An Optimized Segmentation Framework Applied to Glioma Delimitation

László LEFKOVITS[1], Szidónia LEFKOVITS[2], Mircea F. VAIDA[3]

[1] Sapientia Univesity,
Corunca 1C, Tg. Mureș, 540485, Romania
lefkolaci@ms.sapientia.ro
[2] "Petru Maior" University, N. Iorga 1, Tg. Mureș, 540088, Romania
szidonia.lefkovits@science.upm.ro
Technical University of Cluj-Napoca, Memorandumului 28, Cluj-Napoca, 400144, Romania
mircea.vaida@com.utcluj.ro

Abstract: In this article we describe our segmentation framework applied to glioma delimitation in multimodal magnetic resonance images. Statistical pattern recognition strategies are applied to create a discriminative function. The discriminative classifier is the result of an automatic learning process based on random forest (RF) algorithm. This algorithm is used for two different purposes as well as in the construction of segmentation classifiers, as in the variable importance evaluation task. In the training phase the most important local image features are selected and the most adequate optimal parameters of the RF classifier are determined. The goal is to find the discriminative model that allows us to obtain the best possible segmentation performances. The segmentation framework obtained was evaluated online using the brain tumor segmentation benchmark system, and the performances were compared to the best ones reported in the literature.

Keywords: image segmentation, features selection, random forest, brain tumor, multi modal MRI.

1. Introduction

Gliomas originating from glial cells are considered some of the most aggressive primary brain tumors [8]. Brain tumors are classified by the World Health Organization into four grades [12]. Tumors with a reduced grade of malignancy, LG (Low-Grade) are called grade I and II tumors, while tumors with an increased grade of malignancy, HG (High-Grade) are grade III and IV tumors. People with high-grade tumors have an average survival rate of less than two years. The slower- growing low-grade variants come with a life expectancy of several years, but their discovery and diagnosis is much more difficult, since in many cases there are no visible symptoms. Usually they are discovered accidentally through a CT or MRI examination. Thus, the increasing number of Magnetic Resonance Scanners can play a vital role in preventive medicine. The combination of various image processing techniques will create, in the near future, an efficient diagnostic tool and will offer a favorable environment in which the information required can be easily discovered and/or extracted from the image content. In medical imaging, brain tumor segmentation can be done manually, semi-automatically and automatically [16]. Manual segmentation is and active tumors, clots and necrotic tissue that vary greatly in size, location, shape and appearance across patients. Moreover, these lesions often show inhomogeneity in their intensity as well as large variations of intensity between subjects, especially if acquired with different scanners or at different imaging centers. Despite the progress made by recent studies, automatic brain tumor segmentation is still a challenging task.

The BRATS Challenges [15] are organized with the main goal of discovering the best methods applied for brain tumor segmentation around the world. The most efficient and recent methods were based on deep learning techniques and random forest classifiers [11]. In deep learning methods the selection of lowlevel features is done automatically, but the choice of learning structure and the large amount of parameters that have to be tuned is more difficult. Conversely, the RF classifier has to be tuned by setting the appropriate domain range (value) of only a few parameters, but it is more difficult to choose the adequate low level features to be used. The selection of

features and RF parameters is based on the intuition and experience of the authors. The exact definition and usage of the features applied remains their secret. They are not clearly described, only vaguely mentioned. Usually, the systems work with a large amount of features having hardly any theoretical or practical study behind their utility.

D. Zikic et al. [18] extracted 2000 attributes from the image intensities and from a generative model. For classification they used 40 decision trees, each having a depth of 20. E. Geremia et al. [5] built a discriminative model based on an ensemble of decision trees that associates a vector of 412 features to each point. M. Goetz et al. [6] created another discriminative model based on an ensemble of Extra-Randomized Trees in which they use 208 attributes; 52 attributes for each of the 4 image modalities. O. Maier et al. [13] created their segmentation model based on the random forest classifier and used a set of features which is in concordance with the discrimination criteria of the human observer. D. Mahapatra's article [14] is the only study that used feature importance of random forest to design a feature selection strategy critical for high segmentation and classification accuracy.

In this paper we created a learning framework which optimized our discriminative model used in brain tumor segmentation of multimodal MR images. The framework has two correlated tasks: the selection of the most important low-level image characteristics and the tuning of the classifier used in segmentation. We built the learning framework around the RF (random forest) classifier, which is used both in feature selection and in segmentation tasks. Therefore we had to tune the RF classifier differently for the two above-mentioned tasks.

The automatic choice of variables used during a training process can improve the performances of the final segmentation considerably. The choice of important variables correlated to the segmentation goal increases predictive segmentation accuracy and reduces the complexity of the final model. The proper choice of parameters in the random forest model can decrease the size of the final classifier, leading to lower complexity and a shorter processing time.

