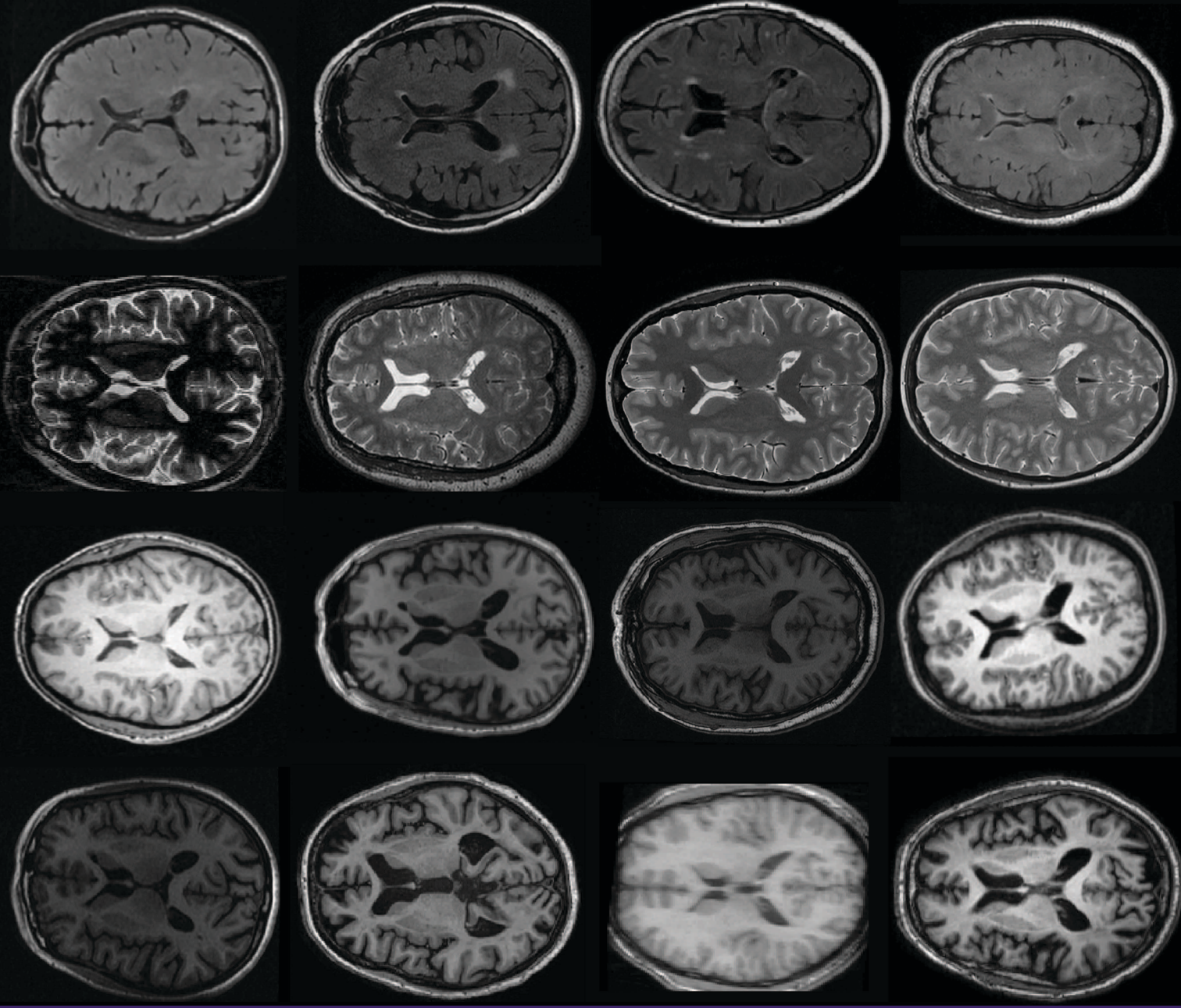


Transfer Learning for Medical Image Segmentation

Annegreet van Opbroek

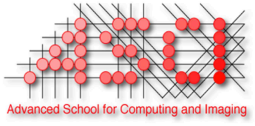
Transfer Learning for Medical Image Segmentation

Annegreet van Opbroek

Cover design by Annegreet van Opbroek
Thesis layout by Annegreet van Opbroek
Epigraph: Adapted from https://waitbutwhy.com/2015/01/artificial-intelligence-revolution-1.html

The work in this thesis was conducted at the departments of Radiology and Medical Informatics of the Erasmus MC, Rotterdam, the Netherlands. The research was performed as part of the research project 'Transfer learning in biomedical image analysis', which is financed by The Netherlands Organization for Scientific Research (NWO).

This work was carried out in the ASCI graduate school.
ASCI dissertation series number 388.



For financial support for the publication of this thesis, the following organizations are gratefully acknowledged: Alzheimer Nederland, the ASCI graduate school, the Department of Radiology and Nuclear Medicine, Erasmus MC, and Quantib BV.

ISBN 978-94-6299-952-7
Printed by Ridderprint BV.

© 2018 A.G. van Opbroek
All rights reserved. No part of this thesis may be reproduced or transmitted in any form or by any means without prior permission of the copyright owner.

Transfer Learning for Medical Image Segmentation

Transfer learning voor
medische beeldsegmentatie

Proefschrift

ter verkrijging van de graad van doctor aan de
Erasmus Universiteit Rotterdam
op gezag van de rector magnificus

Prof. dr. H.A.P. Pols

en volgens besluit van het College voor Promoties.

De openbare verdediging zal plaatsvinden op
woensdag 6 juni 2018 om 11:30 uur

door

Anna Gretha van Opbroek
geboren te Woubrugge

Erasmus University Rotterdam

Promotiecommissie

Promotor:	Prof.dr. W.J. Niessen
Overige Leden:	Dr. B. Glocker Dr. I. Išgum Prof.dr. A. van der Lugt
Copromotor:	Dr. M. de Bruijne

Nothing will make you appreciate human intelligence as much as learning about how unbelievably challenging it is to try to make a computer as smart as we are.

Contents

Chapter 1

General Introduction

Imaging plays a prominent role in the biomedical domain, both in the clinic and in medical and life-science research. In the clinic, imaging aids in diagnosis, prognosis, treatment planning and guidance, for example by providing information on a subject's anatomy or pathology and the ability to better monitor surgical procedures such as image-guided interventions. In epidemiological and clinical research, imaging provides insight into onset and progression of various diseases and response to treatment. Here, comparing images, both within and between subjects, can provide valuable information. This is ideally done in a quantitative manner; using so-called quantitative imaging biomarkers consisting of e.g. sizes and shapes of the various imaged tissues and structures. Quantitative imaging biomarkers can be used to study disease progression and differences between patient groups or to compare subjects with a database of healthy individuals to reveal deviating measures. MRI and CT scans are most commonly used to extract quantitative imaging biomarkers, since they provide three-dimensional images of anatomy. In order to obtain such quantitative information, these images can be segmented into the tissues or structures of interest, as shown in Figure 1.1. Manual segmentation however, has the drawback of being very time consuming as well as being prone to inter- and intra-observer variability, which complicates comparison. Therefore, automatic segmentation has gained considerable attention over the past decades.

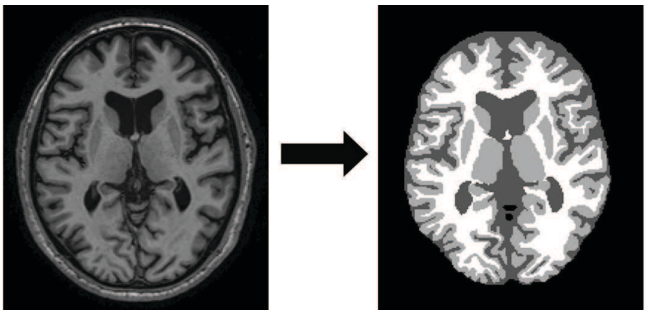


Figure 1.1: *Example of medical image segmentation. Left: a slice of an MR image of the brain. Right: a segmentation of the image into background (black) and three brain tissues of interest: cerebrospinal fluid, grey matter, and white matter (dark grey, light grey, and white respectively).*

1.1 Machine Learning for Image Segmentation

Supervised machine learning has proven to be a very suitable technique for automatic medical image segmentation. Here, a decision framework is trained based on examples, which saves the developer from performing the tedious task of programming how decisions are made. Machine-learning methods for image segmentation are generally developed based on images (or parts of images) that have been segmented manually; the so-called *training images*. From these images, *training samples* are selected for each of the *classes*, i.e. the tissues or structures under consideration. These samples usually consist of voxels or patches of voxels. For these samples, *features* are extracted and the samples are represented in a *feature space*. Note that such a feature space can be high dimensional in case of many features (e.g. dozens, hundreds, or even thousands of features). In the feature space, a *classifier* is optimized (or *trained*) in order to distinguish the different classes. When the decision framework is trained, it can be used to automatically segment a new image, the *test image*. Hereto, the test image is split up in samples; *test samples*. Then, the same features are extracted for the test samples as for the training samples. Finally, the classifier is applied in order to make a decision per test sample on the class it belongs to. The quality of this automatic segmentation can be evaluated if a reference manual segmentation is available for the test image, for example by calculating the percentage of correctly classified samples.

Such supervised methods generally work well if the training dataset is sufficiently large and representative of the test data, i.e. if training and test samples follow the same distribution in the feature space used for classification. However, performance can deteriorate dramatically if training and test samples follow different distributions. This can occur because of differences in the scanners or acquisition parameters used to acquire training and test data or because of differences in subject groups (e.g. age, disease stage, diseased versus healthy). Supervised-learning methods that work well on images from one study might therefore not perform well on images from a different study, since they often use different scanners, parameters, or comprise different patient groups. This seriously hampers the use of these methods in practice, since obtaining a large enough representative training set for every study can be very time consuming.

