Automatic Brain Tissue Segmentation in MRI using Supervised Machine Learning

Michael Rebsamen*, Jan Riedo*, Michael Mueller*

***Abstract*—Magnetic resonance imaging (MRI) is a widespread imaging modality for the brain. Information in MR images can provide valuable markers for the diagnosis of neurodegenerative diseases or the monitoring of tumors. Many analytical methods are based on a labour-intensive volumetric segmentation of the tissue types. An automatic processing of MR images is therefore of high interest in clinical routines. We assess four supervised machine learning algorithms for the automatic segmentation of brain tissue in MR images: decision forest (DF), k-nearest neighbors (kNN), support vector machine (SVM), and stochastic gradient descent (SGD). The results yield similar accuracies but different runtime and scaling behaviours. An ensemble with the predictions of all algorithms outperforms the individual algorithms.**

***Index Terms*—Brain tissue segmentation, MRI, machine learning, DF, kNN, SVM, SGD**

I. Introduction

Segmentation of brain tissues from magnetic resonance images has many clinical applications. Clinicians gain useful information from a separation of tissue into its three main anatomical types: white matter (WM), grey matter (GM), and ventricles (VT). Voxel-based morphometric measures have been used to investigate brain disorders like Alzheimers disease [1], Parkinson's disease [2], or temporal lobe epilepsy [3]. However, manual segmentation of MR images is a labour-intensive task requiring expert skills. Fully automatic approaches for brain tissue segmentation are therefore a topic of active research. A good algorithm classifies the tissue types with high accuracy across a variety of images from different patients. Such a classification is a typical task for machine learning. These algorithms tend to perform well given enough training data during the learning phase. The availability of ground-truth data in sufficient quantity and quality for supervised learning is a particular challenge when working with medical images due to privacy concerns and the costs for manual segmentation. Optimisation of the learning phase with a limited number of training data is therefore required.

Current research is evaluating a variety of discriminative machine learning algorithms on the classification task of brain segmentation. Most proposed methods use ensembles such as decision forests (DFs), with several individual machine learning algorithms combined or sequentially applied. A classic DF implementation was performed by Yaqub et al. [4], achieving scores similar to current benchmarks in brain segmentation. Many extensions to DF with probabilistic models have been successfully tested, such as conditional random fields [5]. Moreover, it was shown that even the simple, instance-based k-nearest neighbors (kNN) algorithm yields accurate results [6]–[8] for brain tissue classification. Furthermore, linear classifiers under convex loss functions, such as linear support vector machines (SVMs) have been successfully applied for automatic brain tumor segmentation [9]. Another approach of this type is the stochastic gradient descent (SGD), which is well established in the field of optimisation problem solvers and a standard for training artificial neural networks [10]. It has gained importance in the field of large-scale machine learning problems [11]. There is no record so far of an application of pure SGD on brain segmentation. Current development tends to bring more accurate results by incorporating 3D neighborhood information [12], [13], prior information from atlases [14], [15], deformable models [16], or combinations thereof [17].

In this experiment, we compare the performance of four well-known machine learning algorithms for the segmentation of brain tissue in MRI data. The chosen metrics include accuracy, computational efficiency, and the amount of training data required. We use the existing medical image analysis pipeline to assess four supervised learning algorithms: decision forest (DF), k-nearest neighbors (kNN), support vector machine (SVM), and stochastic gradient descent (SGD). The algorithms were trained using seven features extracted from a set of 70 MR images and the prediction of the segmentation was evaluated on a different set of 30 MR images. All algorithms are able to classify the brain tissue types, although with varying degrees of accuracy and runtime behaviours. Unsurprisingly, the highest dice coefficient is achieved by combining the predictions (probabilities) of all four algorithms to an ensemble and by using all features. An analysis of feature importance reveals that the various features have a different influence on the algorithms.

II. Methods

A. Dataset

All experiments were conducted on a subset of 100 unrelated, healthy subjects from a dataset provided by the *Human Connectome Project* [18]. From each individual, a total of eight 3-tesla head MR images are available: T1 and T2-weighted image volumes not skull-stripped (but defaced for anonymization) and skull-stripped with a bias field correction, and both modalities once in native T1 space and once in MNI-atlas space [19]. The image sizes are $182 \times 182 \times 217$ voxels with a spatial resolution of 1 mm.