The rest of paper is organized as follows: Section 2 describes the theoretical aspects of random forest classifiers; it continues with the details of our segmentation model (Section 3). Next our results and the performance evaluation of the created model are presented compared to the results of BRATS Challenge (Section 4). Finally we draw some conclusions and propose further future development.

2. Theoretical Aspects

Statistical calculus confirms the fact that the combination of multiple uncorrelated, but strong classifiers in an ensemble usually improves the classification performance of the final aggregated classifier. These two contradictory conditions (strength but uncorrelation) are brilliantly combined in the Random Forest (RF) classifier ensemble proposed by L. Breiman [3]. This ensemble is a large collection of strong classifiers structured in binary decision trees. The uncorrelation between the trees is the main strength of the forest. In the learning process each tree is built up based on two random mechanisms:

- the random creation of the training set separately for each tree;

- the random feature selection used for decision-making in each node of every tree.

First, the training set is randomly sampled with replacement N times. The bootstrap set will have the same size N as the initial training set. It is demonstrated [3] that if the bootstrap set and the training set size are equal, the bootstrap set will contain 63.2% different observations, which represents approximately 2/3 of *N*. The remaining 1/3 are duplicates. This means that approximately 1/3 of the training samples are not included at all in the bootstrap set. These instances form the out-of-bag (*OOB*) set. Thus, for each tree with the above described random sampling mechanism we obtain two different set: the bootstrap set and the *OOB* set.

Secondly, for each node of every tree, a split criterion must be specified. In case of RF ensemble the split function is the maximum information gain computed from the Gini impurity. The maximization of the information gain does not consider all the *M* decision features, but only a randomly selected small part of them. The number of the features considered is denoted by m_{tries} ($m_{tries} << M$). Hence, the binary trees are grown by splitting the instances in each node and considering only a random part of the existing features.

Consequently, each tree is built considering its corresponding bootstrap set. The decisions of the tree-nodes use a limited number of randomly obtained features. After the creation process the performance of each tree is measured on its own *OOB* set. The overall *OOB* error is the classification error obtained by the individual trees on their *OOB* sets averaged over all of the trees in the forest.

L. Breiman [3] shows that the upper bound for the generalization error is given by:

$$GE = \rho\left(\frac{1}{S^2} - 1\right) \quad (1)$$

where: ρ - is the mean value of correlation, S – is the strength of the ensemble. In order to decrease the error, the correlation should be decreased and the strengths increased. An interesting characteristic of RF is the fact that the general error (*GE*) can be estimated through the *OOB* error. This error is important for choosing and tuning RF parameters. The final goal is to minimize the *OOB* error; this means that the *GE* would decrease as well.

The RF classifier contains a large amount of information concerning the relationship between attributes and classes. This information can be used for prediction, clustering and variable importance measurement. The random forest framework can be constructed by considering three different variable importance measures, either the selection frequency or the Gini importance (*GI*) or the permuted importance (*PI*).

In our work we used the permutation importance. *PI* is the increase in misclassification for the *OOB* set after variable *j* has been permuted. Consider the following quantities: w_k - the number of wrong decisions of the *k-th* tree on its own OOB_k set and w_{jk} -the number of wrong decisions of the *k-th* tree on its own OOB_k set by randomly permuting the values of variable *j*. All the other variables remain unchanged. By scrambling the values of variable *j*, the *PI* difference obtained will measure the importance of variable *j* in the OOB_k set. The difference divided by the number of instances in OOB_k is the average permuted importance of variable *j* over all trees from the ensemble:

$$PI_j = \frac{1}{K_{trees}} \sum_{k=1}^{K_{trees}} (w_{jk} - w_k) / |OOB_k| \quad (2)$$

The *PI* is the predictive importance of variables because it is calculated from the *OOB* samples. Variables with no importance will have a very low *PI* value, close to *0*. At the same time, the value can be negative, because the number of errors in the permuted *OOB* can be lower than the error in the non-permuted *OOB*. The relevance of the extent of permuted importance can be increased by applying the random permutation of variables in *OOB* sets several times.

3. The Segmentation Model

The segmentation task is done voxel-wise by a discriminative function [17]. This function is determined during a supervised learning algorithm (Section 3.4.). The main part of the learning process is based on Random Forest classification algorithm. In this article we present the most important parts of the learning framework considered to be significant and bring essential improvement in segmentation performances.We also analyzed in detail and described two further, even more important parts of the learning process in this article: the choice of features to be used and the choice of parameters for the RF classifier. The main components of the complete segmentation model are: the annotated image database, the preprocessing, the learning framework and the post-processing.

The difference between this model and the standard discriminative model is the feature selection step. In this step, the feature selection algorithm consists of the variable importance evaluation for the defined segmentation task.

Using the results of the variable importance we are able to eliminate the unimportant variables. At the same time it allows testing new low-level features that should improve the segmentation performances or be more important than the existing features.