1.2 Transfer Learning

In this thesis, I study the added value of *transfer learning* in case of differences between training and test images. Transfer learning is a relatively new field of machine learning where

training and test data as well as training and test tasks may be somewhat different.[1] It distinguishes between *target data*, i.e. training data that is representative of the test data, and *source data*, i.e. training data that is similar, yet somewhat different from the test data to be segmented. Pan and Yang [71] provide a comprehensive overview of transfer-learning literature (although many new methods have been published since).

We can identify four differences between source and target data: differences in used features, differences in classes, differences in data distributions $P(\boldsymbol{x})$, and differences in labeling functions $P(y|\boldsymbol{x})$. In this thesis, I focus on situations with differences in data distributions and labeling functions, hence a difference in the joint distribution $P(\boldsymbol{x}, y)$. This setting corresponds to machine-learning based segmentation of medical images from different scanners, scanning parameters or patient groups. Source and target data are assumed to have the same class labels and the same features. I investigated two different data settings, *transductive transfer learning*, where the source data is labeled and the target data is unlabeled (i.e. no class labels are available for target samples) and *inductive transfer learning*, where labeled training data is available from both source and target (generally, much more labeled source than target data).

This thesis presents various transfer-learning methods for medical image segmentation. I split these methods op into two approaches, which compensate for the difference between source and target data at different stages of the classification framework. First of all, I study *transfer classification*, where differences between source and target data are incorporated in the classifier. Secondly, I study *feature-representation transfer*, where differences between source and target data are reduced in the feature representation used for classification. The added value of combining both approaches is also studied.

1.3 Neuro-Image Segmentation

The methods presented in this paper are applied to MR neuro-image segmentation. Automated MR neuro-image-segmentation methods are widely used to investigate the value of quantitative imaging biomarkers for studying development and diagnosis of brain diseases. For example, atrophy (i.e. shrinkage) of the brain in general and the hippocampus specifically has shown to be a biomarker for Alzheimer's disease; atrophy of the frontotemporal lobe for frontotemporal dementia; and brain atrophy and volume and location of white-matter lesions for multiple sclerosis. Neuro-image-segmentation methods are also used to study brain

[1] Transfer Learning is sometimes also called *domain adaptation*, although some fields use this term to describe the situation where the data's labeling function $P(y|\boldsymbol{x})$ is unchanged.

changing over time in healthy individuals to gain insight in healthy brain aging. Many automatic segmentation methods based on supervised learning are available for MR brain images, including whole-brain segmentation [50, 57], brain-tissue segmentation [24, 64, 67], white-matter-lesion segmentation [10, 25, 36], brain-tumor segmentation [65, 74, 105], and brain-structure segmentation [28, 95]. Investigating the power of these measures for studying development and diagnosis requires methods that are invariant to dataset-specific properties such as used scanner and patient characteristics.

Many neuro-image-segmentation methods that are based on supervised learning use image normalization to compensate for distribution differences between images [17, 44, 54, 60, 69, 78, 81, 83, 113]. While image normalization often improves performance of supervised-learning techniques, we can identify two shortcomings. First of all, normalization uses sample information, but no information on the class label of samples. It can therefore handle differences in $P(\boldsymbol{x})$, but struggles to overcome differences in $P(\boldsymbol{x}, y)$. Differences in used scanners or scanner parameters are often class-label specific as they image different tissues differently. Therefore, if class labels are available, it would be useful to treat samples from different classes differently. Secondly, image normalization is generally performed on voxel intensities only, rather than in the feature space used for classification. As a result, derived features are likely not normalized correctly. To study the added value of transfer learning over image normalization, I compare the performance of the proposed transfer-learning methods with that of supervised-learning methods with image normalization.

Another method for segmentation besides supervised learning is multi-atlas segmentation, which is based on *image registration*. Here, a training image (or *atlas*) and its corresponding manual segmentation are transformed in such a way that the transformed training image best matches a test image. Image registration is especially well used in neuro image segmentation because brains are relatively similar between subjects. Many multi-atlas-segmentation methods have been published for brain-tissue and brain-structure segmentation [18, 22, 31, 75, 111]. In case of differences in appearance between source and target data, image registration can be optimized based on the mutual information measure [92]. Image registration can also be used in combination with supervised learning, for example to extract new representative training voxels from a test image [18], or to combine atlas features and appearance features in a classifier [28, 95]. In many of the applications of this thesis, I compare the performance of our method with that of registration-based methods. I also study the combination of atlas features and appearance features in a classifier for hippocampus segmentation.

1.4 Thesis Aims and Outline

In this thesis, transfer-learning techniques are developed, applied, and evaluated for neuroimage segmentation across scanners, scanner protocols, and patient groups. These methods are compared to traditional non-transfer machine-learning techniques. We study various applications that are all based on voxelwise classification: brain-tissue segmentation, white-matter-lesion segmentation, whole-brain segmentation, and hippocampus segmentation.

In Chapter 2, we present a non-transfer baseline method we developed as starting point for the other methods presented in the later chapters of this thesis. This method uses voxelwise classification with intensity features and derived Gaussian-scale-space features together with a support vector machine (SVM) [21] for classification. The method is evaluated on the single-scanner dataset of the 2013 MRBrainS challenge [64], where it won second place.

Chapter 3 presents four transfer classifiers for the inductive data setting (the situation with much labeled source data and some labeled target data). It compares the performance against that of traditional (non-transfer) classifiers for brain-tissue and white-matter-lesion segmentation. Three of the studied transfer classifiers weight source samples (voxels) according to target data resemblance, one transfer classifier uses the source data to regularize a classifier trained on the target data. We investigate the added value of the transfer classifiers for different amounts of labeled target data and also study the influence of various commonly used normalization techniques, for both the transfer and non-transfer classifiers.

Chapter 4 proposes a method to weight training images rather than individual samples according to target data resemblance. It assumes the presence of a heterogeneous set of training images (consisting of only source images or both source and target images) that are weighted such that the distribution of the weighted training samples best resembles the distribution of the test samples. The generated image weights are then used in a weighted classifier. We investigate three methods that use different measures for the difference between distributions. The performance of the proposed methods is evaluated on brain-tissue segmentation, white-matter-lesion segmentation, and whole-brain segmentation.

Chapter 5 presents a feature-representation transfer method for the transductive setting. Here, unlabeled images of subjects scanned with both the source and target scanner/scanning parameters are used to determine a mapping in the feature space from source to target voxels. This mapping is then applied to feature values of labeled source samples in order to calcu-

late feature values that are observed in target images. This way, source-image samples are transformed to the feature distribution of target samples. Next, a classifier is trained on the transformed source samples and applied to the target data. The method is compared with regular supervised-learning methods on hippocampus segmentation.

Chapter 6 combines transfer classification and feature representation transfer. For transfer classification, we compare two of the image weighting methods presented in Chapter 4 and a newly proposed method. For feature representation transfer, we used an existing and a newly developed kernel learning method, which both aim at learning a kernel that makes source and target feature representations more similar. The image weighting and kernel learning methods are combined, optimized either individually or jointly, to study the added value of using any or both of the two approaches. Experiments are performed on brain-tissue segmentation, white-matter-lesion segmentation, and hippocampus segmentation.

While this thesis applies the proposed methods to neuro-image segmentation only, most of the presented methods are applicable outside neuro-image segmentation and the conclusions drawn are more widely valid in medical image segmentation. I will discuss the validity on other medical-image-segmentation tasks in the general discussion in Chapter 7.

Chapter 2

Automated Brain-Tissue Segmentation by Multi-Feature SVM Classification

Annegreet van Opbroek, Fedde van der Lijn, & Marleen de Bruijne. ***Grand Challenge on MR Brain Image Segmentation workshop, MICCAI 2013****.*

We present a method for automated brain-tissue segmentation through voxelwise classification. Our algorithm uses manually labeled training images to train a support vector machine (SVM) classifier, which is then used for the segmentation of target images. The classification incorporates voxel intensities from a T1-weighted scan, an IR scan, and a FLAIR scan; features to encode the voxel position in the image; and Gaussian-scale-space features and Gaussian-derivative features at multiple scales to facilitate a smooth segmentation.
An experiment on data from the MRBrainS13 brain-tissue-segmentation challenge showed that our algorithm produces reasonable segmentations in a reasonable amount of time.

2.1 Introduction

The segmentation of brain images can provide useful information about neuro-generative diseases such as dementia and multiple sclerosis, which is useful both in research and clinical practice. Manual segmentation of brain images is a tedious task however, which is why a variety of methods have been developed for automated segmentation.

Three types of automated brain-tissue-segmentation techniques can be distinguished: techniques that use manually segmented images to train a segmentation algorithm, techniques that require no manually segmented training images, and techniques that use atlases. Methods that fall in the first category are usually based on supervised classification. In supervised classification labeled training data is used to extract features and train a classifier, such as a k-nearest neighbor (kNN) classifier [2], a random decision forest [119], or a support vector machine (SVM) [100]. The labeled training data used in the classification is usually obtained with the same scanner as the target data, but Van Opbroek et al. [100] propose a transfer-learning SVM that can deal with labeled training data from other scanners.

Methods that do not require manually labeled training data include clustering algorithms such as the fuzzy c-means algorithm [6] and mean-shift clustering [63].

A third option is to obtain a segmentation with the help of manually constructed atlases, which can be used to automatically select and label training data from a target image [18], to initialize an expectation-maximization algorithm [98], or to improve parameter estimation in classification with an EM algorithm [4].

In this paper we present a brain-tissue-segmentation framework based on supervised voxel-wise classification with an SVM classifier, which is a state-of-the-art machine-learning classifier. In contrast to other techniques [2][119][6][63] our segmentation scheme uses Gaussian-scale-space features and Gaussian-derivative features next to the image intensities, to facilitate a smooth segmentation of the different tissues. The performance of our method has been tested on 12 images from the brain-tissue segmentation challenge MRBrainS13[1].

