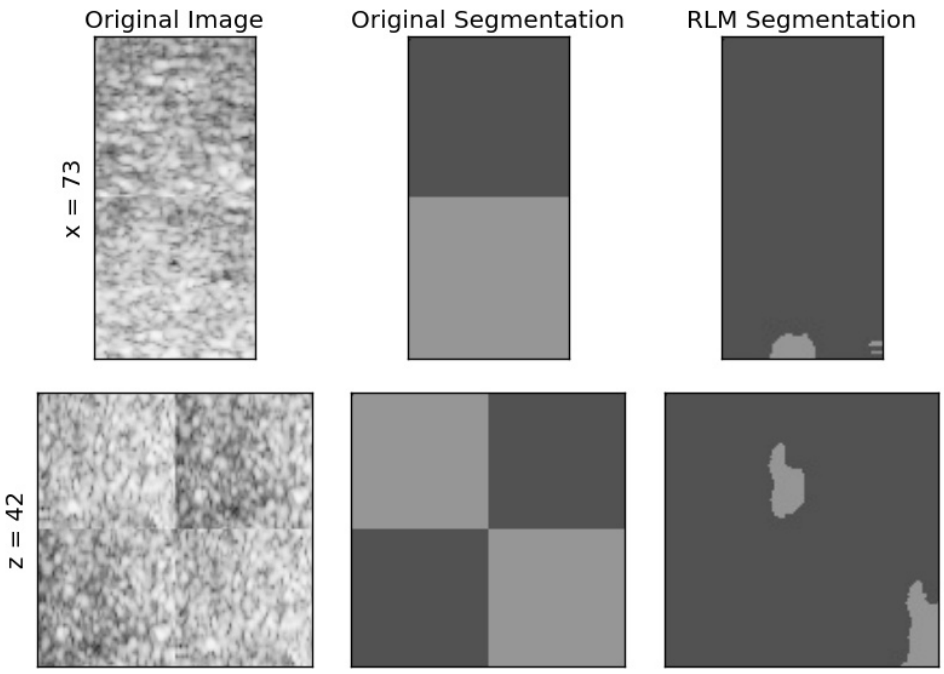

Figure 28: RLM segmentation for fl2fl4 with neighborhood size eight

	FOS	RLM	GLAM	GLCM
fl2fl4	0.805399	0.503078	0.689067	0.705634
fl1fl6	0.861742	0.827873	0.812884	0.802762
fl1fl3	0.541458	0.642769	0.594645	0.568048
ph78fl7	0.857200	0.888358	0.862222	0.85492
ph78fl8	0.846393	0.831083	0.847825	0.813007
ph34fl1	0.891532	0.698958	0.690863	0.714349

Table 8: Best performance of all methods for images, fl2fl4, fl1fl6, fl1fl3, ph78fl7, ph78fl8 and ph34fl1.
The method with the highest accuracy for each image is marked with blue.

Although in some cases GLCM is able to perform better than GLAM, these are only two (fl2fl4 and ph34fl1) and the difference in accuracy is much lower then in the cases where GLAM performed better. For fl2fl4 and ph34fl1 the difference in favor of GLCM is a bit more than 1% whilst the difference exceed even 3% in favor of GLAM for ph78fl8. Considering also the smaller time complexity of GLAM and the great difference in the number of features produced, GLAM seems to be a more interesting approach.

The above experiments show that different features and different methods work better on different images. Even with this set up of images, where the images have the same type of differences in texture (Agar with flower and Phantom image with agar and flower), on different examples different methods and features seem to perform better. Thus, in order to have a system that can perform on many different images, a combination of GLAM, RLM and FOS features should be used. For each method, features for two different setups are extracted and these features are used for segmenting the image, after they go through the redundancy filter and the irrelevancy filter. The setups used are based on the results shown in figures 19, 20, 24 - 27. For GLAM the neighborhood sizes chosen are ten and sixteen whilst for FOS

features and RLM eight and fifteen. It is important to mention that for different applications different setups should be used. For example in the MRI datasets one setup of each method should be used, and with much smaller radii.

The methods and setups, defined in the previous paragraph, are used on the third set of images, which has more combinations of flower concentrations and phantom images than the previous two. The results for these images, for the combination of the methods and setups as well as the methods used independently can be seen in table 9.

	FOS $N_S = 8$	FOS $N_S = 15$	RLM $N_S = 8$	RLM $N_S = 15$	GLAM $N_S = 10$	GLAM $N_S = 16$	Combination
fl1357	0.737397	0.757823	0.708975	0.757800	0.734656	0.775311	0.773662
fl2468	0.709346	0.768114	0.657740	0.754852	0.705196	0.747729	0.765773
flall	0.793944	0.803784	0.765232	0.789245	0.766251	0.802790	0.794024
ph1234fl48	0.642598	0.665933	0.663425	0.670354	0.611121	0.624573	0.686684
ph5678fl26	0.678107	0.693932	0.665791	0.698915	0.659556	0.664971	0.693944
phallfl1537	0.744530	0.770819	0.723215	0.742285	0.699199	0.722342	0.644232

Table 9: Performance of all used setups, independently and in combination for the third set of images.
The method with the highest accuracy for each image is marked with blue.

The behavior of the algorithms is similar to the one from the previous experiments. It can be noted that in most cases, the combination of the methods and their setups produce similar results to the best performing setup. Only in one case the results of the combination were the best of all. The reason behind it is probably the large number of irrelevant features that confuse the clustering. This is the effect that we want to tackle with the feature selection step. Its goal is to discard the redundant and irrelevant features so the data are as separable as possible. It is noticeable that the more different textures the system needs to segment, the closer its accuracy to the one produced by the random generator (Table 5). Especially in the case of eight different flower sets (flall) for some set ups the accuracy is even lower than the one produced by the random generator. Although the resulted clustering usually looks quite random, thats not always the case. For example RLM with neighborhood size fifteen, even though it has slightly higher accuracy than the random generator, it is able to identify some clusters with reasonable performance (Figure 29). Some resulted clustering can be seen in figures 30, 31. Note that these figures show only the clustering of the best performing method for the specific image.