*All authors contributed equally. Biomedical Engineering, University of Bern. Authors e-mail: michael.rebsamen@students.unibe.ch, michael.mueller4@students.unibe.ch, jan.riedo@students.unibe.ch

Ground-truth labels are automatically generated using FreeSurfer [20], assigning each voxel either to background (BG), WM, GM, or VT. The dataset was split in a training set with 70 images and a test set with 30 images.

B. Pipeline

Training and testing data are loaded sequentially, each put through the pipeline consisting of: registration, pre-processing, feature extraction and training/classification. For testing, two additional steps, namely post-processing and evaluation are added.

The data was registered to an atlas space with a multi-modal rigid transformation, using a regular step gradient descent optimizer to establish a common coordinate system. Skull stripping and bias field correction are applied in order to have images of the brain only, with less influence of the MRI scanning characteristics. Furthermore, the preprocessing module applies a gradient anisotropic diffusion filter and z-score normalization to ensure comparable grey scale ranges among all images in the subsequent steps.

Preprocessed data is then fed into the feature extraction module, where seven features are computed. The feature matrix consists of three coordinate features corresponding to the position of the voxel in the atlas space, and an intensity and a gradient on both the T1 and T2 modalities. During feature extraction, a random mask is applied on the training data in order to randomly select a subset of the voxels available. The mask is adjustable individually for BG, WM, GM, and VT. During the learning phase, the selected algorithm is trained to classify voxels based on the given features. In the segmentation phase, the trained algorithm is used to classify all voxels in previously unseen test images. The classified testing data is then forwarded to a post-processing module where a dense conditional random field [21] aims to reduce noise by eliminating small isolated regions. Finally, the evaluation phase assesses the performance of the segmentation by comparing the result to ground-truth and calculating a dice coefficient (see Sec. II-G) for each class.

The medical image analysis pipeline is implemented in Python using TensorFlow [22], scikit-learn [23] and ITK [24]. The tests were performed on a Linux cluster with 4 CPUs and 36 GB memory assigned to the jobs.

C. Decision Forest (DF)

The decision forest is an algorithm for supervised regression and classification. A DF is an ensemble of many decision trees. A decision tree starts at the tree root and splits the data based on the feature which results in the largest information gain (IG). The splitting procedure is repeated at each child node down to the terminal nodes. The terminal nodes at the bottom are the resulting predictors. The optimisation of the information gain is described as:

$$
\begin{aligned}
IG &= H(S) - \sum_{i \in L,R} \frac{|S^i|}{|S|} H(S^i) \\
H(S) &= \sum_{y \in Y} p(y)\ log\ p(y)
\end{aligned}
\tag{1}
$$

where the information gain is the difference of the entropy $H(S)$ before and after the split point. Entropy measures impurity in a node. Decision tree is all about finding the highest possible information gain at each node. Connecting these nodes results in the most homogeneous branch, which has the lowest entropy.

The depth of the trees can be regulated by the chosen number of nodes. A high number of trees improves the accuracy, but increases the computational costs due the learning of the additional trees. Bias of the trees can be reduced by choosing a higher number of nodes.

The hyperparameters were determined by a grid search on the train data, individually for WM, GM and VT. Visualised in Fig. (1) are the results of the grid search for different number of nodes and trees. Therefore the chosen result of 160 trees and 3000 nodes is a trade off for the three tissue types. The result is marked with a red cross in Fig. (1) for each tissue type.

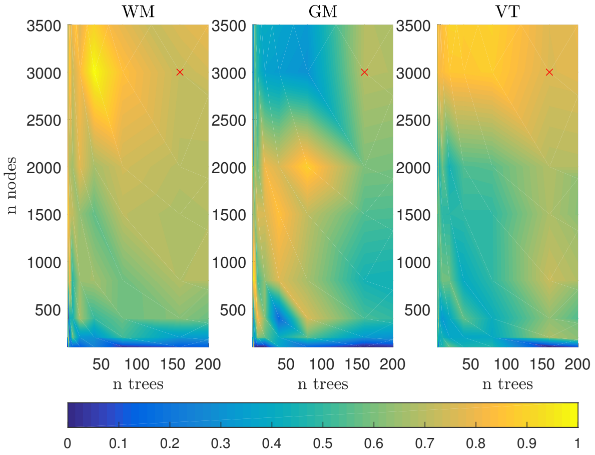


Fig. 1. Optimisation of hyperparameters for DF with a grid search for WM, GM and VT. The red cross marks the chosen *number of trees* = 160 and *maximum nodes per tree* = 3000. Colour represents a scaled dice, individually for each subplot.