2.2 Material and Methods

2.2.1 MRBrainsS13 Training and Test Data

Training and test images have been acquired at the UMC Utrecht in the Netherlands and concern patients with diabetes and matched controls (age$>$50) with varying degrees of atrophy and white-matter lesions. From all subjects a T1-weighted scan, a T1-weighted inversion recovery (IR) scan, and a fluid attenuated inversion recovery (FLAIR) scan have been obtained, all with $0.958 \times 0.958 \times 3.0$ mm^3 voxel size. The three sequences have been registered and the images have been bias corrected.

Five images were provided to train a segmentation algorithm. The training images were manually segmented into background and eight tissues: cortical gray matter (GM), basal ganglia, white matter (WM), white-matter lesions, cerebrospinal fluid (CSF) in the extracerebral space, the ventricles, the cerebellum, and the brainstem. Also, twelve test images were provided which had to be segmented into background, gray matter (cortical gray matter + basal ganglia), white matter (white matter + white-matter lesions), and cerebrospinal fluid (CSF in the extracerebral space + ventricles). Segmentation of the cerebellum and the brainstem was not included in the evaluation.

[1] http://mrbrains13.isi.uu.nl

2.2.2 Preprocessing

All images were normalized by a range-matching procedure that scaled the voxels within the mask so that the 4th percentile was mapped to zero, and the 96th percentage to one. Since the images of the challenge were already bias corrected, no further bias correction was performed.

2.2.3 Brain Segmentation

For the test images brain masks were created with multi-atlas segmentation. As atlases we used the five T1-weighted training scans, both in the original and in a left-right-flipped version. Brain masks were obtained for these ten atlases by binarizing the training labels (including the brainstem and the cerebellum). Each atlas image was registered to the unlabeled test images using Niftyreg [66] by computing an affine transformation, followed by a non-rigid deformation using a 5mm B-spline grid and normalized mutual information. A final mask was computed using STEPS [11]. This method deforms both atlas images and labels, selects per voxel location the five most similar atlases (based on local normalized cross correlation), and fuses their labels using STAPLE [112].

2.2.4 Features

From each of the three sequences, T1, IR, and FLAIR, 10 features were extracted:

- The intensity
- The intensity after convolution with a Gaussian kernel with $\sigma = 1, 2, 3$ mm^3
- The gradient magnitude of the intensity after convolution with a Gaussian kernel with $\sigma = 1, 2, 3$ mm^3
- The Laplacian of the intensity after convolution with a Gaussian kernel with $\sigma = 1, 2, 3$ mm^3.

The spatial information was incorporated by adding three spatial features: the x, y, and z coordinate of a voxel, divided by respectively the length, width, and height of the brain. This resulted in a total of 33 features. All features were scaled to zero mean and unit standard deviation.

2.2.5 SVM Classification

We performed supervised voxelwise classification with a soft-margin SVM [21]. An SVM learns a decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \phi(\boldsymbol{x}) + v_0$, where the model parameters $\boldsymbol{v}$ and v_0 are determined from the training data, and ϕ is a mapping to project a sample $\boldsymbol{x}_i$ into a feature space $\phi(\boldsymbol{x}_i)$. This mapping defines a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ by means of the kernel trick.

The model parameters $\boldsymbol{v}$ and v_0 are determined from the training data by optimizing the following criterion

$$\begin{aligned} \min_{v} \quad & \frac{1}{2}\|\boldsymbol{v}\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & y_i \boldsymbol{v}^T \phi(\boldsymbol{x}_i) + v_0 \geq 1 - \xi_i \\ & \xi_i \geq 0 \ . \end{aligned} \tag{2.1}$$

The first term, $\|\boldsymbol{v}\|^2$ maximizes the margin around the decision function, and $C\sum_{i=1}^{N}\xi_i$ minimizes the number of training samples that are either misclassified or lie within the margin. C is a parameter to trade off between maximizing the margin and minimizing $\sum_i \xi_i$ where a sample receives a value $\xi_i > 1$ if it is misclassified, a value $0 < \xi_i \leq 1$ if it is correctly classified but lies within the margin, and a value $\xi_i = 0$ otherwise.

We performed six-class classification to classify cortical GM, basal ganglia, WM, white-matter lesions, CSF in the extracerebral space, and the ventricles. Since SVMs are designed for two-class classification, the classification was extended to multi-class classification by one-versus-one classification which means that a total of 15 SVMs were trained to distinguish between the six classes.

For the SVM classification an implementation in LIBSVM [13] was used.

2.2.5.1 Classification Parameters

To enable for non-linear decision functions a radial basis kernel was chosen. A suitable value for the SVM parameter C and the kernel parameter γ were determined in a grid-search experiment on the five training images in which the total classification error was minimized. This resulted in values of $C = 8$ and $\gamma = 0.01$.

The SVM classifier was trained on a total of 10 000 samples per training image, which were randomly selected from inside the provided brain mask excluding the cerebellum and the brain stem.

2.2.6 Postprocessing

Postprocessing involved fusing the voxels segmented as cortical gray matter and basal ganglia, white matter and white-matter lesions, and CSF in the extracerebral space and the ventricles. Subsequently a closing algorithm was performed on the regions segmented as CSF, by a dilation of 1 voxel, followed by an erosion of 1 voxel. This was done to reduce the amount of voxels in the CSF that were segmented as WM or GM.

2.2.7 Outcome Measures

Segmentation results on the 12 test images were compared to the manual segmentations based on three measures:

- The DICE overlap [27]
- The modified Hausdorff distance (MHD) [48]
- The absolute volume difference (AVD) [5].

Structure	DICE (%) Mean	DICE (%) Std	MHD (mm) Mean	MHD (mm) Std	AVD (%) Mean	AVD (%) Std
Gray Matter	84.51	1.44	1.97	0.34	6.92	3.09
White Matter	88.30	0.98	2.41	0.50	6.79	5.19
Cerebrospinal fluid	78.00	5.43	3.31	0.80	25.98	18.49
Brain	95.05	0.53	2.79	0.82	3.51	1.61
All Intracranial Structures	95.86	1.32	4.02	1.20	5.67	3.14

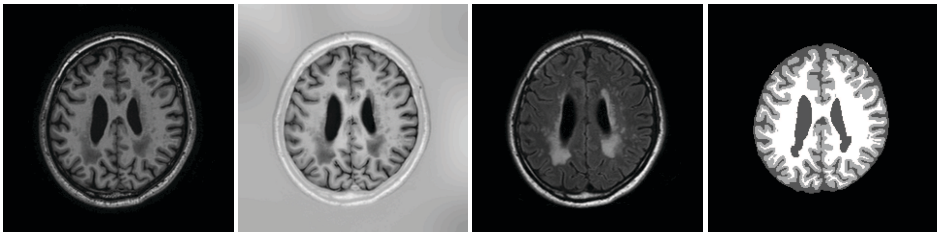
Table 2.1: *Results on 12 test images: mean and standard deviation (std) for GM, WM, CSF, brain (WM+GM), and all intracranial structures (WM+GM+CSF).*

2.3 Results

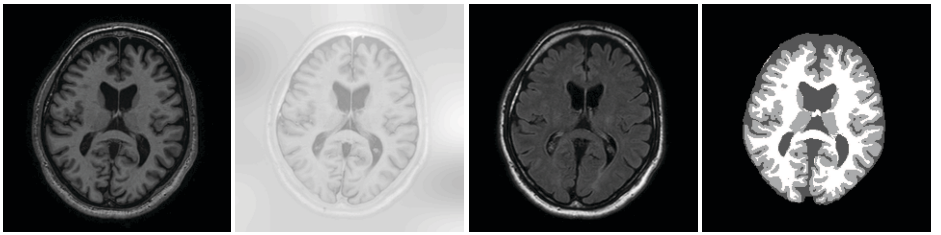
Table 2.1 shows the mean and standard deviation (std) of the scores. The evaluation measures were calculated for 5 tissues: GM, WM, CSF, brain (WM+GM), and all intracranial structures (WM+GM+CSF). For the DICE score our algorithm scored best on white-matter segmentation, with a mean DICE coefficient of 88.3%, and slightly lower for gray matter, with a mean DICE of 84.5%. The most errors were made in the CSF, which had a mean DICE score of only 78.0%. Also on the other two scores, MHD and AVD, white matter and gray matter had a better score than CSF.

Example segmentations for slices from images 3, 8, and 10 are presented in Fig. 2.1. Fig. 2.1(a), (e),(i) show the T1-weighted images, Fig. 2.1 (b),(f),(j) show the T1-weighted IR scans, Fig. 2.1(c),(g),(k) show the FLAIR scans, and Fig. 2.1(d),(h), (l) show the segmentations. Image 3 in Fig. 2.1(a)-(d) has a very large amount of lesions, Image 8 in fig. 2.1(e)-(h) is the image that overall scored the best, and image 10 in Fig. 2.1(i)-(l) overall scored worst.

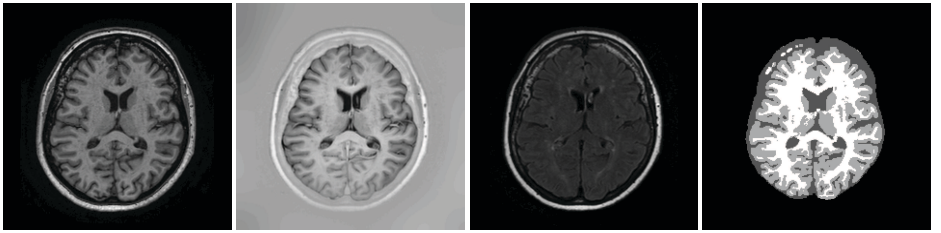
For all images the determined brain mask was too large in the front, as can be seen in the segmentations in Fig. 2.1(d),(h),(l). In most images this led to voxels in the front of the image being erroneously classified as either white matter or gray matter tissue. This effect is most prominent in Fig. 2.1(l), where WM and GM clusters have appeared in the CSF, but it can also been seen in the segmentations in Fig. 2.1(d),(h). In images with a large amount of lesions, as in Image 3, lesion voxels were sometimes erroneously classified as CSF, as is shown in Fig. 2.1(d).