3.1. Database

In our experiments we have used the most recent training dataset, the BRATS 2015 database [1], described in [15]. This version of the database contains 220 high-grade (HG) and 54 low-grade (LG) brain tumor image sets. All image set contains the following four RM images: T1-weighted, T1c:T1-weighted with contrast material Gadolinium, T2-weighted and

FLAIR: T2-weighted FLAIR image. Every brain RM scan was manually annotated by experts in this field. The ground truth provided contains four types of tumor structures: 1-edema, 2-non-enhancing core, 3-necrotic core and 4-enhancing core.

In this paper we describe the experiments and the classification performances obtained for two classes: whole tumor (WT-including all the tumor tissues) and tumor core (TC-not containing the edema). These classes are more significant in medical practice.

3.2. Preprocessing

MRI acquisition is associated with many artifacts. Some of them can be eliminated by medical staff, by setting the acquisition parameters adequately. The images acquired are sufficiently appropriate for human visual analysis, but the main issue is that these artifacts significantly influence automatic segmentation. In our work we have dealt with three important artifacts: inhomogeneity correction, noise filtering and intensity standardization. More details can be found in our previous works [4, 9].

3.3. Feature Extraction

Image processing offers many procedures for the extraction of characteristics from images. In the field of tumor segmentation there are many studies that try to find certain characteristics with a high correlation to the brain tumor appearance in MR images. Despite these research efforts, no proper feature sets have been found yet. That is the reason for using a large feature set, with the features having little correlation to the goal of classification. In our approach we started with defining a large feature set, this is later reduced in order to eliminate the irrelevant or noisy features. For each feature, we defined many low-level characteristics that describe the intensities in the neighborhood of the voxels studied. Thus we have chosen 240 low-level features described in detail in our previous article [11]. By extracting all of these features for every voxel in all modalities, we transform the image segmentation task into a statistical pattern recognition problem.

3.4. The learning framework

The learning framework is the main part of our segmentation model, including several interrelated modules such as feature extraction, feature selection, learning algorithm, classification function and segmentation performance evaluation. These modules use a lot of information stored in non-overlapping databases such as the training set, test set and evaluation test set, each containing other examples. In order to create a well-working discriminative function we delimitated eight sequences described in the processing flowchart (Figure 1) of our proposed framework.

1. The low-level image characteristics are extracted from the set of image databases mentioned. For every image the values of low-level image characteristics are organized and stored in a stack image file. Each layer of the image obtained in the stack contains a feature map corresponding to the image. So the total number of slices (images) in the stack is equal to the number of extracted features. The stacks corresponding to one 3D image are concatenated and form a 4D stack of volumes. The feature extraction functionality is implemented by a number of independent scripts or functions. Independence provides flexibility in the choice of corresponding image characteristics that are integrated in the final classifier.

2. The starting point of a statistical pattern recognition system (supervised learning) is the database. The database (*ADB*-annotated database) contains all information extracted from the annotated image feature stacks. Thus, the *ADB* can be very large. Each instance corresponds to an image voxel, and the attributes are values of the local image characteristics extracted for that voxel. Because the *ADB* is too large to be manageable by any of the most recent learning algorithms in existence, we have to extract a certain important part of it. In practice, it is sufficient to work with only a subsampled part of the database. Each class was randomly subsampled considering a weighted balanced number of entities in each class, leading to the creation of the sampled database (*SADB*). The weight ratio was computed according to the cardinality of each class in an image.

3. Unfortunately the obtained *SADB* is still unmanageable by any existing learning algorithm; we must further reduce its size by reducing the number of applied features. Here, the attributes of the *ADB* are reduced. The dimensionality reduction algorithm [10]

evaluates the variable importance (*VI*) and eliminates part of the unimportant variables.

4. Taking into account the segmentation performances we can eliminate a considerable portion of unimportant variables in each cycle. The dimensionality reduction algorithm was repeated several times until all features considered unimportant by the *VI* measure were reduced from the original *SADB*.

5. In this step the feature set can be further enlarged with new low-level image characteristics, if necessary. This evidently leads to the growth of the *ADB* database.

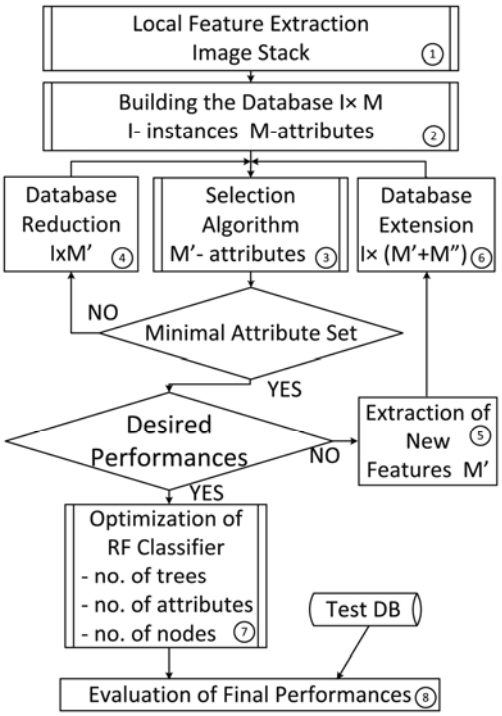

Figure 1. Learning framework

6. The actual feature set is obtained after adding new variables to the existing set. Repeating steps 4, 5 and 6 cyclically, the important variables are kept based on the segmentation performances, and thus unimportant variables can be further reduced from the *SABD*.