D. k-Nearest Neighbors (kNN)

The k-nearest neighbors algorithm does not construct a general model or learn any weights, but actually stores instances of the training data X and labels y. A new data point x is classified by a majority vote of its k nearest neighbors. In case of $k = 1$, this is just the label of the nearest point:

$$
\begin{gathered}
\hat{y} = y_i \\
\text{where } i = \arg\min ||X_{i,:} - x||_2^2
\end{gathered}
\tag{2}
$$

Neighbors are weighted by a function which is inverse proportional to its distance to give closer points more weight. The characteristics of kNN lead to a relatively low training time, and a rather high computation time for new points, depending on the amount of neighbors defined to vote. A weakness of kNN is that is treats all features equally and cannot learn that one feature is more discriminative than another.

A hyperparameter search was conducted for the k-value. The higher k, the better the overall dice with the drawback of a higher computation time. A value of $k = 20$ was found to be fast with only little trade-off in computation time.

E. Support Vector Machine (SVM)

Classification using support vector machines tries to find a hyperplane, separating the data in a high-dimensional feature space. Given the feature vector x_i and the binary label y_i, the SVM solves the following optimisation problem during training:

$$\begin{aligned} \min_{w,b,\xi} \; & \frac{1}{2} w^T w + C \sum_{i=1}^{m} \xi_i \\ \text{s.t. } & y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \; i = 1, \ldots, m \\ & \xi_i \geq 0, \; i = 1, \ldots, m \end{aligned} \tag{3}$$

where $w \in \mathbb{R}^n$ is the normal vector and $b \in \mathbb{R}$ the offset of the separating hyperplane and $\phi(x_i)$ maps x_i into a higher-dimensional space.

The SVM implementation is based on libSVM [25]. Multi-class classification is solved with a *one-against-one* approach. To output probabilities, the predictions are calibrated using *Platt* scaling in which multiclass problems require an additional cross-validation which is an expensive operation for large datasets.

Given the relative low number of available features, we have chosen a radial basis function (RBF) kernel. A regularization term C and a model capacity γ needs to be chosen. These hyperparameters were determined with an exhaustive search and cross-validated on a subset of the training data, yielding $C = 500$ and $\gamma = 0.00005$.

F. Stochastic Gradient Descent (SGD)

Stochastic gradient descent is widely used in machine learning for solving optimisation problems iteratively. SGD has proven to be efficient for large-scale linear predictions [26].

In current context, SGD learns a linear scoring function $f(x) = w^T x + b$ with model parameters w and b by minimizing the training error

$$\arg\min_{w,b} \frac{1}{m} \sum_{i=1}^{m} L(y_i, f(x_i)) + \alpha R(w) \tag{4}$$

where L is a loss function that measures miss-classifications, R is a regularization term penalising model complexity, and α is a non-negative hyperparameter.

In each iteration, a sample is randomly chosen from the training set and the model parameters are updated accordingly:

$$w \leftarrow w - \eta \left(\alpha \frac{\partial R(w)}{\partial w} + \frac{\partial L(w^T x_i + b, y_i)}{\partial w} \right), \tag{5}$$

where η is the learning rate controlling the step size.

We use a smoothed hinge loss (*modified_huber*) for the loss function L, a l_2 penalty ($||w||_2$) for the regularization term R, and a gradually decaying learning rate. This makes SGD similar to a linear SVM. Again, the hyperparameters $\eta = 0.5$ and $\alpha = 0.01$ were determined with an exhaustive search and cross-validated on a subset of the training data.