(a) Img 3, T1 (b) Img 3, IR (c) Img 3, FLAIR (d) Img 3, Segm



(e) Img 8, T1 (f) Img 8, IR (g) Img 8, FLAIR (h) Img 8, Segm



(i) Img 10, T1 (j) Img 10, IR (k) Img 10, FLAIR (l) Img 10, Segm

Figure 2.1: *T1, IR, FLAIR images (Img) and segmentations (Segm) of 3 of the 12 test images. Image 3 in (a)-(d) contains a large amount of lesions, image 8 in (e)-(h) was given the overall best score, image 10 in (i)-(l) was given the overall worst score.*

2.4 Discussion

We have presented an algorithm for automated brain extraction and brain-tissue segmentation. The brain-extraction algorithm is based on multi-atlas segmentation with the STEPS [11] algorithm; the tissue classification is based on voxelwise SVM classification. In the voxelwise classification T1, IR, and FLAIR intensities, spatial features, Gaussian-scale-space features and Gaussian-derivative features were used.

Our algorithm produced reasonable segmentations which were generally quite smooth without further spatial regularization because of the use of the Gaussian-scale-space features and the Gaussian-derivative features. In some slices however, mainly around the basal ganglia, the segmentations were not completely smooth, which was caused by the low contrast between the basal ganglia and the surrounding white matter.

The largest errors were made in the segmentation of the CSF in the extracerebral space, which was mainly because the determined brain mask was too big in the frontal lobe due to a slight misregistration of the atlases. As a result, skull voxels were incorporated in the brain mask, which were sometimes segmented as white or gray matter. As a post-processing step a closing algorithm was performed on the CSF tissue, which segmented some of these voxels as CSF. We believe that refining the masks by including a background class in the SVM classification may improve the results.

Other weaknesses of our algorithm are a slight under segmentation of the basal ganglia, and the misclassification of voxels in the center of large white-matter lesions that are close to the ventricles, which were erroneously segmented as CSF. This second problem could be resolved by excluding the lesions from the tissue segmentation and including a separate lesion-classification step. This separate classification step allows for the exclusion of spatial features from the feature set, which can be very misleading features for lesion segmentation when only a small number of training images is available.

The total runtime of our algorithm per test image was 10 times 8 minutes to perform the registrations for the image mask, 25 seconds to determine the image features, 1.5 minutes to train the SVM classifiers (note that this only needs to be done once for segmentation of all images), and 35 minutes for classification of the test image. The registrations for the image mask were computed on a cluster with AMD Opteron 2216 2.4GHz nodes without multi-threading,

the rest was implemented in Matlab and computed on a machine with an Intel Xeon E5620 2.40 GHz CPU. For a voxelwise classification 35 minutes is relatively long, which is due to the fact that a total of 15 one-vs-one SVMs were calculated. The calculation time could be drastically reduced by training on only three labels: gray matter, white matter, and CSF. In a cross-validation experiment on the training set this only slightly increased the mean classification error from 13.9% to 14.3%, but decreased the calculation time with approximately a factor of 5.

To conclude, the proposed multi-feature SVM classification produces reasonable segmentations in a reasonable amount of time. We believe that if the registrations of the training masks to the target images could be improved, and a separate lesion-segmentation algorithm could be included in the segmentation, the resulting segmentations would come close to those of human observers.

Chapter 3

Transfer Learning Improves Supervised Image Segmentation Across Imaging Protocols

Annegreet van Opbroek, M. Arfan Ikram, Meike W. Vernooij, & Marleen de Bruijne. ***IEEE Transactions on Medical Imaging, 2015, 34(5)****, 1018-1030.*

The variation between images obtained with different scanners or different imaging protocols presents a major challenge in automatic segmentation of biomedical images. This variation especially hampers the application of otherwise successful supervised-learning techniques which, in order to perform well, often require a large amount of labeled training data that is exactly representative of the target data.
We therefore propose to use transfer learning for image segmentation. Transfer-learning techniques can cope with differences in distributions between training and target data, and therefore may improve performance over supervised learning for segmentation across scanners and scan protocols. We present four transfer classifiers that can train a classification scheme with only a small amount of representative training data, in addition to a larger amount of other training data with slightly different characteristics. The performance of the four transfer classifiers was compared to that of standard supervised classification on two MRI brain-segmentation tasks with multi-site data: white matter, gray matter, and CSF segmentation; and white-matter-/MS-lesion segmentation.
The experiments showed that when there is only a small amount of representative training data available, transfer learning can greatly outperform common supervised-learning approaches, minimizing classification errors by up to 60%.

3.1 Introduction

Segmentation of biomedical images plays a crucial role in many medical imaging applications, forming an important step in enabling quantification in medical research and clinical practice. Since manual segmentation is very time consuming and prone to intra- and inter-observer variations, a variety of techniques have been developed to perform segmentation automatically.

Many successful approaches to automatic segmentation rely on voxelwise classification by supervised-learning techniques. In supervised learning (manually) labeled training data is used to train a classification scheme for the target data. First, features are extracted from the training and target data, after which a classifier is trained. This classifier can then be used to segment the target data into the different tissue classes, based on the extracted features.

Examples of successful voxelwise-classification methods can, among many other applications, be found in brain-tissue, lesion, cartilage, and plaque segmentation. Anbeek et al. [2] performed brain-tissue segmentation by a kNN classifier with intensity and spatial features. The same classification framework was also used for segmentation of white-matter lesions [3]. Geremia et al. [36] performed MS-lesion segmentation with a spatial decision forest classifier on local and context features. Here, local features consisted of voxel intensities, while context features consisted of mean intensities of a three-dimensional box around the voxel. Folkesson et al. [33] performed knee-cartilage segmentation with a kNN classifier with intensity and spatial features, as well as intensity after convolution with a Gaussian, and first-, second-, and third-order derivative features. Liu et al. [61] performed plaque-component segmentation by first performing a voxelwise classification with a Parzen classifier on features like intensity and distance to the lumen. Next, the region boundaries were determined with an active-contour model in order to eliminate isolated voxels.

In order for supervised-learning algorithms to perform well, the used training data needs to be representative of the target data. However, in medical image segmentation a sufficient amount of exactly representative manually labeled training data is often not available because of between-patient variability or because images are acquired with different scanners and/or different scan protocols.

We propose to perform segmentation through a different type of machine learning, called *transfer learning*. Transfer-learning algorithms exploit similarities between different classification problems or datasets to facilitate the construction of a new classification model. They possess the ability of supervised-learning algorithms to capture class-specific knowledge in the training phase without requiring exactly representative training data. Except for a preliminary study presented in [100], to the best of our knowledge transfer learning has not yet been applied to medical image segmentation.

The purpose of our study was to investigate whether transfer-learning techniques can improve upon regular supervised segmentation of images obtained with different scan protocols. We compare the performance of four transfer classifiers with that of standard supervised-learning classifiers. All four transfer classifiers use training data from sources other than the target source, which was acquired with different scan protocols and at different scanners, as well as a small amount of representative training data from the target source acquired with the same protocol as the target data. We performed experiments on voxelwise MRI brain-tissue segmentation and white-matter-lesion segmentation.

This paper is organized as follows: first some background information on transfer learning is given in Section 3.2. Section 3.3 describes the four transfer classifiers we used. Section 3.4 describes the experiments. Section 3.5.1 presents the performance of the four classifiers on brain-tissue segmentation, and Section 3.5.2 on MS-lesion and white-matter-lesion segmentation. The conclusion and discussion are given in Section 3.6.

3.2 Background

Transfer learning is a relatively new form of machine learning that allows for differences between training and target domains, tasks, and distributions. This means that training and test data may follow different distributions $P(\boldsymbol{x})$, may have different labeling functions $P(y|\boldsymbol{x})$, may have different features, and may even consist of different classes. In the transfer-learning literature data that follows the same distribution, has the same labeling function, and the same features is often referred to as data that comes from the same *source*. The goal of transfer learning is to learn a classification algorithm for the target data, that benefits from already available data that originates from different sources, i.e. data that is somehow similar, but not necessarily exactly representative for the target data. This approach is opposed to that of traditional supervised-learning algorithms, which assume that training and target data come from the same source.

Pan and Yang [71] provide an overview of the transfer-learning literature, where they distinguish between three types of transfer learning: inductive transfer learning, transductive transfer learning, and unsupervised transfer learning. In this paper we are dealing with inductive transfer learning, where the training and target data may have different labeling functions $P(y|\boldsymbol{x})$, as well as different features and/or prior distributions $P(\boldsymbol{x})$. We assume that a small number of labeled training samples from the target source is available, the so-called *same-distribution training data*, and aim to transfer knowledge from a much larger amount of labeled training data that is available from sources other than the target data, the so-called *different-distribution training data*. Inductive transfer learning assumes that even though labeling functions vary between training and target sources, they are still somewhat similar, in such a way that different-distribution sources give some extra information in areas of the feature space where same-distribution training data is scarce.

We present four transfer classifiers that use this same- and different-distribution training data,

all based on support vector machine (SVM) classification. Three of the four classifiers use sample weighting. First of all, the Weighted SVM [117], in which both same- and different-distribution training samples are used for training, the latter with a lower weight than the former. Secondly, the Reweighted SVM, which we proposed in [100], which is an extension to the Weighted SVM where iteratively the weights of misclassified different-distribution training samples are reduced. And thirdly, TrAdaBoost [23], which builds a boosting classifier for transfer learning by iteratively increasing the weights of misclassified same-distribution samples while reducing the weights of misclassified different-distribution samples. Removing misleading different-distribution samples is considered a common approach in transfer learning [55]. The fourth transfer classifier presented in this paper, Adaptive SVM [118], is not based on sample weighting. The Adaptive SVM trains an SVM on the same-distribution samples only, with the restriction that the resulting classifier should be close to an SVM on the different-distribution samples. The next section will discuss the four transfer classifiers in detail.