7. In the previous step we were able to find the significant variable set for the target segmentation task. Next, we need to tune the RF classifier. By tuning the RF parameters we were able obtain an efficient classifier for the proposed segmentation task. The detailed description and the result obtained by tuning the RF classifier are given below.

8. Finally, the tuned RF ensemble obtained classifies all instances of the test database, which is built from unseen test images (*UTI*). The results obtained are converted in segmented images. The goodness of segmentation is evaluated using special segmentation measures to obtain the average final segmentation performances.

3.4.1. Dimensionality Reduction

Dimensionality reduction in the database can be achieved in two independent ways:

1. Reduction of the number of instances used;
2. Reduction of unimportant variables.

The effect of such reduction can be tracked in the diagram below, which illustrates the dimensionality reduction (Figure 2). Here, the first step is the reduction of instances by random subsampling and by eliminating similar or redundant instances. Redundant instancescan be determined by using the proximity evaluation offered by the RF classifier and computed during the creation of the ensemble.

In the second reduction (Figure 2) we used our dimensionality reduction algorithm, described in detail in a previous article [10]. For this purpose we used the permuted variable importance (*PI*) evaluation obtained during the RF training process. The evaluation is repeated several times on a random sampled part of the database. Because each sample set is only a very small part of the complete database, we proposed a statistical evaluation of the variable importance (*VI*) obtained in each cycle.

The *SADB* contains many instances (*I*) and for each instance we defined a considerable number of features (*F*), thus the size of the *SADB* is very large; it is practically unmanageable by any of the learning algorithms. In order to reduce the large number of features (*F*), we proposed a feature selection algorithm which can handle large databases (Algorithm 1).

Subsequently, we need to distinguish between the training database and the RF-training set (*RFTS*, size $G \times F$, $G \ll I$) which is a part of the *SADB* and is directly involved in training the RF classifier. The principle of the algorithm is to create a different *RFTS* in each cycle using a bagging process. On this *RFTS*,

the RF algorithm evaluates the variable importance (*VI*) and the *OOB* error (step 6). If the *OOB* error is less than 35%, the resulting importance vector is updated by the *VI* values (step 7). The resulting variable importance *RVI* is a sum of *VI* weighted by the OOB_{error} obtained in each cycle (steps 8, 9). The cycles are repeated until one of the stop conditions (step 11) is satisfied: time limit, number of cycles or the condition related to the repetitions (i.e. *number of cycles>2I/G*).

Input: training database $TDB_{I\times F}$
Output: set of reduced variables
1 set the maximum size of RF-training set ($RFTS_{G\times F}$
2 initialize the RVI result variable importance vector;
3 **repeat**
4 create the $RFTS$ by bagging from TDB;
5 train RF on $RFTS$;
6 evaluate VI of every variable;
7 **if** $OOB_{error} < 35\%$ **then**
8 compute VI/OOB_{error};
9 update $RVI := RVI + VI$;
10 **end**
11 **until** *stop condition*;
12 eliminate each variable being in 20% of lowest ranks
13 **return** reduced variable set;

Algorithm 1. Feature selection algorithm

Assuming that the statistical evaluation tends to its real limit, the importance value of significant variables increase, while for noisy or insignificant variables, it remains low. Following the algorithm proposed and resumed above, we could eliminate a considerable portion of unimportant variables.

Furthermore, the elimination of variables depends on the decrease of the segmentation performances. The proportion of reduction is empirical; it depends on the number of attributes used and the performances obtained. In the first step we are able to reduce a large number of attributes, whereas in the last steps, only a small number. This must be in correlation with the attainable performances.

In our framework we defined 240 image characteristics on each voxel of a 3D image. Thus, considering four acquisition modalities, there are (4×240) 960 different characteristics for every voxel in a brain. Because a single brain contains about 1.5 million voxels it requires 10 GB of memory for storage. The memory size necessary to store 100 brain images in the annotated database (*ADB)* is about 1 TB. In a statistical learning algorithm the more data is included, the better its generalization. Consequently, we should include more than 100, but clearly the 100 brain images also produce an unmanageable dataset size. In the first reduction step we could decrease the number of instances used by a factor of 12. Thus, the sampled database *(SADB)* becomes 80 GB instead of 1TB (Figure 2).