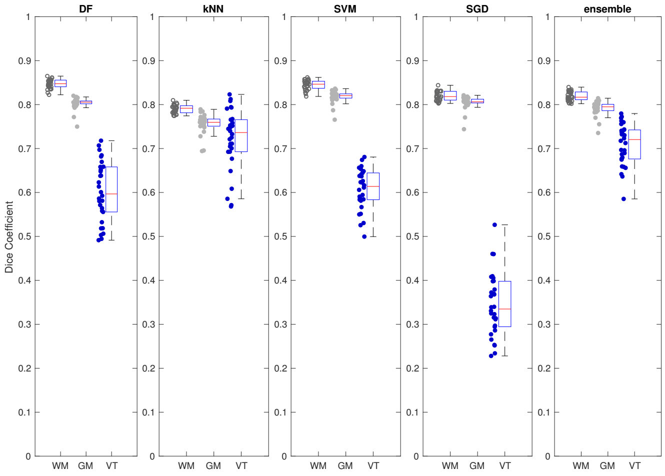


Fig. 2. Distribution of dice coefficients of the 30 test images for WM, GM, and VT. All algorithms were trained with optimal hyperparameters on the full training set of 70 images.

G. Performance Evaluation

The dice coefficient is a commonly used metric to compare the spatial overlap, ranging from 0 (no overlap) to 1 (perfect overlap). To evaluate the accuracy of the segmentation, a dice coefficient is calculated between the prediction (E) and ground-truth (G) for each of the three labels.

$$D = \frac{2|E \bigcap G|}{|E| + |G|} = \frac{2TP}{2TP + FP + FN} \tag{6}$$

III. Results

A. Segmentation Accuracy

All four optimized and examined algorithms are able to segment the MR images into the three tissue types. The performance measured by the dice coefficient for various configurations can be found in Tab. (I). Boxplots with the statistical distributions for each algorithm on the whole test set and with all seven features can be seen in Fig. (2).

The highest dice coefficients for white and gray matter are reached by SVM with 0.84 ± 0.01 (mean ± SD) for WM and 0.82 ± 0.01 for GM but at the cost of a lower value of 0.61 ± 0.05 for VT. The highest mean value for VT is reached by kNN with 0.72 but also with the largest standard deviation of ± 0.07 and lower values for WM (0.79 ± 0.01) and GM (0.76 ± 0.02). Although SGD is competitive on WM (0.82 ± 0.01) and GM (0.80 ± 0.01), the performance on VT is the lowest (0.34 ± 0.07).

The coordinate features seem to have a negative effect on the segmentation of WM and GM with kNN as this is the only algorithm reaching significant higher scores on those tissue types without using the coordinates, but with a negative effect on the ventricles.

The best dice coefficients are reached by combining the predictions from all algorithms to an ensemble.

TABLE I
PERFORMANCE COMPARISON

Features	Size Dataset	DF	kNN	SVM	SGD	ensemble
All (f1-f7)	3	0.85/0.81/0.62	0.70/0.57/0.50	0.83/0.80/0.61	0.82/0.80/0.35	-
	12	0.85/0.81/0.59	0.75/0.66/0.67	0.84/0.81/0.61	0.82/0.80/0.34	-
	70	0.85/0.80/0.60	0.79/0.76/0.72	0.84/0.82/0.61	0.82/0.80/0.34	0.82/0.79/0.71
Coordinates only (f1-f3)	3	0.67/0.63/0.22	0.70/0.55/0.41	0.59/0.52/0.0	0.17/0.23/0.00	-
	12	0.67/0.64/0.11	0.74/0.63/0.56	0.59/0.57/0.0	0.19/0.22/0.00	-
	70	0.67/0.64/0.16	0.77/0.71/0.62	0.60/0.58/0.31	0.17/0.21/0.00	-
All non-coordinates (f4-f7)	3	0.84/0.80/0.50	0.85/0.80/0.45	0.84/0.79/0.0	0.82/0.80/0.34	-
	12	0.85/0.80/0.49	0.85/0.81/0.45	0.85/0.80/0.45	0.82/0.80/0.33	-
	70	0.85/0.80/0.48	0.85/0.81/0.54	0.85/0.80/0.44	0.82/0.80/0.34	-

Overview of achieved accuracy for the different algorithms. Mean dice coefficients for white matter/grey matter/ventricles.
f1-f3: Coordinate features, f4: T1 intensity, f5: T1 gradient, f6: T2 intensity, f7: T2 gradient.