3.3 Methods

Let $\boldsymbol{x}_i \in \mathbb{R}^n$ denote a training sample i which is a vector containing a value for each of the n features. We assume to have a total of N_s same-distribution training samples $\boldsymbol{x}_i^s$ ($i = 1,2,\ldots,N_s$) with their corresponding labels y_i^s. The total of all same-distribution training samples is denoted by $T_s = \{\boldsymbol{x}_i^s, y_i^s\}_{i=1}^{N_s}$. In a similar way, the different-distribution training samples are denoted by $T_d = \{\boldsymbol{x}_i^d, y_i^d\}_{i=1}^{N_d}$, so that there is a total training set $T = T_s \cup T_d$ of size $N = N_s + N_d$. For the moment we assume $y_i^s, y_i^d \in \{1,-1\}\ \forall i$, but all the presented algorithms can easily be adapted to more than two classes by one-vs-one or one-vs-all classification.

We compared the performance of four transfer classifiers with the performance of the traditional SVM classifier. The traditional, soft-margin SVM by Cortes and Vapnik [21] constructs a linear decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \boldsymbol{x} + v_0$, where $\boldsymbol{v}$ and v_0 are model parameters that have to be optimized from the data by minimizing the SVM optimization criterion:

$$\min_{\boldsymbol{v}} \quad \frac{1}{2}\|\boldsymbol{v}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{3.1}$$

$$\begin{aligned} \text{s.t.} \quad & y_i(\boldsymbol{v}^T\boldsymbol{x}_i + v_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & \forall\, \boldsymbol{x}_i \ . \end{aligned}$$

In this optimization the term $\|\boldsymbol{v}\|^2$ maximizes the margin around the decision function, and $C\sum_{i=1}^{N}\xi_i$ minimizes the number of samples that are either misclassified or lie within the margin. C is the SVM parameter to trade off between maximizing the margin and minimizing $\sum_i \xi_i$, where a sample $\boldsymbol{x}_i$ receives a value $\xi_i > 1$ if it is misclassified, a value $0 < \xi_i \leq 1$ if it is correctly classified but lies within the margin, and a value $\xi_i = 0$ otherwise.

The original soft-margin SVM presented above can only produce linear decision functions. By using kernel learning one can obtain a non-linear decision function [82]. In kernel SVM a map ϕ is created that maps every sample $\boldsymbol{x}_i$ into a (possibly high-dimensional) feature space $\phi(\boldsymbol{x}_i)$, where an SVM decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \phi(\boldsymbol{x}) + v_0$ can be calculated. This results in a decision function that is linear in the new feature space $\phi(\boldsymbol{x})$, but depending on the mapping ϕ can be non-linear in the original feature space. Explicitly calculating $\phi(\boldsymbol{x})$ however could be very expensive. Luckily, the resulting decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \phi(\boldsymbol{x}) + v_0$ can be calculated without explicitly calculating the feature space $\phi(\boldsymbol{x})$, by use of a kernel matrix. This kernel matrix $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ gives the inner product between every combination of samples in the feature space $\phi(\boldsymbol{x})$. The decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \phi(\boldsymbol{x}) + v_0$ can be calculated entirely by means of inner products of samples in $\phi(\boldsymbol{x})$. This means that only the kernel matrix K needs to be calculated in order to obtain a non-linear decision function, and the accompanying mapping ϕ need not be calculated.

3.3.1 Weighted SVM

Sample weighting can be incorporated in the original SVM definition by assigning a weight $w_i \geq 0$ to every training sample $\boldsymbol{x}_i$, which indicates the importance of the sample. The sum of all weights, $|\boldsymbol{w}|$ should equal the total number of training samples, N. Incorporating sample weights in the SVM objective function results in the following objective function [13]

$$\min_{\boldsymbol{v}} \frac{1}{2}\|\boldsymbol{v}\|^2 + C\sum_{i=1}^{N} w_i\xi_i. \tag{3.2}$$

The constraints remain the same as in the traditional SVM.

Now, one way to perform transfer learning is by training a classifier on T where T_s samples receive a weight of one and T_d samples receive a weight of R_W, as is also done in the transfer SVM classifier presented by Wu and Dietterich [117]. This results in the following SVM objective function:

$$\min_{v} \frac{1}{2}\|\boldsymbol{v}\|^2 + CR_W \sum_{i:\boldsymbol{x}_i \in T_d} \xi_i + C \sum_{i:\boldsymbol{x}_i \in T_s} \xi_i. \tag{3.3}$$

In our experiments R_W was determined with cross validation as described in Sect. 3.4.1.

We will refer to this method as the Weighted SVM (WSVM).

3.3.2 Reweighted SVM

The second transfer classifier we studied is a transfer SVM we presented in a preliminary workshop paper [100]. This algorithm is an adaptation of the Weighted SVM that performs N_{it} iterations in which the weights of misclassified T_d samples are decreased in order to reduce the influence of T_d samples that contradict the rest of the data. This algorithm is a hybrid between the WSVM and TrAdaBoost, which is described in the next subsection.

The algorithm starts by giving each sample $\boldsymbol{x}_i$ a weight

$$w_i^1 = \begin{cases} R_R & \text{for } \boldsymbol{x}_i \in T_d \\ 1 & \text{for } \boldsymbol{x}_i \in T_s \end{cases}, \tag{3.4}$$

where similar to R_W in the WSVM the optimal value for R_R was set with cross validation. Then a total of N_{it} iterations are performed where for each iteration $t = 1, 2, \ldots, N_{it}$ first the weights are normalized to sum up to N,

$$\boldsymbol{w}^t = N \frac{\boldsymbol{w}^t}{|\boldsymbol{w}^t|}, \tag{3.5}$$

a weighted SVM classifier $f^t(\boldsymbol{x})$ is calculated from T and $\boldsymbol{w}^t$, and the weights for the next iteration are determined by

$$w_i^{t+1} = \begin{cases} w_i^t & \text{for } \boldsymbol{x}_i \in T_s \\ w_i^t \beta^{\frac{1}{2}|f^t(\boldsymbol{x}_i)-y_i|} & \text{for } (\boldsymbol{x}_i, y_i) \in T_d \end{cases} . \tag{3.6}$$

Here $\beta = 1/(1+\sqrt{2\ln N_d / N_{\text{it}}})$. This value equals the value used in the TrAdaBoost algorithm, and is derived from AdaBoost [34]. The final classifier is the weighted SVM with the weights from the last iteration.

We made a small adaptation to the algorithm presented in [100] to make it more robust. A disadvantage of reducing the weights of the T_d samples is that it can dis balance the classes, since reduction of weights may happen more in one class than in the other. This is undesirable because it will change the priors of the classes, which will shift the classifier towards the class with the lowest total weight. This problem was solved by in each iteration t normalizing the weights of the different classes, so that

$$\sum_{i:y_i=1} w_i^{t+1} = \sum_{i:y_i=1} w_i^t \text{ and } \sum_{i:y_i=-1} w_i^{t+1} = \sum_{i:y_i=-1} w_i^t \tag{3.7}$$

The resulting algorithm will be referred to as the Reweighted SVM (RSVM).

3.3.3 Transfer AdaBoost

The third transfer classifier we studied is Transfer AdaBoost [23] (TrAdaBoost), which is based on AdaBoost [34]. Like AdaBoost, TrAdaBoost is an iterative algorithm that reduces and increases the weights of training samples according to the outcome of a classifier. The final classifier is obtained by a weighted majority vote of the resulting classifiers.

The TrAdaBoost algorithm is trained on T where each sample $\boldsymbol{x}_i$ is given an initial weight w_i^1, which in our experiments was set with cross validation. In each iteration $t = 1, 2, \ldots, N_{it}$ the weights $\boldsymbol{w}^t$ are normalized to sum up to one, and a weighted classifier $f^t(\boldsymbol{x})$ is trained. The weights for the next iteration are then determined by

$$w_i^{t+1} = \begin{cases} w_i^t \beta^{\frac{1}{2}|f^t(\boldsymbol{x}_i)-y_i|} & \text{for } (\boldsymbol{x}_i, y_i) \in T_d \\ w_i^t \beta^{-\frac{1}{2}|f^t(\boldsymbol{x}_i)-y_i|} & \text{for } (\boldsymbol{x}_i, y_i) \in T_s \end{cases}, \tag{3.8}$$

for $\beta = 1/(1+\sqrt{2\ln N_d/N_{\text{it}}})$. Note that the weights of misclassified T_d samples are reduced by β, as in the Reweighted SVM, whereas the weights of misclassified T_s samples are increased by β, which is not the case in the Reweighted SVM, but is done in AdaBoost. After N_{it} iterations the final classification is determined by a weighted majority vote of the last $\lceil \frac{N_{it}}{2} \rceil$ classifiers $f^t(\boldsymbol{x})$:

$$f(\boldsymbol{x}) = \begin{cases} 1, & \prod_{t=\lceil \frac{N_{it}}{2} \rceil}^{N_{it}} \beta_t^{-f^t(\boldsymbol{x})} \geq 1 \\ -1, & \text{otherwise} \end{cases}, \tag{3.9}$$

where $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$, with ϵ_t the error of $f^t(\boldsymbol{x})$ on the T_s samples multiplied by the weight of each T_s sample:

$$\epsilon_t = \sum_{i:(\boldsymbol{x}_i,y_i)\in T_s} \frac{\frac{w_i^t}{2}|f^t(\boldsymbol{x}_i) - y_i|}{\sum_{i:(\boldsymbol{x}_i,y_i)\in T_s} w_i^t}. \tag{3.10}$$

This leads to a final classifier $f(\boldsymbol{x})$ in which the intermediate classifiers $f^t(\boldsymbol{x})$ that have a good performance on the T_s samples are given a large weight.