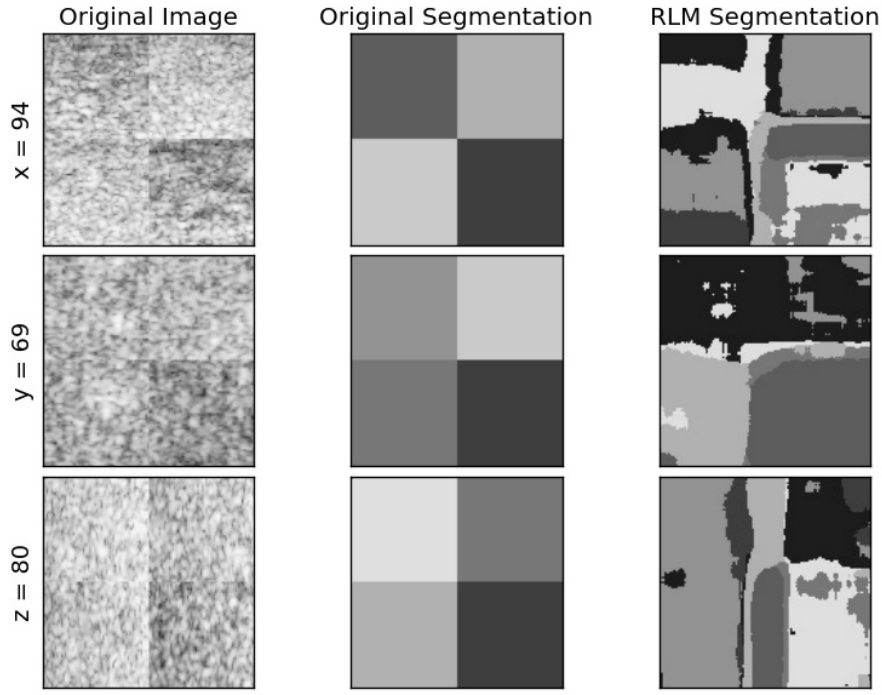

Figure 29: RLM segmentation for flall with neighborhood size fifteen

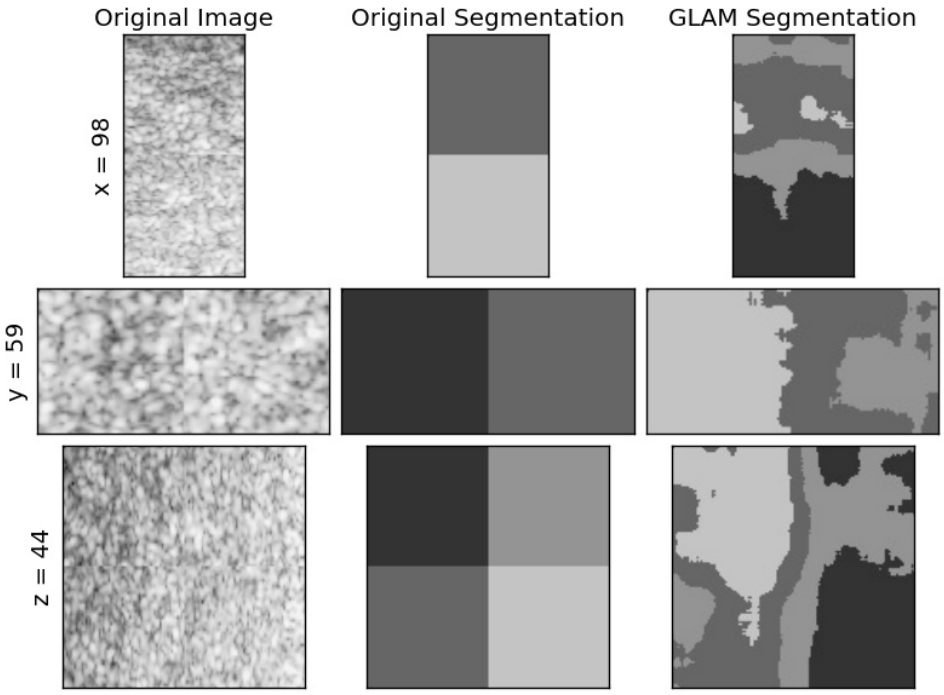

Figure 30: GLAM segmentation for fl1357 with neighborhood size sixteen

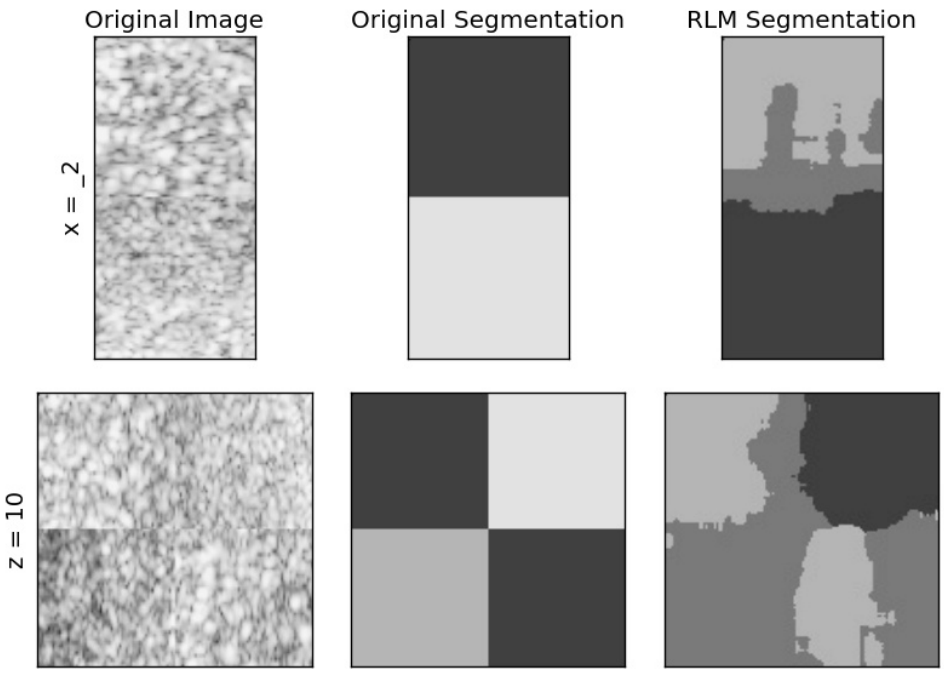

Figure 31: RLM segmentation for ph5678fl26 with neighborhood size fifteen

5.4.3 Unsupervised Feature Selection Filters

In this section, experiments are done in order to explore the behavior as well as the performance of feature selection techniques. Since most of them have very high computational complexity, most of the experiments are done with small images and small number of features. The experiments are organized as following. First, using the third set of images while using all the textural feature extraction methods and their setups decided by the previous section, the variations of Mitra's et al. algorithm are tested for several threshold values. The goal of this experiment is to measure the percentage of features remaining for a specified threshold. The results are not meant to show the quality of the filters results, rather than using the resulted value in order to have a general enough experiment for the irrelevancy filters, without having to test them with an insufficiently large number of features, since the time complexity of most of the methods discarding irrelevant features is $O(d * N^2)$, where d is number of features and N is the number of data points. The second experiment is meant to show how good estimation of the rankings of RANK, can SRANK produce for different number of patch sizes. For this experiment a small dataset is used, one image with two classes and FOS features.

The results of the first experiment, for the first variation of Mitra's et al. algorithm can be seen in figure 32. The results show a specific behavior of the algorithm. For all images, there is a great number of very similar features with similarity less than 0.01. The number of features left after this filter with threshold 0.01 is on average 72.5 with a minimum 62 and maximum 85.

In order to see whether the sets of features kept for each image are similar, the similarity between the sets of features kept is measured. The similarity is defined as the ratio of the features kept for both tested images over the number of all the occurrences of features, for both images.

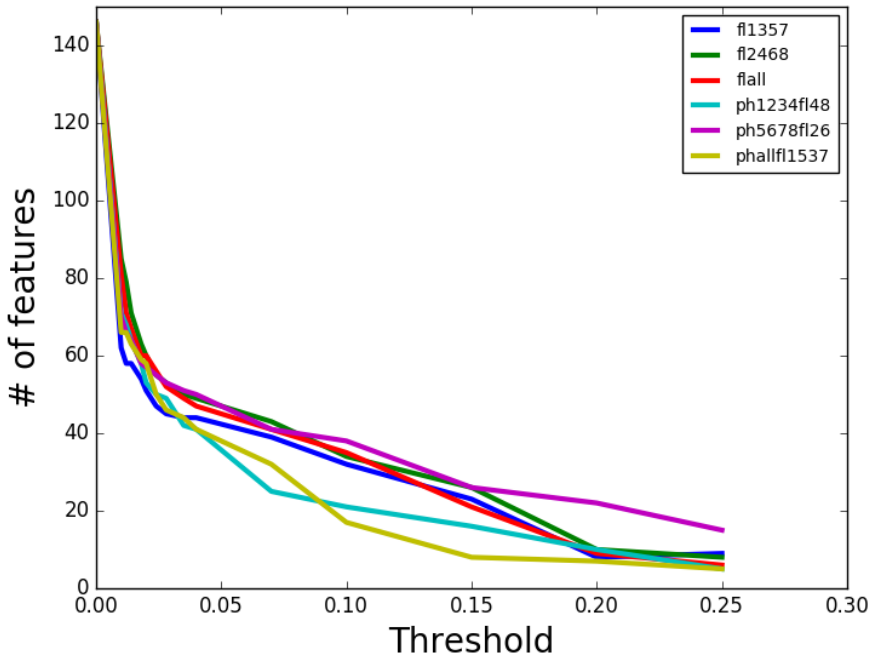

Figure 32: Number of features left after the variation of Mitra's et al. algorithm is applied for different threshold values, for the third set of images.
The value for threshold zero is the initial number of features.