B. Training and Testing Time

A comparison of the computation time both for training and testing is shown in Fig. (3), performed with the infrastructure specified at the end of Sec. II-B. Using the whole training set of 70 MR images, kNN is the fastest to train with $215s$, whereas SGD is the fastest on the prediction with only $42s$ per sample. The training time is growing approximately linear with the amount of training data for kNN and SVM. For DF and SGD, the time required for training is growing significant slower than the amount of training data, leading to a good scaling behaviour. For DF, kNN and SGD, the testing time is independent of the amount of training data used. Only for SVM the testing time is also growing with the size of the training data.

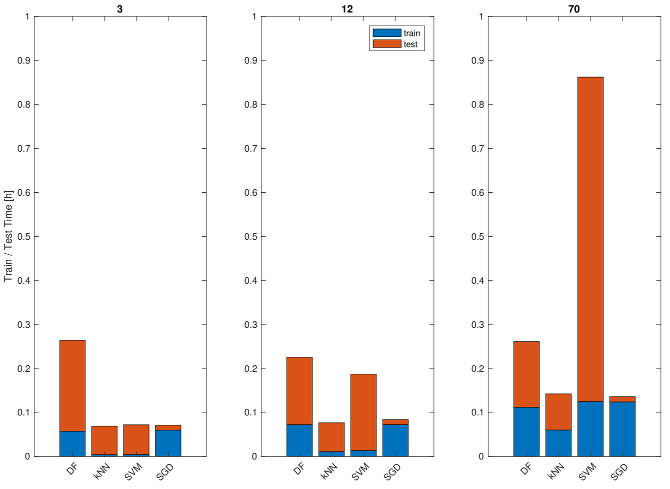


Fig. 3. Time for training and testing of the algorithms with training set sizes of 3, 12 and 70 samples and using all seven features. Test time is for one sample only and includes pre-processing, prediction and post-processing.

C. Feature Inspection

Figure (4) shows the scatter matrix of all features. The whole dataset of 100 images was used for this evaluation. On the diagonal are the histograms to visualise the data distribution of each feature. The first three histograms represent the coordinate features and are normally distributed. The fourth and fifth histogram belong to the intensity features and follow a non-normal distribution. The last two histograms show the gradient features and are partially normal distributed. The right upper part of the diagonal visualises the linear correlation between each of the feature with the associated correlation coefficient. The left bottom part of the diagonal is redundant to the upper part.

The coordinate features with correlation coefficients between -0.11 and 0.05 are independent and do not correlate with any other feature at all. The T1 intensity feature has a moderate linear relationship with the T1 gradient feature and a weak with the T2 gradient feature. The T1 gradient feature has also a moderate correlation with both of the T2 features. The T2 intensity and gradient feature correlate with an correlation coefficient of 0.83 very strong among themselves.

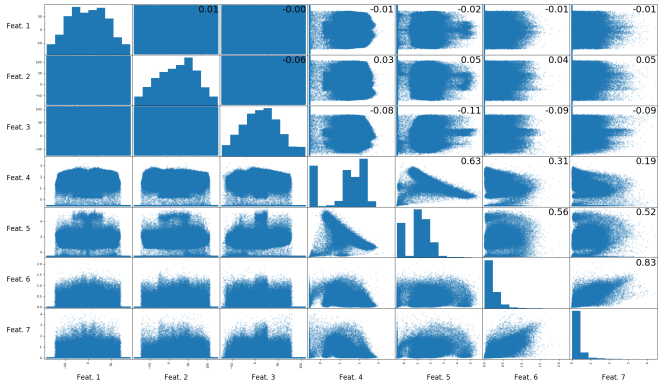


Fig. 4. Scatter matrix plot for the seven features on the full dataset of 100 samples. Each feature is compared against each other. The diagonal shows the data distribution of every feature. The correlation coefficients between two features are shown in the right upper part. f1-f3: Coordinate features, f4: T1 intensity, f5: T1 gradient, f6: T2 intensity, f7: T2 gradient.

The following Fig. (5) visualises the influence of each feature type and the used algorithms. The feature types are divided in three coordinate, two intensity and two gradient features. The first column in this figure is calculated with all the features combined and is considered as a reference. Each column belongs to a single feature type in which the dice of the four used algorithms is visualised. Vertically aligned are dices for WM, GM and VT displayed.