3.3.4 Adaptive SVM

The fourth transfer classifier is based on a different approach than the previous three. Instead of adding the T_d samples as training samples, one could also train a separate classifier on the T_d samples, and use this classifier to regularize a classifier trained on the T_s samples. This idea is presented in the Adaptive SVM [118] (A-SVM). First a regular SVM on the T_d samples is trained, resulting in a different-distribution classifier $f^d(\boldsymbol{x})$. This classifier is then adapted to the target data by training a "delta function", $\Delta f(\boldsymbol{x})$, which adapts $f^d(\boldsymbol{x})$ to obtain the final classifier $f(\boldsymbol{x})$:

$$f(\boldsymbol{x}) = f^d(\boldsymbol{x}) + \Delta f(\boldsymbol{x}) \tag{3.11}$$
$$= f^d(\boldsymbol{x}) + \boldsymbol{v}^T\boldsymbol{x} + v_0. \tag{3.12}$$

The parameters $\boldsymbol{v}$ and v_0 of $\Delta f(\boldsymbol{x})$ are determined from T_s by optimizing

$$\begin{aligned} \min_{\boldsymbol{v}} \quad & \frac{1}{2}\|\boldsymbol{v}\|^2 + C^s \sum_{i=1}^{N} \xi_i, \\ \text{s.t.} \quad & y_i f^d(\boldsymbol{x_i}) + y_i(\boldsymbol{v}^T\boldsymbol{x}_i + v_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & \forall\, (\boldsymbol{x}_i, y_i) \in T_s. \end{aligned} \tag{3.13}$$

Note that the first constraint differs from the definition of the original SVM in (3.1). This constraint favors an answer where the total classifier $f(\boldsymbol{x})$ correctly classifies the T_s samples. The regularization term $\|\boldsymbol{v}\|^2$ in the objective function on the other hand, favors an answer close to $\Delta f(\boldsymbol{x}) = 0$, resulting in a total classifier $f(\boldsymbol{x})$ that is close to the different-distribution classifier $f^d(\boldsymbol{x})$. The above optimization criterion therefore results in a classifier $f(\boldsymbol{x})$ that is close to $f^d(\boldsymbol{x})$, but is also adapted to improve classification on the T_s samples.

Contrary to the parameter C in (3.1) the cost factor C^s in (3.13) does not balance between optimization of the margin and classification of the training samples. The role of C^s is to balance between a classifier $f(\boldsymbol{x})$ that is close to $f^d(\boldsymbol{x})$, and correctly classifying the T_s samples, where a higher value for C^s gives a larger weight to the T_s samples. As with the parameters in the other transfer classifiers, in our experiments C^s was set with cross validation.

Similar to the original SVM, A-SVM can also be used with kernels, by changing $\boldsymbol{x}_i$ in (3.12) and (3.13) to $\phi(\boldsymbol{x}_i)$.

An advantage of the A-SVM is that the classifier on the T_d samples only has to be calculated once, which reduces the computational load of the classifier. The memory load of the A-SVM is also lower than for the other classifiers, since all samples T need not be loaded in the memory at the same time.

3.4 Experiments

We performed experiments on segmentation through voxelwise classification on data from multiple sources acquired with different MRI scanners. We evaluated two different applications of voxelwise classification: segmentation of white matter (WM) / gray matter (GM) / cerebrospinal fluid (CSF), and white-matter-lesion (WML) and multiple-sclerosis lesion (MSL) segmentation. In both cases we compared the performance of the four transfer classifiers to that of two regular supervised-learning classifiers: a regular SVM trained on all training samples, T, and an SVM trained on the same-distribution training samples, T_s only. Figure 3.1 schematically shows the usage of the different training sources in the different classifiers.

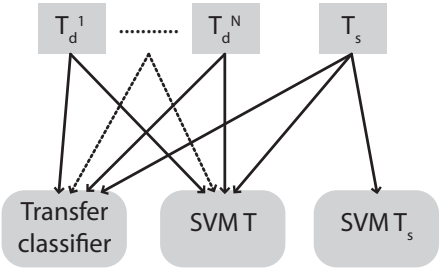


Figure 3.1: *Schematic figure of the T_d data from sources 1 to N, the T_s data, and what training data is used in the different classifiers. The Transfer classifier denotes any of the four transfer classifiers presented.*

3.4.1 Experimental Setup

Both in the WM/GM/CSF segmentations and the WML and MSL segmentations we used data from multiple sources: four for WM/GM/CSF segmentation, and three for lesion segmentation. We performed cross-validation experiments where in turn one source was selected as same-distribution source, where same-distribution training data and test data was obtained, while the data from the other sources was used as different-distribution training data.

In each experiment the performance of the four transfer classifiers was compared to the performance of the two supervised-learning classifiers. A fixed number of T_d samples was selected from the images of the different-distribution sources, while the number of T_s samples was var-

ied, to study the influence of the amount of same distribution training data. All classifiers used exactly the same test samples and where possible the same T_s and T_d training samples.

All six classifiers were based on SVM classification with a Gaussian kernel. For the regular SVM and the weighted SVMs in WSVM, RSVM and TrAdaBoost an implementation in LIBSVM [13] was used. For A-SVM we used an adaptation to the LIBSVM algorithm by the authors of the A-SVM paper[1].

For the RSVM we chose $N_{it} = 20$, which is enough to achieve convergence. For TrAdaBoost we set $N_{it} = 100$, which should be sufficient according to [23].

For each source, suitable values for the SVM parameter C and the kernel parameter γ were determined with grid search on T_d, where the best C and γ were selected according to the accuracy of a regular SVM. The same C and γ were used in all classifiers.

All four transfer classifiers have a transfer parameter that has to be tuned according to the data: for WSVM the ratio R_W, for RSVM the ratio R_R, for TrAdaBoost the initial weights $\boldsymbol{w}^1$ of the T_s samples, and for A-SVM the parameter C^s. For each of the sources this was done on the available T_d samples. Note that in all experiments T_d consisted of data from multiple sources. Each of the different-distribution sources was in turn selected as same-distribution source, where T_s training data and test data was selected, while the other different-distribution source/sources were used to extract T_d samples. In each experiment the transfer parameter optimizing the accuracy was recorded. The final parameters were obtained by averaging over the optimal parameters obtained for each of the different-distribution sources.

All images were corrected for non-uniformity using the N4 method [87], and basic image normalization was performed by a range-matching procedure that scaled the intensities such that the voxels between the 4th and the 96th percentage in intensity within the brain mask were mapped between zero and one. In each of the sources the features were normalized to zero mean and unit standard deviation.

For both applications the performance is reported in learning curves, showing the accuracy of the six classifiers as a function of the used number of T_s samples.

[1] http://www.cs.cmu.edu/ juny/AdaptSVM/

3.4.2 Brain-Tissue Segmentation Experiments

The segmentation of MRI brain images into the different tissues present (GM, WM, CSF) can give insight in the presence, severity, and location of brain atrophy. This can provide useful information about neuro-degenerative diseases such as dementia, as well as other brain disorders such as multiple sclerosis (MS) and schizophrenia. Many automated brain-tissue segmentation methods have been developed over the past 20 years, which are used in medical research as well as in the clinic.

In our experiments we performed brain-tissue segmentation by three-class voxelwise classification within a manually selected brain mask. Within this brain mask every voxel was classified as either WM, GM, or CSF.

3.4.2.1 Data Description

MR images with corresponding manual segmentations from the following four sources were used:

1. 6 T1-weighted images from the Rotterdam Scan Study [52], acquired with a 1.5T GE scanner with $0.49 \times 0.49 \times 0.80$ mm^3 voxel size
2. 12 half-Fourier acquisition single-shot turbo spin echo (HASTE) images scanned with a HASTE-Odd protocol (inversion time = 4400 ms, TR = 2800 ms, TE = 29 ms) from the Rotterdam Scan Study [52], acquired with a 1.5T Siemens scanner with $1.25 \times 1 \times 1$ mm^3 voxel size. These HASTE-Odd images have image contrast comparable to inverted T1 intensity.
3. 18 T1-weighted images from the Internet Brain Segmentation Repository (IBSR) [116], acquired with an unknown scanner, with voxel sizes between $0.84 \times 0.84 \times 1.5$ mm^3 and $1 \times 1 \times 1.5$ mm^3
4. 20 T1-weighted images from the IBSR [116], 10 acquired with a 1.5T Siemens scanner, 10 acquired with a 1.5T GE scanner, all with $1 \times 3.1 \times 1$ mm^3 voxel size

All four sources used different scanners and different scanning parameters. Figure 3.3 shows a

slice of an image from each of the four sources. The HASTE-Odd images were inverted prior to classification, because of their inverted tissue intensities compared to the T1-weighted images.

3.4.2.2 Features

To study the influence of the number of features, we performed classification on two different feature sets. The first feature set consisted of four features:

- The intensity
- The x, y, and z coordinate of the voxel, divided by the maximum width, length and height of the brain.

The second feature set consisted of 13 features – the four features mentioned above, together with nine scale-space features:

- The intensity after convolution with a Gaussian kernel with $\sigma = 1, 2$, and 3 mm^3
- The gradient magnitude of the intensity after convolution with a Gaussian kernel with $\sigma = 1, 2$, and 3 mm^3
- The absolute value of the Laplacian of the intensity after convolution with a Gaussian kernel with $\sigma = 1, 2$, and 3 mm^3.