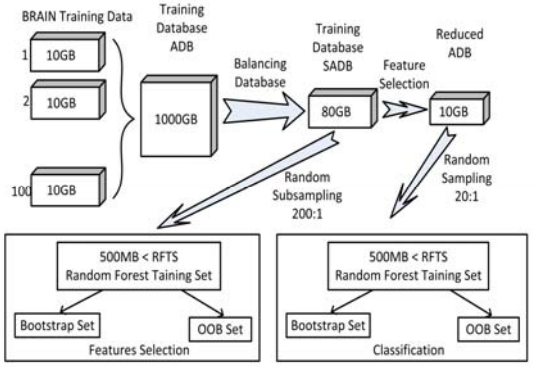

Figure 2. Dimensionality reduction

In the second step of the reduction we used our feature selection algorithm repeatedly. By monitoring the overall evaluation of the *OOB* error we can see a significant increase in a restricted interval at only about $M \in [80,120]$, illustrated in Table 1. and Figure 3. The *OOB* error is a type of estimated error which does not reflect the segmentation performances directly; instead, it represents the goodness of the model. Segmentation performances can be evaluated in several ways, but the Dice coefficient is one of the most accepted measures of segmentation similarity.

$$Dice = 2 \cdot |S_1 \cap S_2| / (|S_1| + |S_2|) \quad (3)$$

| | is cardinality, S_1 the region in the annotated image, S_2 that of the segmented image.

Table 1. The effect of parameter *M* on the classification performances

M - attributes	960	480	240	120	80	60	45	30
OOB error	5.01	5.08	5.1	5.22	5.32	5.51	5.8	6.35

M-attributes	240	120	80	60	45	30
Dice WT-ADB	0.868	0.866	0.848	0.843	0.838	0.828
Dice TC-ADB	0.861	0.861	0.868	0.860	0.849	0.806
Dice WT-UTI	0.805	0.803	0.783	0.780	0.777	0.767
Dice TC-UTI	0.757	0.755	0.757	0.748	0.737	0.685

We evaluated the Dice coefficient of our segmentation obtained for two classes (WT-whole tumor, TC-tumor core) on the whole *ADB* (containing all voxels of 100 brain

image sets) and on *UTI* (unseen test images of 20 brain image sets). The evaluation can be done by including *M=240* features at most, because learning time increases exponentially with the increase of this parameter. Analyzing the segmentation results obtained, we see almost the same behavior on the two image sets; i.e., the Dice coefficient increases significantly from about *M=80* features and remains almost constant for *M>120*. According to this result, a value of *M=120* is an appropriate choice for the number of features. The elimination of unimportant features reduced the database to 10 GB (Figure 2.), which was used to create the classification function for segmentation.

Simultaneously, we emphasize the relationships between the *OOB* error and the Dice coefficient obtained in segmentation for both classes *WT* and *TC*. The Dice index is evaluated on two different databases, *ADB* and *UTI*. The reason for using both databases is to demonstrate the similar behavior in the tuning of RF parameters, so that the test on the *UTI* set can be omitted.

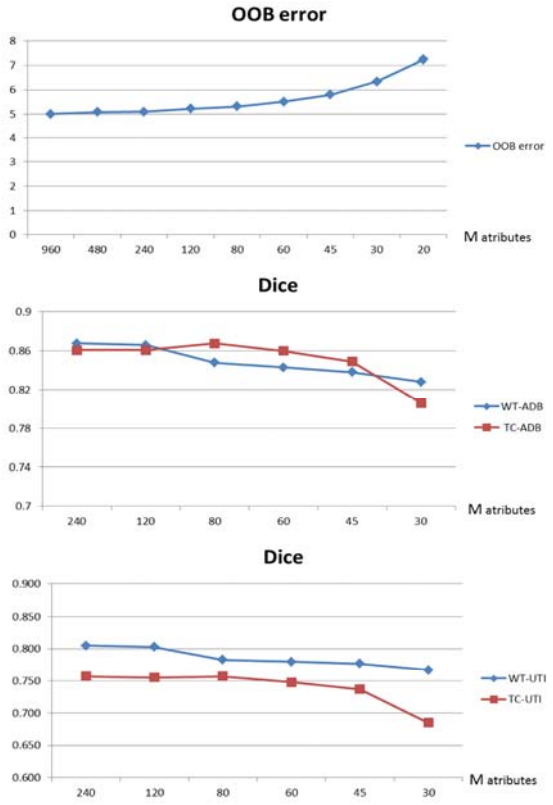

Figure 3. OOB error and Dice coefficient against the parameter *M*

As we will see, the classifier is trained and evaluated several times until we attain the final tuned classifier. The RF parameter-tuning can be done by using only one test set, thus we used only the *ADB* set.