$$Similarity = \frac{Number\ of\ Same\ features}{Number\ of\ unique\ features\ in\ both\ images} \tag{26}$$

The results shown in table 10 are what was expected. For every image the redundant features are not only different in number but also there is a difference in which features are discarded and not. For small values of the threshold, the sets features that are kept seem very similar. The reason behind this is that for these values of the threshold, on average, more than half of the features are kept and thus it is expected for different images to have a big number of the same features. Moreover, it is worth mentioning that the sets of features kept for images that contain similar building blocks (fl1357 with flall or fl2468 with flall), on average, have higher similarity than pairs of images with different building blocks.

threshold	0.01	0.012	0.014	0.018	0.02
Similarity	0.745 ± 0.083	0.737 ± 0.071	0.749 ± 0.106	0.717 ± 0.122	0.701 ± 0.109
threshold	0.024	0.028	0.035	0.04	0.07
Similarity	0.692 ± 0.116	0.686 ± 0.118	0.655 ± 0.132	0.63 ± 0.175	0.447 ± 0.204
threshold	0.1	0.15	0.2	0.25	
Similarity	0.365 ± 0.179	0.254 ± 0.128	0.276 ± 0.155	0.286 ± 0.216	

Table 10: Average similarity of kept features for all combinations of images for all tested thresholds.

The goal of the second experiment is to show us how good approximation of the results of RANK does SRANK provide, for different patch sizes. For this experiment the number of

patches has been set to $Patch_{size}^{-1}$, where the $Patch_{size}$ is the proportion of the number of data points in the Patch to the number of points in the data set. Thus, the total number of random data points chosen is the same with the number of data points in the data set. Given that the time complexity of the algorithm is $O(Patch_{points}^2 * N_{patches})$, the smaller the patch size the faster the algorithm. Given our use of RANK and SRANK, what we are interested in, is not the absolute values of the rankings, but their relationship, since the weights given to the features is proportional to the difference of the rankings. Thus the results shown are the weights calculated from the rankings produced by the experiment. For each $Patch_{size}$ the experiment is done ten times and the results given are the average results. The image used for this experiment is fl1fl6, since the results of clustering of the individual FOS features for this image are already available from previous experiments (Table 6). The results for the weighting scheme proposed in this paper can be seen on table 11, and the results for the weighting scheme proposed by Li et al. [41](≡ original) on table 12.

	$Feature_1$	$Feature_2$	$Feature_3$	$Feature_4$	$Feature_5$
0.001	1 ± 0	0.2 ± 0	0.9098 ± 0.0021	0.3793 ± 0.0018	0.2425 ± 0.0017
0.005	0.9331 ± 0.0035	0.2 ± 0	1 ± 0	0.4411 ± 0.0016	0.241 ± 0.0015
0.01	0.9109 ± 0.0027	0.2 ± 0	1 ± 0	0.4509 ± 0.0012	0.2396 ± 0.0008
0.02	0.9054 ± 0.0024	0.2 ± 0	1 ± 0	0.4582 ± 0.0016	0.2393 ± 0.0018

Table 11: Average Weights for the proposed weighting scheme for different $Patch_{size}$s of sRank. The columns represent each feature and the rows the different $Patch_{size}$s. The order of the features is the same with Table 6.

	$Feature_1$	$Feature_2$	$Feature_3$	$Feature_4$	$Feature_5$
0.001	0.462 ± 0.0012	$\approx 10^{-8}$	0.4099 ± 0.0008	0.1035 ± 0.0008	0.0246 ± 0.0009
0.005	0.4039 ± 0.0017	$\approx 10^{-9}$	0.4407 ± 0.0006	0.1328 ± 0.0009	0.0226 ± 0.0008
0.01	0.3946 ± 0.0007	$\approx 10^{-9}$	0.4441 ± 0.00097	0.1393 ± 0.0005	0.02198 ± 0.00042
0.02	0.3913 ± 0.001	$\approx 10^{-10}$	0.4437 ± 0.0009	0.1432 ± 0.0007	0.0218 ± 0.00097

Table 12: Average Weights for the weighting scheme proposed by [41] for different $Patch_{size}$s of sRank. The columns represent each feature and the rows the different $Patch_{size}$s. The order of the features is the same with Table 6.

Unfortunately, the results for RANK are not available, since for one image it needed more than 24 hours to finish, which is the time limit for processes on the machine that all the experiments are done. The machine has 32 cores, each having its clock speed at 3.3GHz. Since that time limit is exceeded for a small image for only five features, RANK is considered as extremely expensive and inefficient. Unfortunately the computation of FFEI has similar computational complexity and thus the Neuro-Fuzzy approach for discarding irrelevant features, as well as the complete approach (not the simplification introduced here) for the irrelevance filter of the two layer filter approach are as well considered as inefficient and unable to be tested properly. Moreover, similar time complexity has the second variation of Mitra's et al. algorithm which computes the entropy for every feature. Since the first variation has only time complexity $O(N)$ (instead of $O(N^2)$) and its results are satisfying, experiments for the second variation are avoided.

Since it is not possible to compare the results of SRANK with the ones of RANK, the results for Patch size 0.02 are considered as the closest to the RANK results and the rest are compared to them. For all Patch sizes, except the 0.001, the order of the features is the same. Moreover, with the exception of feature 3 (Absolute Gradient Value) the order of the features and their weights are consistent with tables 6 and 7, i.e. features that produce clustering with high accuracy, with respect to the ground truth, get higher weights than features that produce clustering with low accuracy. Regarding the Absolute Gradient Value, the difference between its performance recorded by tables 6, 7 and the very high weight it gets from the filter is caused by the fact that it is the only feature not computed in a neighborhood around the voxel, rather that the immediate neighbors of the voxel. Although it produces highly separable clusters int the feature space, these clusters are not separable in the image space. Thus, although it should be considered as a "bad" feature, the algorithm is not capable of differentiating between the two. Since the rest of the features are computed for a neighborhood radius 7 (Neighborhood size $= (2 * 7 + 1)^3$), their values will change much smoother while we "move" in the image. Considering the completely different behavior of this feature from the rest, as well as the fact that it "confuses" the irrelevance filter, and does not produce a quality clustering, it is discarded completely.

Regarding the stability of the algorithm, in every separate "run" of the algorithm (the results are averaged over 10 "runs"), the order of the features is always the same (the standard deviation is three to four orders of magnitude smaller than the actual values). Although this is the case, the "bad influence" of the small value of the Patch sizes is visible. For both weighting schemes the weights vary a lot, almost 10% in the proposed weighting scheme and 6% in the original. Considering the different scale of the weights (the maximum value of the proposed weighting scheme is 1 whilst the maximum value of the original weighting scheme is 0.462) the variation of the weights is very similar. Since the original weighting scheme always weights a feature with an extremely small value (no matter how worse or not from the other features it is), the weighting scheme proposed in this paper is chosen for the rest of the experiments.

The next and final experiment of this section has as goal to provide an insight on the threshold of the first filter, responsible for discarding redundant features, for which the features discarded are truly redundant. In order to do that, the whole system, with the methods and parameters decided from the previous experiments, is going to be used on some images of the first, second and third set of images with varying thresholds of the first filter. The images used are fl1fl6, fl3fl5, ph56fl6, ph56fl3, fl1357 and ph5678fl26. For this experiment the Patch size of the sRank algorithm is set to 0.005 and considered fifty patches in total. The smaller number of patches is chosen due to the fact that for large images with many features, even with a patch size so small sRank needs a lot of time to finish. Given that the standard deviations of the weights of the previous experiment are much smaller than the actual weights, it is considered as an acceptable approximation, given the trade off with the time complexity of the system. The results of the experiment can be seen in table 13.

	fl1fl6	fl3fl5	ph56fl3	ph56fl6	fl1357	ph5678fl26
0.07	0.821824	0.798668	0.690129	0.555909	0.762047	0.682975
0.1	0.841162	0.825185	0.691261	0.566535	0.736425	0.70144
0.15	0.835845	0.816565	0.701919	0.853661	0.709364	0.825097
0.2	0.779208	0.65205	0.738894	0.853738	0.738654	0.800996

Table 13: Success Rate for different thresholds of the redundant filter, for images fl1fl6, fl3fl5, ph56fl6, ph56fl3, fl1357 and ph5678fl26.