3.4.2.3 Train and Test Sets

From the same-distribution source in turn one image was selected, where between 3 (1 for every class) and 200 T_s samples were selected randomly, while the other images in the source were used as test images. For training a total of 1 500 T_d training samples per source were selected randomly from all images of the three different-distribution sources. From each of the test images 4 000 random samples were selected, on which the accuracy was evaluated. Mean classification errors were obtained by performing multiple experiments where every image in the source was once selected as training image.

3.4.2.4 Comparison with Existing Methods

To compare the performance of our SVM classification framework with that of existing methods, complete image segmentations were obtained and compared against manual segmentations and segmentations obtained with SPM8 [4]. SPM8 is a state-of-the-art brain-tissue-segmentation tool. It performs automatic segmentation based on mixture of Gaussians with incorporation of tissue probability maps of the three tissues, that are non-linearly registered to the target image, and intensity non-uniformity estimation. The segmentation is determined with the expectation-maximization algorithm.

Evaluations were performed with the Dice coefficient [27] on the WM, GM, and CSF. The Dice coefficient is defined as

$$\text{Dice} = \frac{2\,\text{TP}}{2\,\text{TP} + \text{FP} + \text{FN}}, \tag{3.14}$$

where TP denotes the true positives, FP the false positives, and FN the false negatives.

The performance on the data from Source 4 was compared to that of several other automatic techniques as reported in literature. For this, the Tanimoto coefficient (which is also known as the Jaccard index) was used:

$$\text{TC} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}. \tag{3.15}$$

Note that $\text{TC} \leq \text{Dice}$.

3.4.2.5 Influence of Normalization

We also performed classification with two different types of image normalization in order to study the added value of the transfer classifiers over various normalization techniques. In the experiments mentioned above all images were normalized by a range-matching procedure which maps the 4th and the 96th percentage of intensity within the brain mask to zero and one. We studied the influence of two other normalization techniques. For the first method the minimum intensity within the brain mask is mapped to zero, and the maximum to one. This method should be less robust to outliers in intensity than mapping the 4th and 96th percentile.

For the second method we performed the tenth-percentile normalization procedure of Nyúl et al. [69] within the 4th and 96th percentage of intensity. This procedure first applies a range matching which maps the 4th and 96th percentile to zero and one, and next maps every tenth percentile within zero and one to the mean intensity over all (training and target) images.

Normalization experiments were performed on 13 features with the SVM T, SVM T_s, WSVM, RSVM, and A-SVM classifier. TrAdaBoost was omitted in these experiments because of its high computational load.

3.4.3 MSL and WML Segmentation Experiments

MS is a chronic inflammatory disease that affects the white matter in the brain, resulting in the formation of WMLs. Automatic methods to segment these lesions in MRI images enable the diagnosis and monitoring of the disease without the tedious task of performing manual segmentations. WMLs also occur in individuals who do not have MS. Typically, WML load increases with age, and a higher WML load is associated with cognitive decline [26], increased risk of stroke [107], and increased risk of dementia [76]. Automatic segmentation of WMLs therefore provides useful information in these research areas, as well as for the monitoring of patients.

In our experiments we performed WML and MSL segmentation by voxelwise classification. First a brain mask was determined with the brain-extraction tool [88], after which every voxel within the brain mask was classified as either lesion (WML or MSL were treated the same) or non-lesion tissue.

3.4.3.1 Data Description

We used data with manual segmentations from three different sources:

1. 20 healthy elderly subjects from the Rotterdam Scan Study [52], scanned with three sequences: T1, PD, and FLAIR, with $0.49 \times 0.49 \times 0.80$ mm^3 voxel size
2. 10 MS patients from the MS Lesion Challenge [89], scanned at the Children's Hospital of Boston with three sequences: T1, T2, and FLAIR, with $0.5 \times 0.5 \times 0.5$ mm^3 voxel size

3. 10 MS patients from the MS Lesion Challenge [89], scanned at the University of North Carolina with three sequences: T1, T2, and FLAIR, with $0.5 \times 0.5 \times 0.5$ mm^3 voxel size

Figure 3.6 shows slices of the three sequences for the three sources. As the PD images of Source 1 appear similar to the T2 images of Sources 2 and 3, we decided to treat these modalities to be the same.

3.4.3.2 Features

We performed experiments with a small and a large feature set, which were composed similarly to the feature sets for WM/GM/CSF segmentation discussed in Section 3.4.2.2, with the difference that three MRI sequences were used instead of one, and the Gaussian kernels used for the convolution had sizes $\sigma = 0.5, 1,$ and 2 mm^3. The smaller kernel sizes account for the higher resolution of the images compared to the images used in the WM/GM/CSF experiments. This resulted in a feature set of 6 features and a set of 33 features.

3.4.3.3 Train and Test Sets

Since lesion voxels appear bright on FLAIR scans, we first discarded all voxels with a low FLAIR intensity. The threshold was set to 0.75 on the normalized FLAIR image, discarding most of the CSF and some GM voxels. For the reported learning curves only voxels with a FLAIR intensity above this threshold were selected for training and testing.

For Sources 1 and 2 train and test data was obtained by randomly selecting 1% of the lesion voxels in the image and then randomly selecting non-lesion voxels above the FLAIR threshold, so that a total of 5 000 samples per image were selected. The images of Source 3 contain only few lesion voxels, since these subjects were less affected and the images were also more conservatively segmented. To still have a reasonable number of lesion samples in Source 3 4% of all lesion voxels was selected. This resulted in training and test sets with a lesion percentage of 13% for Source 1, 15% for Source 2, and 10% for Source 3.

One to eight same-distribution training images different from the test images were selected from the same-distribution source, where from each image 200 same-distribution training

samples were randomly selected in the way mentioned above. From the different-distribution sources 2 000 T_d samples were selected per source.

Mean classification errors were obtained by performing multiple experiments for differing numbers of T_s images, where every image in the same-distribution source was once used as first training image, once as second training image, etcetera. The images from the same-distribution source that were not used for training were used for testing, where the accuracy was determined on test sets of 5 000 samples per image.

3.4.3.4 Experiments for MS Lesion Challenge

We also calculated complete segmentations on 30 test images of the MS Lesion challenge, and submitted these to the challenge. Of the 30 test images 17 were acquired at the Children's Hospital of Boston (CHB, Source 2), and 13 at the University of North Carolina (UNC, Source 3). Segmentations were performed with RSVM on 33 features, which was the classifier that overall performed best in the experiments with eight same-distribution images.

In order to obtain a competing segmentation framework, the classifier was trained on more T_s samples than used in the learning curves. To speed up the calculation, only few T_d samples were used. A total of 50 000 T_s samples were selected from the ten same-distribution training images and 4 000 T_d samples were selected from the two different-distribution sources.

The classification parameters were set in a slightly different way than for the previous experiments. The SVM parameters C and γ were obtained with a grid-search experiment on the ten same-distribution images with a regular SVM. The parameter R_R was determined with a cross-validation experiment on the ten same-distribution images. In turn one same-distribution image was selected as test image, while the other nine same-distribution images were used as training data, together with the T_d samples. The value for R_R with the highest accuracy was selected.

The RSVM classifier was used to calculate a posterior probability $P(y = 1|\boldsymbol{x})$ per test voxel. The final segmentations were obtained by thresholding the posterior probability. The threshold was set differently for the two sources in the challenge data, in such a way that for the ten same-distribution training images the lesion volume in the manual and the automatic segmentation was equal.

We noticed that lesions voxels in the middle of large lesions often had lower intensities than the surrounding lesion voxels, which sometimes caused these voxels to be misclassified as non-lesion voxels. This was solved by a post-processing step, where groups of non-lesion-voxels that in the x and y direction were surrounded by lesion voxels, were classified to be lesion voxels.

The performance of our classifier on the test images of the MS Lesion Challenge was evaluated against two expert manual segmentations: segmentations from the expert who segmented the training data in Source 2, and segmentations from the expert who segmented the training data in Source 3. The segmentations were evaluated on four error metrics: relative absolute volume difference (RAVD), average symmetric surface distance (ASSD), true positive rate (TPR), and false positive rate (FPR) [89].

3.4.3.5 Influence of Normalization

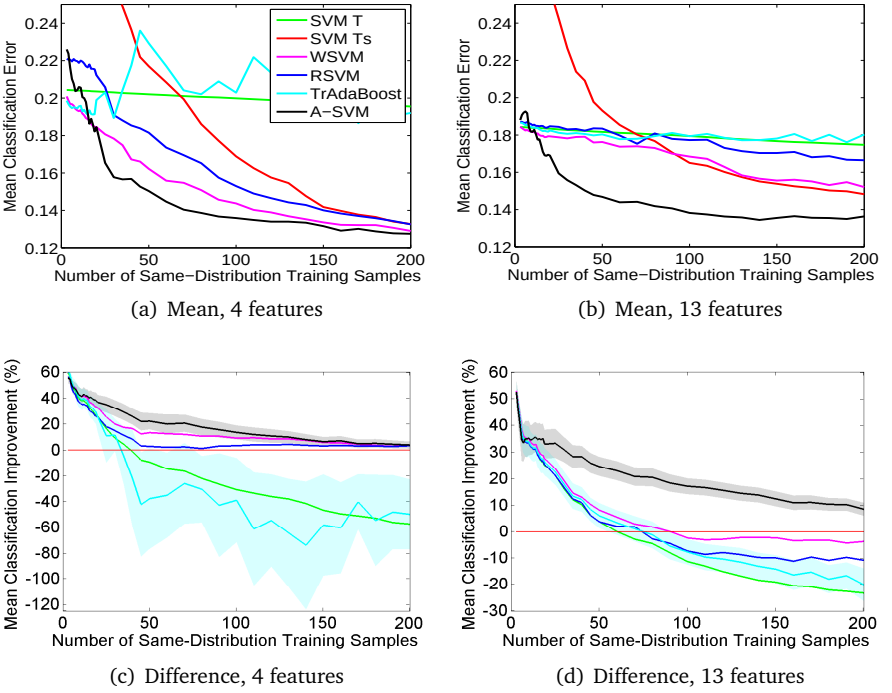
We performed experiments with two different types of normalization, similar to the experiments on GM/WM/CSF segmentation. In these experiments images from the three modalities (T1, T2/PD, and FLAIR) were all normalized with 4-96th percentile range matching, min-max range matching, and the tenth-percentile matching of Nyúl et al. [69]. These experiments were performed on the dataset with 33 features for the SVM T, SVM T_s, WSVM, RSVM, and A-SVM classifier.