All four algorithms achieve similar results on the intensity features. The GM and WM are equal or even higher compared to the result of all the features combined. The dice for VT with the gradient features is between 0.0 and 0.05. With the coordinate features only, kNN reaches dice coefficients of 0.65 to 0.77 for all tissue types. Whereas the scores of SGD are as low as 0.22 and below with the coordinate features.

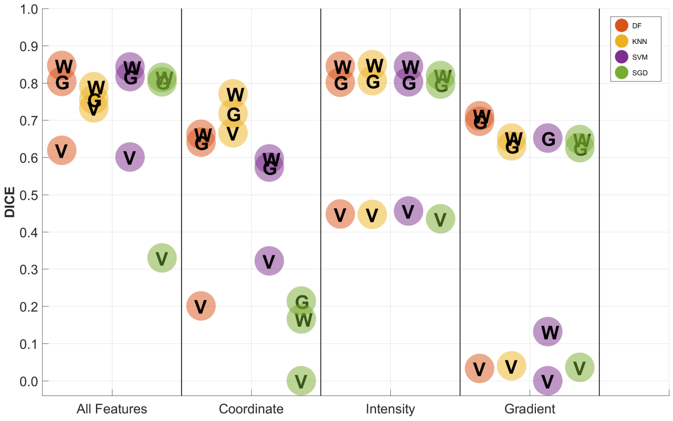


Fig. 5. Comparison of the importance of the individual features for different algorithms. Each algorithm was trained on the full training set of 70 samples and tested on the 30 samples of the test set. The dice score of each algorithm is calculated separately for the three tissue types. W: white matter, G: grey matter, V: ventricles.

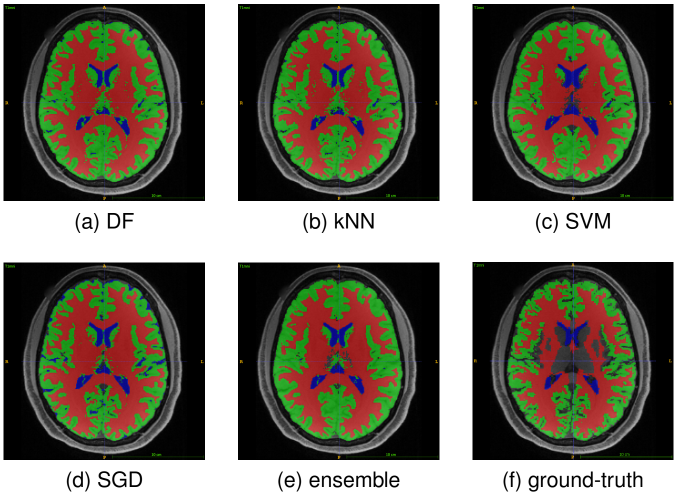
D. Ground-Truth Validity

A sample of a ground-truth segmentation can be seen in Fig. (6f). By comparing this image with the other images in Fig. (6) there is obviously some misclassification in the ground-truth. Especially the center of the brain is mainly classified as background, which is undeniably wrong.

Furthermore, the region between GM and the scull is labelled as background in the ground-truth images. As we only classify WM, GM and VT this might be correct, but anatomically, there is cerebrospinal fluid (CSF) on the boarder like in the ventricles. For algorithms less dependent on the coordinates such as SGD and kNN, this leads to VT classifications outside GM (see Figs. 6b, 6d), which is wrong in terms of ventricles but correct for CSF.

E. Random Mask Optimisation

Tuning the random mask (see Sec. II-B) turned out to be crucial to improve the classification of the ventricles. A first approach to take approximately the same absolute numbers from all voxel types turned out to be suboptimal. Better results were achieved by including roughly the same proportion of voxels from WM, GM and VT. From the background class however, which has the highest absolute number of voxels, approximately the same number of voxels as from the other classes were taken. The fraction of voxels taken into account are therefore 0.004 for WM and VT, 0.003 for GM, and 0.0003 for background. Except for SGD where a better segmentation is achieved with 0.04 for VT which results in approximately the same amount of voxels from all types to be used for training.