The table above shows inconsistency of the behavior of the filters with respect to the accuracy measured for each image. Although threshold 0.15 seems to demonstrate the best performance over all, still for image fl1fl6 it produces the worst results of all. Moreover, in some cases the filters seem to rescale the features in a way that the data are no longer separable, even in cases that the features usually work well. The most extreme example is ph56fl6, for which, with small value of the threshold the accuracy of the resulting clustering is even 30% lower than for greater values of it and the individual methods. In other cases they are even capable of improving the accuracy of clustering up to 10.6% on ph56fl3 and 12.62% on ph5678fl26. Based on these results the threshold of the redundancy filter is set to 0.15

5.4.4 Validating Two Layer Filter

In order to validate the two layer filter, with input methods and parameters the ones decided in by the previous experiments, one last experiment is done. In this experiment the accuracy of the clustering produced by the resulted system, is compared with the individual performances of the methods and their setups involved as well as their combination, without the use of filters. The results can be seen in table 14.

	FOS $N_S = 8$	FOS $N_S = 15$	RLM $N_S = 8$	RLM $N_S = 15$	GLAM $N_S = 10$	GLAM $N_S = 16$	Combo	Overall
fl1fl6	0.861331	0.795643	0.826415	0.785667	0.811305	0.764938	0.500000	0.835845
fl3fl5	0.723041	0.784788	0.625583	0.765159	0.646198	0.695791	0.522856	0.816565
ph56fl3	0.619796	0.616701	0.650175	0.718919	0.504925	0.513483	0.538315	0.701919
ph56fl6	0.808422	0.847512	0.887659	0.855643	0.844839	0.833073	0.565468	0.853661
fl1357	0.737397	0.757823	0.708975	0.757800	0.734656	0.775311	0.773662	0.709364
ph5678fl26	0.678107	0.693932	0.665791	0.698915	0.659556	0.664971	0.693944	0.825097

Table 14: Success Rate for different methods, as well as their combination without filters and with filters, for images fl1fl6, fl3fl5, ph56fl6, ph56fl3, fl1357 and ph5678fl26.
Blue is the overall best accuracy for an image, yellow between the combination without and with filter and green if both previous are true.

The system proposed, when using two different scales from each textural feature extraction method and the two layer filter, although not always capable of producing the best result, its result is still comparable to the best accuracy measured. For most images that it did not produce the best result, the difference with the best is 1% - 3%. Only for one example, the accuracy of this system is lower than the accuracy of the combination of textural feature extraction methods without filters. It should be noted that the image for which this is true, is the image that contains blocks that maximize the within blocks textural difference due to the gravity effect, explained before. This image is the fl1357. With respect to the results from the individual experiments, we can conclude that the system demonstrates a level of robustness when dealing with different sets of images, as for most images, the results are comparable to the best individual method, and in some cases are even better. Some examples segmentations can be seen on figures 33, 34.

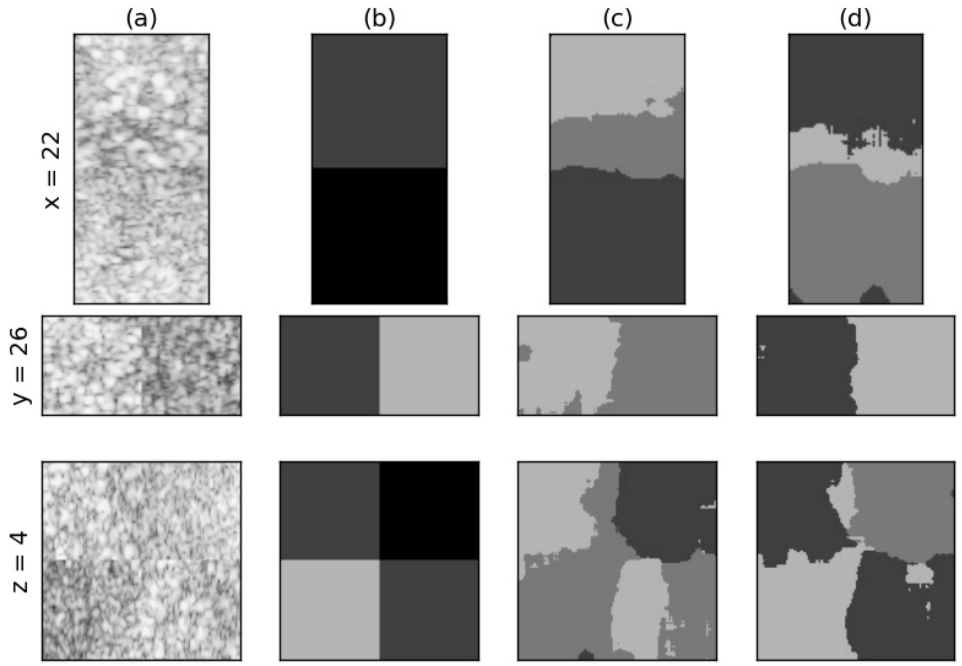

Figure 33: Segmentation of ph5678fl26 for RLM with Neighborhood radius 15 and the two layer filter
(a) is the original image, (b) is the original segmentation, (c) is the segmentation produced by RLM and (d) the segmentation produced by the two layer filter.

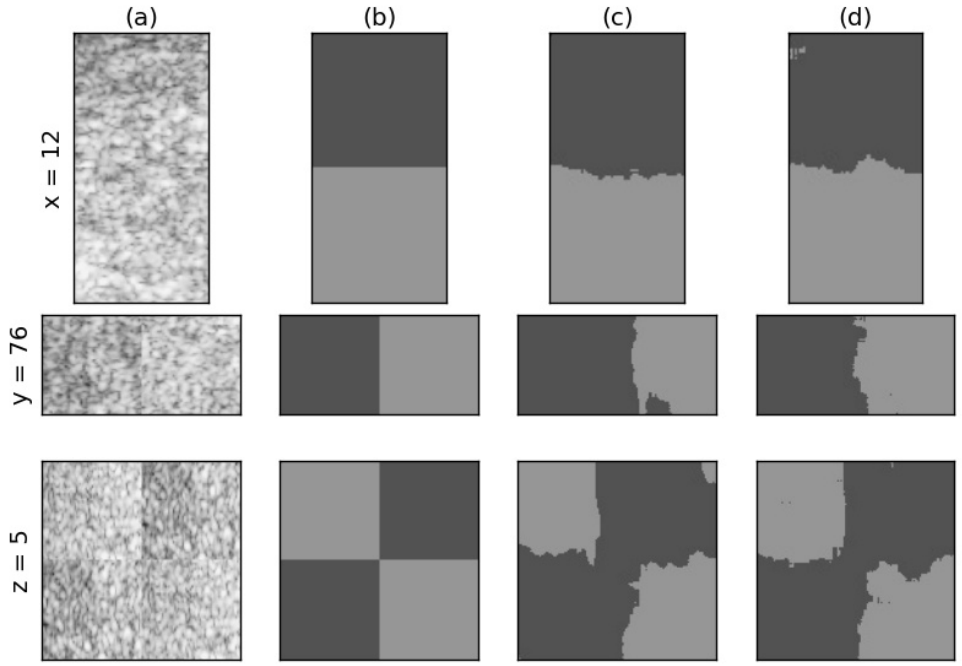

Figure 34: Segmentation of fl3fl5 for FOS features with Neighborhood radius 15 and the two layer filter
(a) is the original image, (b) is the original segmentation, (c) is the segmentation produced by RLM and (d) the segmentation produced by the two layer filter.

5.5 Two Layer Filter on MR Image

The last experiment of the previous section is also done on one of the MRI phantom images from BrainWeb, a T1-weighted image with inhomogeneity level 40% and Gaussian noise 9%. Due to the big difference of behavior of the textural feature extraction methods on MRI and UltrSound, the setups, of each method used are different, i.e. one setup from each method is used, all with radius two.