3.4.2. Tuning the RF

The RF has only two main parameters: the number of trees K_{tree} and the number of features m_{tries}, selected randomly in each node. By increasing the value m_{tries}, the correlation between trees increases, and by increasing K_{tree}, the generalization error of the ensemble decreases. If we include enough trees in the forest, the classification overfit is avoided. In practice, unpruned trees can become exceedingly large and deep. In order to avoid classification overfit, we should add a fairly high number of such large trees to the forest. The memory requirements of the RF classifier increase with each tree included, and decision times are also drastically extended. Therefore, we must limit the number of nodes (T_{nodes}) in each tree.

The primary goal of this study is to find the best parameters for the final classifier used in the proposed segmentation task.

In the literature, there are no theoretical suggestions with regard to choosing the main RF parameters:

1. the number of trees K_{tree} in the ensemble,
2. the number of selected features m_{tries} in each node
3. the number of nodes T_{nodes} in each tree.

By considering these three parameters, we were able to tune the RF classifier according to our goal.

1. The first parameter tuned was the number of trees K_{tree}.

It is clear that the generalization error is reduced with the increase in the number of trees K_{tree}. But increasing the number of trees K_{tree} also increases the memory storage required as well as the computation complexity.

The adequate number of trees can be estimated by following the variation of the *OOB* error dependent on the number of trees. In the final model, only the classification accuracy can determine the overall number of trees K_{tree} used

(Table 2 and Figure 4). Fewer trees are required if we consider prediction purposes, because the *OOB* error stabilizes rapidly, but more trees are necessary to refine and stabilize the variable importance [7]. Stronger predictors will lead to faster convergence. The Dice coefficient is not increasing further at *100* trees or above, and thus choosing $K_{tree}=100$ is enough for our segmentation task. Taking into account the computation of variable importance, about $K_{tree}=300$ trees are needed in the ensemble.

2. The second parameter analyzed was the number of selected features m_{tries}. An m_{tries} number of randomly selected variables is chosen in each node of the trees. These variables restrict the search for the optimal split; from the total of M, only m_{tries} are computed.

Table 2. OOB error and Dice coefficient against the parameter K_{tree}

K_{trees}	30	40	50	100	200	300	400	500
OOB err.	6.05	5.81	5.68	5.44	5.34	5.33	5.27	5.28

K_{trees}	50	100	200	300
Dice WT-ADB	0.865	0.866	0.867	0.867
Dice TC-ADB	0.895	0.898	0.898	0.899

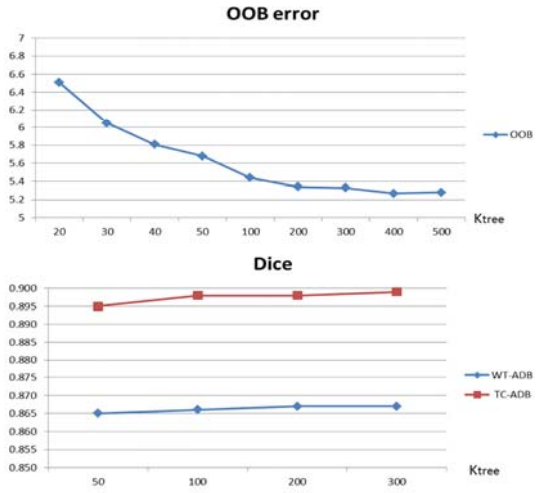

Figure 4. OOB error and Dice coefficient against the parameter K_{tree}

The parameter m_{tries} controls the correlation of trees. Hence, for a small value of m_{tries}, the trees are uncorrelated, and by increasing m_{tries} the correlation increases. As the number of m_{tries} increases, the variance of the randomized variables decreases. At the maximum value of $m_{tries}=M$, which means that in each node, all variables are used to compute the best split. In this case the uncorrelation between trees arises only from the different bootstrap sets used. At the same the RF classifier is transformed into bagging. A large value of m_{tries} considers fewer variables in the tree, resulting in a sparse solution [7]. In addition, the generalized error becomes larger, yet the variable importance measure will be more reliable. If the value of m_{tries} is reduced, the chance of selecting the important variables is higher as well; this will add additional noise to the tree. Therefore, this leads to an increase of the variable variance, and the correlation between trees will also be lower. The quantity of the *OOB* error and the variable importance are not particularly sensitive to this parameter.

However, a default m_{tries} value cannot be determined; it is data-dependent. In every distinct application based on RF, a course search is recommended. In Table 3 we can see that the *OOB* error reaches its minimum at about $2\sqrt{M}$, and further increasing its value is useless. ($m_{tries}=25$ for $M=120$). More interesting results can be obtained by analyzing the Dice coefficient versus m_{tries} (Table 3. and Figure 5). This coefficient does not change significantly in the domain analyzed. A good choice can be at about $\sqrt{M}$; such a low value ensures a strong uncorrelation between the trees of the ensemble.