(a) DF

(b) kNN

(c) SVM

(d) SGD

(e) ensemble

(f) ground-truth

Fig. 6. Segmentation of optimally tuned algorithms, ensemble, and ground-truth. Size Trainingset: 70 images. All algorithms trained with all features except for kNN, which is trained only on non-coordinate features. All images are from the same head and show the same slice.

IV. Discussion

Our experiments confirm that DF is a good default choice for the segmentation of brain tissue in MR imaging data. All four algorithms reach a similar accuracy but with different runtime behaviours. DF and SGD allow incremental training, in which input data can be processed sequentially in batches of arbitrary sizes. On the other hand, kNN and SVM require all training data to be held in memory, which inherently limits their application. Large amounts of data are trained most efficiently with SGD, few data with SVM. This observation is consistent with the mathematical foundation of these two algorithms. The good scaling behaviour of stochastic gradient descent is based on the approximation of a gradient by randomly (stochastically) choosing samples from a large dataset and not necessarily having to consult all data points in each iteration. A support vector machine might find a solution with a limited number of samples by mapping low-dimensional features to a higher-dimensional feature space, in which the data is more likely to be separable. However, this comes with high computational costs for fitting such a complex model to a larger amount of data. The required tuning of the hyperparameters is still mainly an empirical and experience driven task.

From Table (I) we observe that the non-coordinate features seem to be a better discriminator for WM and GM, since using only the intensity and gradient features consistently leads to high dice coefficients for these classes. The lower score of kNN for WM and GM using all features is most likely due to its inherent inability to give the coordinate features less weight.

We have observed a rather small influence of the size of the training set, DF, SVM, and SGD reaching a similar dice coefficient with both, 3 and 70 training samples. This might be due to the limited number of features used. With additional features, the amount of training data might become more important.

The segmentation of the ventricles yielded the lowest dice coefficients across all algorithms. We attribute this partially to the ground-truth data, which is of equivocal quality, especially in the regions of the ventricles. A manual augmentation of the data by experts is required to judge how much is indeed related to this.

The dataset includes only MR images from healthy individuals. How well the segmentation generalizes for anatomical disturbances, tumors or brain diseases remains to be tested.

Finally, we have only combined the algorithms to a simple ensemble by taking the max. probability for each voxel from the predictions. Advanced combinations like sequentially applying two methods or use one method on a global level and the other on a local level might further improve the results.

V. Conclusion

A variety of machine learning algorithms are capable of solving the problem. To the best of our knowledge, this is the first description of a successful application of a plain SGD for brain tissue segmentation. Besides accuracy, different runtime behaviours and the clinical application might influence the choice of a preferred algorithm. Careful tuning of hyperparameters is required to yield good results.

The major challenge remains the quality of ground-truth data for training and validation. As long as the test set is the output from another (imperfect) algorithm, any approach is just an approximation of the other mechanism.

In the current setup, the number of available features was limited to seven which is known to be on the lower bound for these algorithms. A deep learning approach that directly processes the raw input data and implicitly learns how to extract features might be a better choice in this case. Whether such a neural network outperforms the classical machine learning algorithms remains to be investigated.

Acknowledgement

The calculations in the present work were performed on UBELIX (http://www.id.unibe.ch/hpc), the HPC cluster at the University of Bern.

References

[1] G. F. Busatto, G. E. Garrido, O. P. Almeida, C. C. Castro, C. H. Camargo, C. G. Cid, C. A. Buchpiguel, S. Furuie, and C. M. Bottino, "A voxel-based morphometry study of temporal lobe gray matter reductions in alzheimers disease," *Neurobiology of aging*, vol. 24, no. 2, pp. 221–231, 2003.

[2] S. Price, D. Paviour, R. Scahill, J. Stevens, M. Rossor, A. Lees, and N. Fox, "Voxel-based morphometry detects patterns of atrophy that help differentiate progressive supranuclear palsy and parkinson's disease," *Neuroimage*, vol. 23, no. 2, pp. 663–669, 2004.

[3] C. Rummel, N. Slavova, A. Seiler, E. Abela, M. Hauf, Y. Burren, C. Weisstanner, S. Vulliemoz, M. Seeck, K. Schindler *et al.*, "Personalized structural image analysis in patients with temporal lobe epilepsy," *Scientific reports*, vol. 7, no. 1, p. 10883, 2017.