FOS $N_S = 2$	RLM $N_S = 2$	GLAM $N_S = 2$	Average Intensity	Combo	Overall
0.802324	0.575625	0.575375	0.816522	0.584268	0.564531

Table 15: Success Rate for different methods, as well as their combination without filters and with filters, for T1 weighted MR Image from BrainWeb, Noise = 9%, Bias Field = 40%.
Blue is the overall best accuracy for an image, yellow between the combination without and with filter and green if both previous are true.

The above table shows that the overall system was not only incapable of producing results as close as possible to the performance of the average intensity, but also the performance is even lower that the combination of methods without the use of filters. One possible explanation is that although the features of RLM and GLAM, are not capable of producing the desirable segmentation, still they are very "confident" about the segmentation they produce, i.e. the difference in feature values between the homogeneous regions and boundary regions are quite clear. Thus, since the filters are developed for unsupervised feature selection, they don't "know" the difference between the desired result and an other, also clear, segmentation. For that reason, one more experiment is done, using the filters, but this time using only FOS features. The resulted accuracy is 0.818102, which is even higher than the accuracy of the average intensity, which until this point produced the best performance measured. As a result we can conclude that with smart decision of the textural methods and their parameters, the system developed can provide a steady in quality performance. Still, as seen from previous experiments, for every image not only different parameters of the methods are needed but also different threshold of the filter responsible for removing redundant features and different $Patch_{size}$ of sRank produces optimum results. Some example slices of the produced clusterings of the final experiment can be seen in figures 35 - 37.

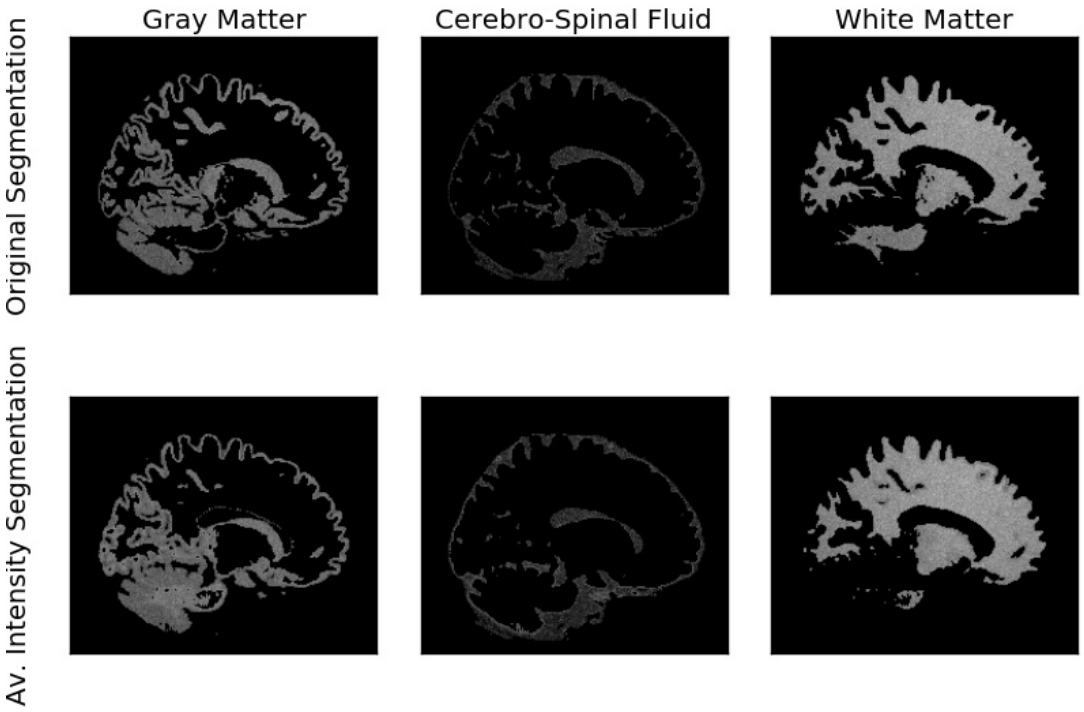

Figure 35: Example Segmentation of Average intensity on BrainWeb image, Noise = 9%, Bias Field = 40%.

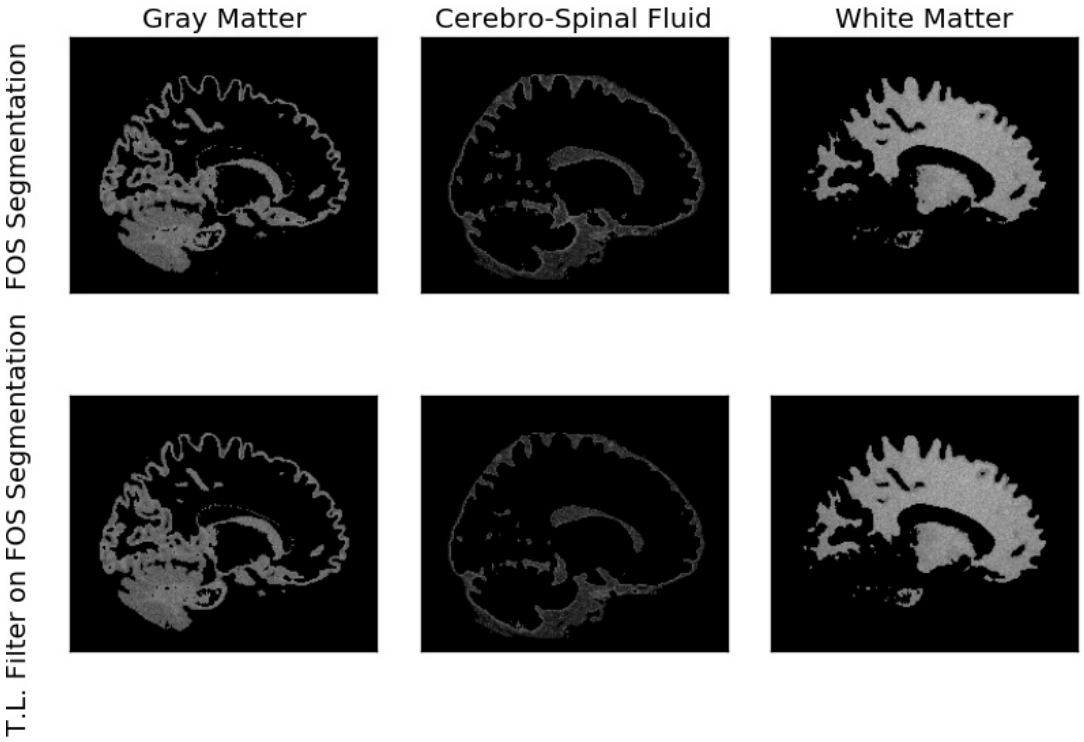

Figure 36: Example Segmentation of FOS features and the Two Layer Filter applied on FOS features on BrainWeb image, Noise = 9%, Bias Field = 40%.