3. The third parameter for tuning the RF is the size of the tree, i.e. the number of nodes T_{nodes}. The trees of the forest are not pruned at a given level, but their growth can be limited by specifying the number of splits (n_{split}) or the number of nodes (T_{nodes}). Theoretically, there is no need to limit the size of the tree because the bagging process already reduces the variance and also avoids over fitting. If the trees grow until no more splits can be performed, then this leads to very large trees. The processing time of data during the training and testing phases increases drastically in this case. Meanwhile, their memory requirements can also become very large. To avoid these disadvantages, we must limit the number of nodes (T_{nodes}) in every tree. Limiting tree size induces additional diversity in the RF and thus creates a smaller, but more efficient classifier.

In order to choose an optimal value for T_{nodes} we analyzed the progress of the *OOB* error according to Table 4 and Figure 6. We obtained the expected theoretical result, i.e., that the *OOB* error decreases with the increase in tree size and reaches its minimum for the unpruned trees.

At the same time, the Dice coefficients decrease, but their variation can be considered small and negligible with regard to the segmentation performances.

Table 3. The influence of m_{tries} on the classification performances

m_{tires}	7	9	11	15	19	25	31	43
OOB err.	5.54	5.44	5.39	5.34	5.27	5.25	5.22	5.22

m_{tires}	9	11	15	19	25
Dice WT-ADB	0.850	0.850	0.852	0.849	0.846
Dice TC-ADB	0.869	0.872	0.864	0.871	0.867

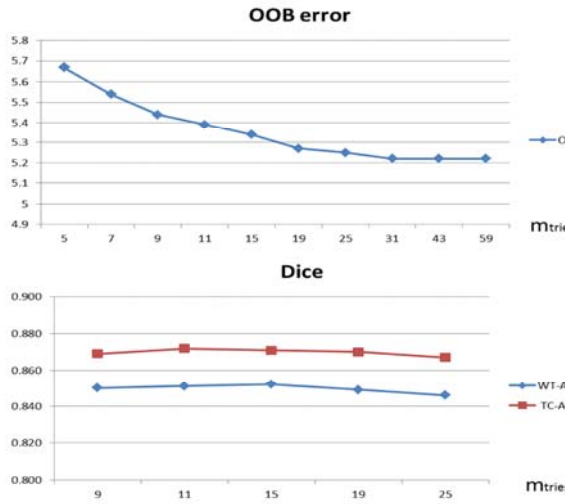

Figure 5. OOB error and Dice coefficient against the parameter m_{tries}

Thus, we can consider T_{nodes} at *2048* an optimal choice in our application; this means approximately *1/4* of the nodes in the unpruned tree.

3.5. Post-processing

In the last phase, post-processing can correct the false detection rate of the classifier obtained in the testing phase. As a consequence of voxel-wise segmentation, some noise is also detected and considered to be part of the tumor. These regions are single isolated voxels or small voxel zones far away from the main tumor region detected. These small, standalone and unconnected volumes are clearly the effect of false detection and can be removed.

Table 4. The influence of tree-size T_{nodes} on the classification performances

T_{nodes}	64	128	256	512	1024	2048	4096	MAX
OOBerr.	12.46	11.04	9.76	8.71	7.68	6.86	6	5.22

T_{nodes}	512	1024	2048	4096	MAX
Dice WT-ADB	0.817	0.830	0.841	0.853	0.867
Dice TC-ADB	0.798	0.823	0.838	0.850	0.899

The elimination of these parts leads to the reduction in false detections. The post-processing step applied assumes that the tumor is a connected region in each brain. This nature is found in the BRATS annotated database used in our experiments.

4. Results and Evaluation

The main results are presented in Section 3. This section only describes the detection performances of our optimized tumor segmentation framework. The training database (ADB) was created from a set of 100 3D brain images, each containing 4 images corresponding to the acquisition modalities and the fifth image was the expert annotated image used in our supervised learning model.

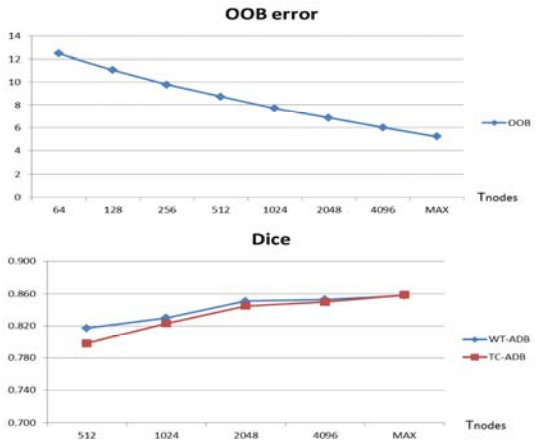

Figure 6. OOB error and Dice coefficient against the T_{nodes} parameter

For each image in the 100 sets, we extracted 120 image features. We worked with an ADB training database of more than 150 million instances. From this we obtained the sampled database STDB with about 10 million instances.