[4] M. Yaqub, M. K. Javaid, C. Cooper, and J. A. Noble, "Investigation of the role of feature selection and weighted voting in random forests for 3-d volumetric segmentation," *IEEE Transactions on Medical Imaging*, vol. 33, no. 2, pp. 258–271, feb 2014.

[5] S. Pereira, A. Pinto, J. Oliveira, A. M. Mendrik, J. H. Correia, and C. A. Silva, "Automatic brain tissue segmentation in MR images using random forests and conditional random fields," *Journal of Neuroscience Methods*, vol. 270, pp. 111–123, sep 2016.

[6] P. Anbeek, K. L. Vincken, M. J. van Osch, R. H. Bisschops, and J. van der Grond, "Probabilistic segmentation of white matter lesions in MR imaging," *NeuroImage*, vol. 21, no. 3, pp. 1037–1044, mar 2004.

[7] C. A. Cocosco, A. P. Zijdenbos, and A. C. Evans, "A fully automatic and robust brain MRI tissue classification method," *Medical Image Analysis*, vol. 7, no. 4, pp. 513–527, dec 2003.

[8] S. 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, mar 2000.

[9] S. Bauer, L.-P. Nolte, and M. Reyes, "Fully automatic segmentation of brain tumor images using support vector machine classification in combination with hierarchical conditional random field regularization," in *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 354–361.

[10] Y. LeCun, L. Bottou, G. B. Orr, and K. R. Mller, "Efficient BackProp," in *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1998, pp. 9–50.

[11] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Physica-Verlag HD, 2010, pp. 177–186.

[12] B. N. Li, C. K. Chui, S. Chang, and S. Ong, "Integrating spatial fuzzy clustering with level set methods for automated medical image segmentation," *Computers in Biology and Medicine*, vol. 41, no. 1, pp. 1–10, jan 2011.

[13] I. Despotovic, E. Vansteenkiste, and W. Philips, "Spatially coherent fuzzy clustering for accurate and noise-robust image segmentation," *IEEE Signal Processing Letters*, vol. 20, no. 4, pp. 295–298, apr 2013.

[14] K. M. Pohl, J. Fisher, W. E. L. Grimson, R. Kikinis, and W. M. Wells, "A bayesian model for joint segmentation and registration," *NeuroImage*, vol. 31, no. 1, pp. 228–239, may 2006.

[15] J. Ashburner and K. J. Friston, "Unified segmentation," *NeuroImage*, vol. 26, no. 3, pp. 839–851, jul 2005.

[16] J. C. Moreno, V. S. Prasath, H. Proença, and K. Palaniappan, "Fast and globally convex multiphase active contours for brain MRI segmentation," *Computer Vision and Image Understanding*, vol. 125, pp. 237–250, aug 2014.

[17] A. Ortiz, J. Gorriz, J. Ramirez, and D. Salas-Gonzalez, "Improving MR brain image segmentation using self-organising maps and entropy-gradient clustering," *Information Sciences*, vol. 262, pp. 117–136, mar 2014.

[18] D. C. Van Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, K. Ugurbil, W.-M. H. Consortium *et al.*, "The wu-minn human connectome project: an overview," *Neuroimage*, vol. 80, pp. 62–79, 2013.

[19] J. Mazziotta, A. Toga, A. Evans, P. Fox, J. Lancaster, K. Zilles, R. Woods, T. Paus, G. Simpson, B. Pike *et al.*, "A probabilistic atlas and reference system for the human brain: International consortium for brain mapping (icbm)," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 356, no. 1412, pp. 1293–1322, 2001.

[20] B. Fischl, "Freesurfer," *Neuroimage*, vol. 62, no. 2, pp. 774–781, 2012.

[21] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Advances in neural information processing systems*, 2011, pp. 109–117.

[22] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[24] T. S. Yoo, M. J. Ackerman, W. E. Lorensen, W. Schroeder, V. Chalana, S. Aylward, D. Metaxas, and R. Whitaker, "Engineering and algorithm design for an image processing api: a technical report on itk-the insight toolkit," *Studies in health technology and informatics*, pp. 586–592, 2002.

[25] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[26] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 116.