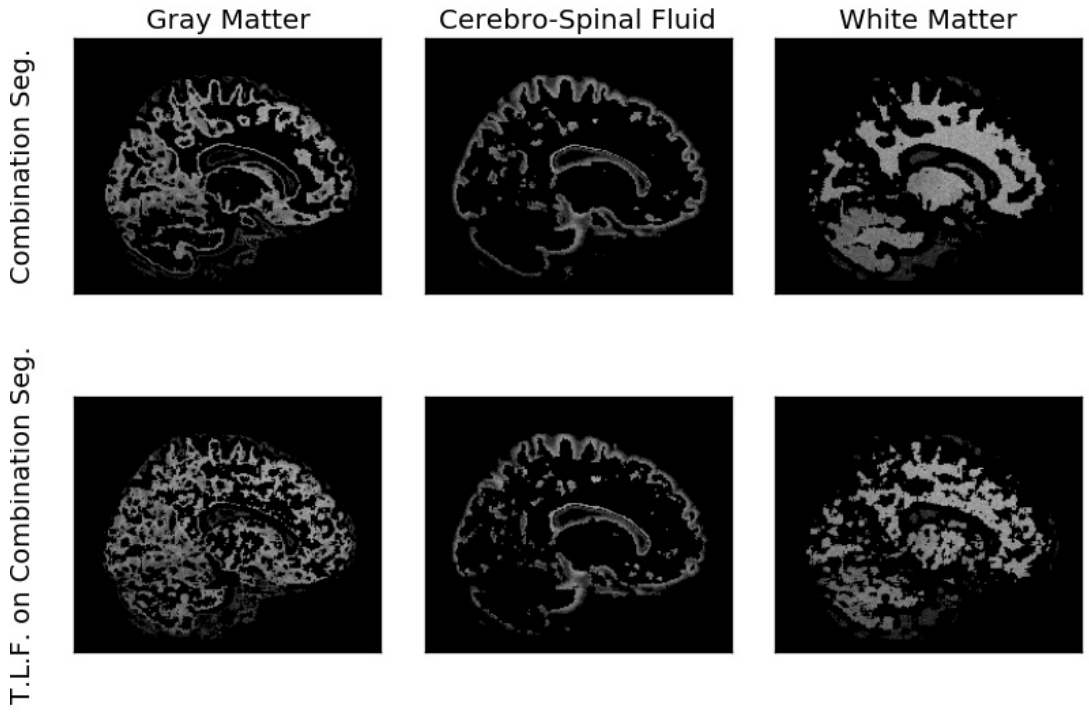

Figure 37: Example Segmentation of Combination of methods with and without the Tow Layer Filter on BrainWeb image, Noise = 9%, Bias Field = 40%.

5.6 Unsupervised Voxel Clustering

As mentioned above there are more than one clustering algorithms considered in this project. All the above results are produced using K-Means++. Nevertheless, all the experiments done until now also produced results for Fuzzy C-Means (FCM) with no modifications. As initial centers, the best centers of K-Means++ are provided. The behavior of the algorithm is quite unstable. There are a lot of examples were the accuracy either did not change, changed a little (higher and lower) and in many cases it significantly dropped, with respect to the results of K-Means++. An example figure that demonstrates the above observation can be seen in figure 38. More examples are not presented since the FCM algorithm demonstrates very similar behavior for the rest of the experiments.

One of the reasons of the unstable behavior, is that FCM is extremely sensitive to the initial centers. Thus in order to make it more stable, instead of just providing one set of initial centers, the best that K-Means++ produced, more sets should be provided, and the set that minimizes the cost function should be chosen. Since even in the good scenario FCM does not provide much improvement on the K-Means++ algorithm, it is avoided since the risk of getting far worse results, is not eliminated.

The FCM with the modification of the neighborhood attraction, is avoided for two reasons. One, the dependency of FCM to the initial centers remains on this variation as well. Moreover, since our features are calculated for a neighborhood around the voxel, the speckle noise that this variation tries to tackle, is in most cases not present and in most cases that is present it is negligible (See all figures with example segmentations).

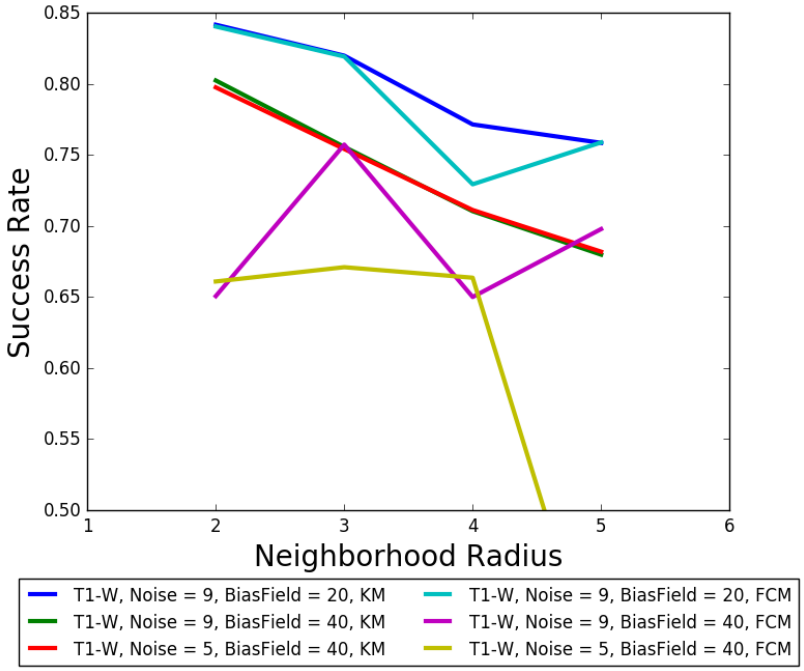

Figure 38: Accuracy for FCM and K-Means on T1-weighted images from BrainWeb, using FOS features.

To conclude, KMeans++ demonstrates the best average performance for all images, and proved to be the most robust. Thus, KMeans++ is our choice for unsupervised image segmentation using the textural features tested in this project, considering the algorithms explored.

6 Conclusion

In this paper several texture analysis techniques were studied, in order to examine how they behave in different images and textures. Then, the findings of the experiments were used in order to build an unsupervised system that segments 3D medical images. According to the results, different textures are better discriminated by using different methods, as well as different setups, i.e. different scales. In order to be able to build a generic system many of the feature extraction methods need to be used, with many different setups, making as certain as possible that the methods and setups best qualified for the specific image are used. After experimentation we concluded that for Ultrasound images, FOS features, GLAM and RLM need to be used, each for two different scales. Our experiments showed that these methods can provide close to the best performing individual method of each image. For different images, i.e. different textural differences, the method and scale closer to the best is different and thus all of them have to be used for a more generic system. For clustering White matter, Gray matter and Cerebro-Spinal Fluid using MR Images, the best performance is achieved using only FOS features with radius two, i.e. one scale. This set of features (FOS features) is the only set capable of separating these tissue types, as there is very small difference in texture and as a result the other methods separate boundary from non-boundary regions.

Moreover, we show how the produced clusterings help visualize different 3D textures in an image. For example, when segmenting 3D MRI scans the segmentation produced by GLCM, GLAM and RLM show the different textures within the brain. When dealing with 3D Ultrasound

images, of Agar with flower blocks, the visualization of the segmentation help to show the difference when the flower "drops" due to gravity to the bottom side of the block.

Due to the large number of features that this approach produces, many of them are irrelevant and different for each image. Thus a need arises for an unsupervised feature selection algorithm that can tackle this problem and discard the irrelevant features. There are many proposed methods that try to do feature selection in an unsupervised manner. Some of these methods, were studied and variations of them, that tackle very specific problematic behaviors of them, according to our needs, were proposed and tested. The results showed the strengths as well as the weaknesses of these methods. Our variation of Mitra's et al. algorithm coupled with our variation of the Irrelevancy filter in a Two Layer Filter seems to have the overall best performance, with respect to time complexity as well as the performance of the features kept. Moreover the resulted filter is able to produce comparable results to the best individual method and in some cases even better. The accuracy of the segmentation with respect to the ground truth is 1-3% lower than the accuracy of the best performing individual method and in some cases it achieves even 12.65% higher accuracy. Additionally, it produces better results than the combination of textural feature extraction methods without the use of filters. Thus, with smart decision of the textural feature extraction methods and their parameters, the system proposed in this provides a level of robustness, with respect to the individual textural feature extraction methods. The experiments show that this system (the feature extraction methods described in the previous paragraph with the Two Layer Filter) using KMeans++ for the unsupervised clustering of the voxels, achieves the best, on average, performance. Although this is the case, for different types of images a generic selection of textural feature extraction methods and their setups is still needed. For example, different setups of the textural feature extraction method performs better for MRI and different for Ultrasound.