The Random Forest classifier was trained on this database by using the previously determined parameters: K_{trees}=120, m_{tries} = 9 and T_{nodes}=2048. The segmentation results obtained by this system were evaluated with the SICAS online evaluation system, specifically implemented to compare brain segmentation frameworks. [1]. The online segmentation performances (Table 5) are compared with the performances reported on BRATS 2012 and BRATS 2013 Great Challenges [15].

Table 5. Compared Dice indexes

HG	**Our segm.**	**Brats 12[15]**	**Brats 13 [15]**
WT	75-86[%]	63-78[%]	71-87[%]
TC	71-82 [%]	24-37[%]	66-78[%]

5. Conclusion and Future Work

In this paper we described a learning framework developed for brain tumor segmentation in multimodal MR images. As future work we propose further improvements which may lead to a considerable performance increase in segmentation.

It is necessary to improve the learning framework by completing the training with all 272 annotated image sets from BRATS 2015 [1]. Secondly, we are developing a hierarchical segmentation system which could better delimitate the surfaces between different tissues. With additional steps in preprocessing and post-processing, it can also increase segmentation performances. In the future we would like to test and integrate supplementary low-level image features that may be more relevant for brain tumor segmentation.

Another important aspect is to determine the tumor structure and to forecast its future behavior. Our ultimate goal would be to create a segmentation system which could be used in current medical diagnosis in the near future.

Acknowledgements

This work was supported by a grant of Sapientia Foundation – Institute for Scientific Research (KPI), P.N. 13/19/17.05.2017.

REFERENCES

1. SICAS www.smir.ch/BRATS/Start2
2. Trainable segmentation http://imagej.net/ Scripting_the_Trainable_Segmentation
3. Breiman, L. (2001) Random forests. *Machine learning,* 45(1), 5-32.
4. Chiorean, L. D., Suta, L. & Vaida, M-F. (2010) Medical Fusion Components for a Web Dedicated Application. *Studies in Informatics and Control,* 19(4), 435-444.
5. Geremia, E., Menze, B. H. & Ayache, N. (2012) Spatial decision forests for glioma segmentation in multi-channel MR images. *MICCAI-BRATS*.
6. Goetz, M, Weber, C. et. al (2014) Extremely randomized trees based brain tumor segmentation. *MICCAI-BRATS*.
7. Goldstein, B. A., Polley, E.C.et. al. (2011) Random Forests for Genetic Association Studies. *Statistical Applications in Genetics and Molecular Biology:* 10(1).
8. Holland, E. C. (2001) Progenitor cells and glioma formation. *Current opinion in neurology.* 14(3), 683–688.
9. Lefkovits, L., Lefkovits, Sz. & Vaida, M.-F. (2016) Brain Tumor Segmentation Based on Random Forest *Memoirs of the Scientific Sections of the Romanian Academy,* 39*,* 83-93.
10. Lefkovits, L., Lefkovits, Sz. & Emerich, S., Vaida, M-F. (2017) Random forest feature selection approach for image segmentation. *SPIE 10341, 9th Int. Conf. on Machine Vision*, 1034117-1034117.
11. Lefkovits L., Lefkovits Sz. & Szilágyi L. (2017). Brain Tumor Segmentation with Optimized Random Forest. *Brainles 2016 LNCS 10154 ISBN: 978-3-319-55524-9*.
12. Louis, D. N., Ohgaki, H. et al. (2007) The 2007 who classification of tumours of the central nervous system. *Acta Neuropathologica*, 114(2), 97-109
13. Maier, O., Wilms, M. & Handels, H. (2016) Image Features for Brain Lesion Segmentation Using Random Forests. *Brainles 2015 LNCS Vol. 9556*, 119 - 130.
14. Mahapatra, D. (2014). Analyzing training information from random forests for improved image segmentation.*IEEE Trans Image Processing*, 23(4), 1504-1512.
15. Menze, B. H., Jakab, A. et al. (2015) The Multimodal Brain Tumor Image Segmentation Benchmark *IEEE Trans. on Medical Imaging,* 34(10), 1993-2024.
16. Pham, D. L., Xu, C. & Prince, J. L. (2000) Current methods in medical image segmentation. *Annual review of biomedical engineering*, 2(1), 315–337.
17. Reza, S., & Iftekharuddin, K. M. (2013) Multi-class Abnormal Brain Tissue Segmentation Using Texture Features. *MICCAI-BRATS.*
18. Zikic, D., Glocker, B. et. al. (2012) Context-sensitive classification forests for segmentation of brain tumor tissues. *MICCAI-BRATS.*