7 Future Work

The textural methods studied in this paper show potential in discriminating different textures. These findings can be used with other supervised and unsupervised, as well as application specific, systems in order to raise their performance. Moreover, regarding the system developed, it would be interesting to try and make use of the fact that the segmentation purpose is image segmentation in the feature selection step. As the feature selection algorithm computes a measurement similar to entropy for the features, an extension of it, that makes use of the aforementioned fact, can be introduced and as a result have a more application specific feature selection algorithm. An example use of that fact can be the incorporation of the spatial distance of points in the image to the measurement. This might work due to the fact that different objects in an image are expected to be composed by neighboring voxels.

In terms of trying to make the system more generic, a method that tries to find the optimal number of clusters can also be used. In such a case, the system would be able to decide how many objects it can see with reasonable cluster separability.

References

[1] I. Despotovic, B. Goossens and W. Philips *MRI Segmentation of the Human Brain: Challenges, Methods and Applications* Computational and Mathematical Methods in Medicine, Vol. 2015.

[2] S. Shen, W. Sandham, M. Granat and A. Sterr *MRI Fuzzy Segmentation of Brain Tissue Using Neighborhood Attraction With Neural-Network Optimization* IEEE Transactions on information technology in biomedicine, Vol. 9, No. 3, 2005.

[3] K. V. Leemput, F. Maes, D. Vandermeulen and P. Suetens *Automated Model-Based Bias Field Correction of MR Images of the Brain* IEEE Transactions on medical imaging, Vol. 18, No. 10, 1999.

[4] K. V. Leemput, F. Maes, D. Vandermeulen and P. Suetens *Automated Model-Based Tissue Classification of MR Images of the Brain* IEEE Transactions on information technology in biomedicine, Vol. 18, No. 10, 1999.

[5] N.I. Weisenfeld, S.K. Warfield *Normalization Of Joint Image-Intensity Statistics In MRI Using The KullBack-Leibler Divergence* IIEEE International Symposium on Biomedical Imaging, 2004.

[6] J. Ashburner and K.J. Friston *Unified segmentation* Welcome Department of Imaging Neuroscience, 2005.

[7] Y. Zhang, M. Brady, and S. Smith *Segmentation of Brain MR Images Through a Hidden Markov Random Field Model and the Expectation-Maximization Algorithm* IEEE transactions on medical imaging, Vol. 20, No. 1, 2001.

[8] B. Fischl, D.H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, A. Montillo, N. Makris, B. Rosen and A.M. Dale *Whole Brain Segmentation: Automated Labeling of Neuroanatomical Structures in the Human Brain* Neuron, Vol. 33, pp. 341-355, 2002.

[9] M. Sezgin and B. Sankur *Survey over image thresholding techniques and quantitative performance evaluation* Journal of Electronic Imaging, vol. 13, no. 1, pp. 146-168, 2004.

[10] N. Passat, C. Ronse, J. Baruthio, J.P. Armspach, C. Maillot, and C. Jahn *Region-growing segmentation of brain vessels: an atlas-based automatic approach* Journal of Magnetic Resonance Imaging, vol. 21, no. 6, pp. 715-725, 2005.

[11] T. Weglinski and A. Fabijanska *Brain tumor segmentation from MRI data sets using region growing approach* Proceedings of the 7th International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH '11), pp. 185-188, 2011.

[12] S.K. Warfield, M. Kaus, F.A. Jolesz and R. Kikinis *Adaptive, Template Moderated, Spatially Varying Statistical Classification* Medical Image Analysis, Vol 4, No. 1, pp. 43-55, 2000.

[13] M.N. Ahmed, S.M. Yamany, N. Mohamed, A.A. Farag and T. Moriarty *A Modified Fuzzy C-Means Algorithm for Bias Field Estimation and Segmentation of MRI Data* IEEE transactions on medical imaging, Vol. 21, No. 3, 2002.

[14] S. Chen and D. Zhang *Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure* IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics, Vol. 34, No. 4, pp. 1907-1916, 2004.

[15] T.F. Chan and L.A. Vese *Active contours without edges* IEEE Transactions on Image Processing, Vol. 10, No. 2, pp. 266-277, 2001.

[16] J. C. Moreno, V. B. Surya Prasath, H. Proena, and K. Palaniappan *Fast and globally convex multiphase active contours for brain MRI segmentation* Computer Vision and Image Understanding, vol. 125, pp. 237-250, 2014.

[17] L.A. Vese and T.F. Chan *A multiphase level set framework for image segmentation using the Mumford and Shah model* International Journal of Computer Vision, Vol. 50, No. 3, pp. 271-293, 2002.

[18] Y. Masutani, T. Schiemann, and K.H. Hohne *Vascular shape segmentation and structure extraction using a shape-based region-growing model* Medical Image Computing and Computer-Assisted InterventationMICCAI'98: Proceedings of the 1st International Conference Cambridge, 1998.

[19] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzy *Gradient flows and geometric active contour models* Proceedings of the 5th International Conference on Computer Vision (ICCV 95), pp. 810-815, 1995.

[20] J.H. Xue, A. Pizurica, W. Philips, E. Kerre, R. van de Walle, I. Lemahieu *An Integrated Method of Adaptive Enhancement for Unsupervised Segmentation of MRI Brain Images* Pattern Recognition Letters, vol. 24, no. 15, pp. 2549-2560, 2003.

[21] L.R. Schad, S. Blml, I. Zuna *MR tissue characterization of intracranial tumors by means of texture analysis* Magnetic Resonance Imaging, Vol. 11, pp. 889-96, 1993.

[22] H. Xuea, L. Srinivasana, S. Jianga, M. Rutherforda, A.D. Edwardsa, D. Rueckertb, J.V. Hajnal *Automatic segmentation and reconstruction of the cortex from neonatal MRI* NeuroImage, Vol. 38, No. 3, pp. 461-477, 2007.

[23] S. M. Smith *Fast robust automated brain extraction* Human Brain Mapping, Vol. 17, No. 3, pp. 143-155, 2002.

[24] M. Battaglini, S.M. Smith, S. Brogi and N. de Stefano *Enhanced brain extraction improves the accuracy of brain atrophy estimation* NeuroImage, vol. 40, no. 2, pp. 583-589, 2008.

[25] E.B. Lewis and N.C. Fox *Correction of differential intensity inhomogeneity in longitudinal MR images* NeuroImage, Vol. 23, pp. 75-83, 2004.

[26] J.G. Sled, A.P. Zijdenbos and A.C. Evans *A Nonparametric Method for Automatic Correction of Intensity Nonuniformity in MRI Data* IEEE transactions on medical imaging, Vol. 17, No. 1, 1998.

[27] H. Akbari and B. Fei *3D ultrasound image segmentation using wavelet support vector machines* Med Phys, Vol. 39, Is. 6, pp. 2972-84, 2012.

[28] X. Yang and B. Fei *3D Prostate Segmentation of Ultrasound Images Combining Longitudinal Image Registration and Machine Learning* Proceedings of SPIE, 2012.

[29] Y. Zhan and D. Shen *Deformable Segmentation of 3-D Ultrasound Prostate Images Using Statistical Texture Matching Method* IEEE Transactions on Medical Imaging, Vol. 25, No. 3, 2006.

[30] J. Olivier and L. Paulhac *3D Ultrasound Image Segmentation: Interactive Texture-Based Approaches*, Medical Imaging, Dr. Okechukwu Felix Erondu (Ed.), InTech, DOI: 10.5772/35934, 2011. Available from: http://www.intechopen.com/books/medical-imaging/3d-ultrasound-image-segmentation-interactive-texture-based-approaches

[31] J.D. Quartararo *Semi-Automated Segmentation of 3D Medical Ultrasound Images* Master Thesis, Worcester Polytechnic Institute, 2008.

[32] J. Anquez, E.D. Angelini and I. Bloch *Segmentation of Fetal 3D Ultrasound Based on Statistical Prior and Deformable Model* IEEE International Symposium on Biomedical Imaging: from nano to macro, 2008.

[33] T. Georgiou *Unsupervised Clustering of 3D medical images based on textural features*

[34] A. Kassner, R.E. Thornhill *Texture Analysis: A Review of Neurologic MR Imaging Applications* AJNR. American journal of neuroradiology, vol.31(5), pp.809-16 [Peer Reviewed Journal], 2010.

[35] D.C. He and L. Wang. *Texture Unit, Texture Spectrum, And Texture Analysis.* Geoscience and Remote Sensing, IEEE Transactions on, vol. 28, pp. 509 - 512, 1990.

[36] R.M. Haralick, K. Shanmugam, I. Dinstein *Textural features for image classification.* IEEE Transactions on Systems Man and Cybernetics, vol. 3: no. 6 pp. 610 - 21, 1973.

[37] M.M. Galloway *Texture Analysis Using Gray Level Run Lengths* Comput. Graphics Image Process., vol. 4, pp. 172-179, 1975.

[38] B. S. Manjunath, R. Chellappa *Unsupervised texture segmentation using Markov random field models.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, pp. 478-482, 1991.

[39] Z. Haliche K. Hammouche *The gray level aura matrices for textured image segmentation* Analog Integrated Circuits and Signal Processing, vol. 69, no. 1, pp.29-38 [Peer Reviewed Journal], 2011.

[40] M. Dash, H. Liu *Feature selection for classification* Intelligent data analysis, Vol. 1, pp. 131-156, 1997.

[41] J. Li , B.L. Lu and Z.F. Wu *Hierarchical fuzzy filter method for unsupervised feature selection* Journal of Intelligent & Fuzzy Systems, Vol. 18 , pp. 157-169, 2007.

[42] Z. Li, Y. Yang, J. Liu, X. Zhou and H. Lu *Unsupervised Feature Selection Using Nonnegative Spectral Analysis* Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[43] J.G. Dy and C.E. Brodley *Feature Selection for Unsupervised Learning* Journal of Machine Learning Research, Vol. 5, pp. 845-889, 2004.

[44] M. Dash, H. Liu *Feature Selection for Clustering* Lecture Notes in Computer Science, Vol. 1805, pp. 110-121, 2003.

[45] P. Mitra, C.A. Murthy, S.K. Pal *Unsupervised Feature Selection Using Feature Similarity* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 3, pp. 301-312, 2002.

[46] M. Dash, K. Choi, P. Scheuermann and H. Liu *Feature Selection for Clustering - A Filter Solution* IEEE Transactions on Data Mining, pp. 115-122, 2002.

[47] C. Boutsidis, Petros Drineas[a], Michael W. Mahoney[b] *Unsupervised Feature Selection for the k-means Clustering Problem* [a]Department of Computer Science, Rensselaer Polytechnic Institute Troy
[b]Department of Mathematics, Stanford University, Stanford

[48] J. Basak, R.K. De, S.K. Pal *Unsupervised feature selection using a neuro-fuzzy approach* Pattern Recognition Letters, Vol. 19, pp. 997-1006, 1998.

[49] C. Velayutham, K. Thangavel *Unsupervised Quick Reduct Algorithm Using Rough Set Theory* Journal of electronic science and technology, Vol. 9, No. 3, 2011.

[50] A. Khler, M. Ohrnberger, C. Riggelsen, F. Scherbaum *Unsupervised Feature Selection for Pattern Search in Seismic Time Series* JMLR: Workshop and Conference Proceedings 4: 106-121

[51] D. Arthur, S. Vassilvitskii *k-means++: The Advantages of Careful Seeding* Proceeding SODA '07 Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 1027-1035, 2007.

[52] J.C. Bezdek *Pattern Recognition with Fuzzy Objective Function Algorithms* Plenum Press, 1981

[53] L.A. Zadeh *Fuzzy Sets* Information and control, Vol 8, pp. 338-353, 1965.

[54] C. D. Manning, P. Raghavan, H. Schtze *Introduction to Information Retrieval* Cambridge University Press. , pp. 356 - 60, 2009.

[55] L. R. Dice, *Measures of the amount of ecologic association between species* Ecology, Vol. 26, No. 3, pp. 297-302, 1945.

[56] K. Kowkabzadeh *Evaluation of Tissue Segmentation of Brain MR Images* Master Thesis, Chalmers University of Technology, 2010.

[57] O. Tsang, A. Gholipour, N. Kehtarnavaz, K. Gopinath, R. Briggs and I. Panahi *Comparison Of Tissue Segmentation Algorithms In Neuroimage Analysis Software Tools* 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 3924-28, 2008.

[58] *http://www.bic.mni.mcgill.ca/brainweb/*

[59] C.A. Cocosco, V. Kollokian, R.K.-S. Kwan and A.C. Evans *BrainWeb: Online Interface to a 3D MRI Simulated Brain Database* NeuroImage, Proceedings of 3-rd International Conference on Functional Mapping of the Human Brain, Vol. 5, No. 4, 1997.

[60] R.K.-S. Kwan, A.C. Evans and G.B. Pike *MRI simulation-based evaluation of image-processing and classification methods* IEEE Transactions on Medical Imaging, Vol. 18, No. 11, pp. 1085-97, 1999.

[61] R.K.-S. Kwan, A.C. Evans and G.B. Pike *An Extensible MRI Simulator for Post-Processing Evaluation* Visualization in Biomedical Computing (VBC'96). Lecture Notes in Computer Science, vol. 1131, pp. 135-140, 1996.

[62] D.L. Collins, A.P. Zijdenbos, V. Kollokian, J.G. Sled, N.J. Kabani, C.J. Holmes and A.C. Evans *Design and Construction of a Realistic Digital Brain Phantom* IEEE Transactions on Medical Imaging, Vol. 17, No. 3, pp. 463-468, 1998.

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

[64] B.H. Menze, A. Jakab S. Bauer, J.K. Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M.A. Weber, T. Arbel, B.B. Avants, N. Ayache, P. Buendia, D.L. Collins, N. Cordier, J.J. Corso, A. Criminisi, T. Das, H. Delingette, . Demiralp, C.R. Durst, M. Dojat, S. Doyle, J. Festa, F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K.M. Iftekharuddin, R. Jena, N.M. John, E. Konukoglu, D. Lashkari, J.A. Mariz, R. Meier, S. Pereira, D. Precup, S.J. Price, T.R. Raviv, S.M.S. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H.C. Shin, J. Shotton, C.A. Silva, N. Sousa, N.K. Subbanna, G. Szekely, T.J. Taylor, O.M. Thomas, N.J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D.H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes and K. Van Leemput *The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)* IEEE Transactions on Medical Imaging, Vol. 34, Is. 10, pp. 1993-2024, 2015.

[65] A.M. Mendrik, K.L. Vincken, H.J. Kuijf, M. Breeuwer, W.H. Bouvy, J. de Bresser, A. Alansary, M. de Bruijne, A. Carass, A. El-Baz, A. Jogh, R. Katyal, A.R. Khan, F. van der Lijn, Q. Mahmood, R. Mukherjee, A. van Opbroek, S. Paneri, S. Pereira, M. Persson, M. Rajchl, D. Sarikayan, O. Smedby, C.A. Silva, H.A. Vrooman, S. Vyas, C. Wang, L. Zhaon, G.J. Biessels, M.A. Viergever. *MRBrainS Challenge: Online Evaluation Framework for Brain Image Segmentation in 3T MRI Scans.* Computational Intelligence and Neuroscience, special issue on Simulation and Validation in Brain Image Analysis, Article ID 813696, 2015.

[66] M. Earle, G. De Portu and E. DeVos *Preparations of a Homemade Ultrasound Biopsy Phantom* Journal of Clinical Ultrasound, vol 17, p.p. 456-8, 1989.

[67] M.P.McNamara and M.E. McNamara *Agar ultrasound phantoms for low-cost training without refrigeration* African Journal of Emergency Medicine, vol 6, p.p. 18-23, 2016.

[68] R.O. Bude and R.S Adler *An Easily Made, Low-Cost, Tissue-Like Ultrasound Phantom Material* Journal of Clinical Ultrasound, vol 23, p.p. 271-3, 1995.

[69] *http://www.terason.com/*