3.5 Results

3.5.1 Brain-Tissue Segmentation

3.5.1.1 Comparison of Classifiers

Figures 3.2(a) and (b) give the learning curves for all classifiers on 4 and 13 features respectively. These learning curves show the mean classification errors on all 56 target images as a function of the number of same-distribution samples T_s, which were obtained from a single image. For both feature sets the SVM on all training samples T outperformed the SVM on only T_s when the number of T_s samples was small. When more T_s samples were added SVM



(a) Mean, 4 features (b) Mean, 13 features

(c) Difference, 4 features (d) Difference, 13 features

Figure 3.2: *Learning curves showing the mean classification error and classification improvement on the test sets as a function of the number of T_s samples, for the six classifiers: an SVM on T, an SVM on T_s, WSVM, RSVM, TrAdaBoost, and A-SVM. (a) and (b) show the average learning curves over all 56 images from the four sources, for 4 and 13 features respectively. (c) and (d) show the percentage reduction in mean classification error compared to the SVM T_s classifier, averaged over all 56 images, for 4 and 13 features respectively. For TrAdaBoost and A-SVM 95%-CIs for the mean improvement were included, the CIs of SVM T, WSVM, and RSVM were similar to the CI of A-SVM.*

T_s outperformed the SVM T classifier. Transfer learning improved classification compared to these two supervised-learning techniques. For SVM T classification errors were slightly lower for 13 features than for 4 features, which shows that the nine extra features hold additional information over the first four features. For the SVM T_s classifier errors were lower for four

features, because of the curse of dimensionality.

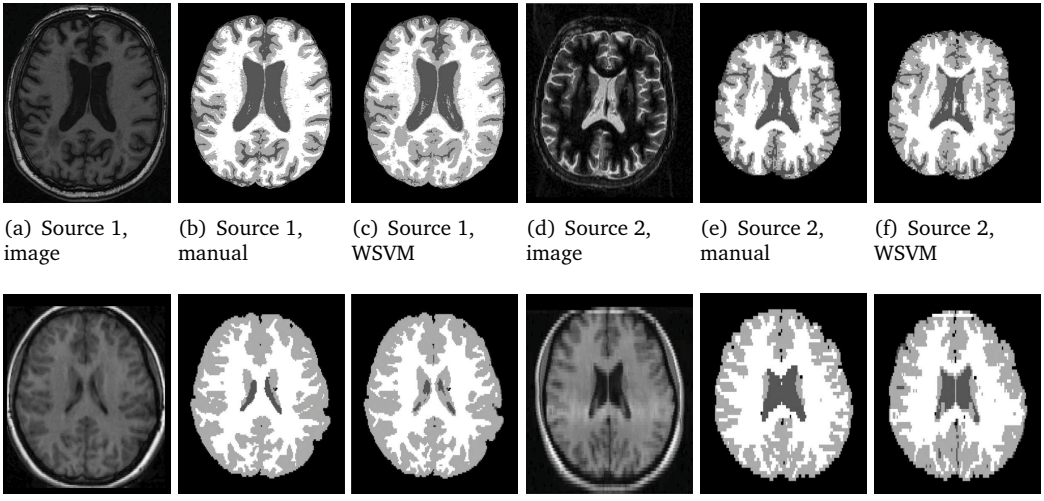
Overall, the use of transfer learning improved classification compared to the two supervised-learning techniques. Figures 3.2(c) and 3.2(d) show the percentage reduction in classification error of the different classifiers compared to the SVM T_s classifier. These two figures include 95%-confidence intervals (CIs) of the mean improvement of TrAdaBoost and A-SVM. To avoid cluttering the figure not all CIs are shown, but those of SVM T, WSVM, and RSVM were similar to the CI of A-SVM. Overall A-SVM performed best, except for fewer than 15 T_s samples, where WSVM performed best. A-SVM significantly outperformed SVM T_s for the whole range of T_s samples, WSVM significantly outperformed SVM T_s for up to 150 T_s samples for four features, and 70 T_s samples for 13 features. RSVM performed slightly worse than WSVM for both configurations, and only outperformed SVM T_s for less than 50 T_s samples on four features. TrAdaBoost performed poorly for both feature sets, and showed much higher variance than the other classifiers.

Table 3.1: *Dice coefficients for CSF, GM, and WM for complete image segmentations with the best supervised-learning classifier(SVM T_s), WSVM, RSVM, and A-SVM, and SPM8. All dices scores are given on the four sources with four features. For each experiment we used 200 T_s samples from one same-distribution training image and 4 500 T_d samples. The kNN classifier is the best classifier in [24] on the data from Source 1.*

	Source 1			Source 2			Source 3			Source 4		
Classifier	CSF	GM	WM	CSF	GM	WM	CSF	GM	WM	CSF	GM	WM
SVM T_s	0.90	0.90	0.92	0.87	0.90	0.91	0.08	0.92	0.87	0.45	0.86	0.78
WSVM	0.92	0.92	0.93	0.83	0.91	0.94	0.47	0.92	0.88	0.40	0.86	0.78
RSVM	0.91	0.92	0.93	0.91	0.93	0.94	0.43	0.92	0.87	0.37	0.84	0.77
A-SVM	0.89	0.90	0.92	0.91	0.93	0.94	0.34	0.92	0.87	0.24	0.86	0.77
SPM8	0.81	0.86	0.91	0.85	0.89	0.95	0.19	0.82	0.86	0.19	0.79	0.81
kNN [24]	0.81	0.90	0.94	-	-	-	-	-	-	-	-	-

3.5.1.2 Comparison with Existing Methods

We compared full image segmentations with existing brain-tissue-segmentation methods for the rightmost point in the learning curves (200 T_s samples). Except for A-SVM on 13 features, the transfer classifiers did not give an improvement over SVM T_s at this point of the feature curves, as can be seen from Fig. 3.2. The goal of these experiments was therefore not to investigate whether the transfer classifiers improve over other techniques, but to investigate whether our SVM T_s and transfer classifiers compare to available segmentation techniques.



(a) Source 1, image (b) Source 1, manual (c) Source 1, WSVM (d) Source 2, image (e) Source 2, manual (f) Source 2, WSVM

(g) Source 3, image (h) Source 3, manual (i) Source 3, WSVM (j) Source 4, image (k) Source 4, manual (l) Source 4, WSVM

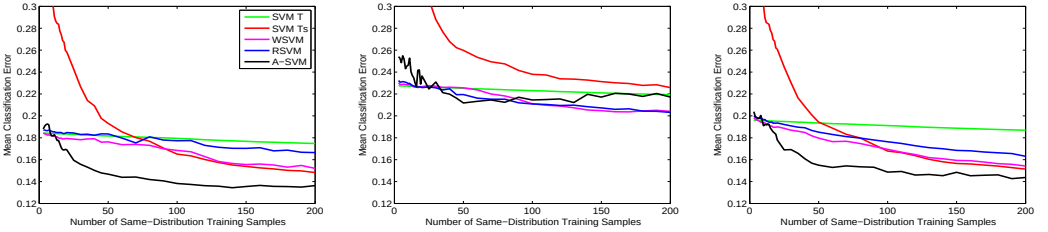
Figure 3.3: *Segmentations with the WSVM classifier with four features. The classifier was trained on a total of 4 500 T_d samples and 200 T_s samples from one image from the target source, which corresponds to the right-most point of the learning curves in Figure 3.2(a). The classification errors for the shown slices were (c) 8.1%, (f) 9.2%, (i) 6.9%, (l) 15.2%.*

Table 3.1 compares the performance of the SVM T_s classifier and three transfer classifiers: WSVM, RSVM, and A-SVM, with segmentations obtained with SPM8 [4]. For SVM T_s and the three transfer classifiers four features were used. TrAdaBoost was not included because of its poor performance and high computational load, which was caused by the large number of iterations. SVM T_s, WSVM, RSVM, and A-SVM were all significantly better than SPM8, but not significantly different from each other, based on a Friedman test with the significance threshold at $P = 0.05$. The table also includes the Dice scores of the best classifier in the brain-tissue-segmentation accuracy study of De Boer et al. [24], who used the Source 1 data to evaluate several brain-tissue-segmentation methods. In this study the best results were obtained with a kNN classifier [108]. Our classifiers obtained similar scores on WM and GM as the kNN classifier. Our classifiers also greatly outperformed the kNN classifier on CSF, but the main reason for this is that we tested within the manually segmented brain mask, whereas for the kNN classifier the brain was segmented by atlas registration. This causes additional errors, especially in the sulcal CSF.

Figure 3.3 shows examples of the resulting segmentations with the WSVM on the four sources.

We also compared our complete segmentations on Source 4, the IBSR data with 20 subjects, to various methods that reported their performance on the same data. Table 3.2 shows mean Tanimoto coefficients for CSF, GM, WM, and the sum of GM and WM, and CSF, GM, and WM. The first six entries are clustering methods, reported on the IBSR website[2], the other nine methods were collected from literature. Not all methods reported overlap values for the CSF. The best results were obtained with a decision forest classifier [119], which was trained and tested in cross validation on all remaining images. Our SVM T_s classifier and the three tested transfer classifiers WSVM, RSVM, and A-SVM, outperformed most of the other methods as well as SPM8.

3.5.1.3 Influence of Normalization



(a) range matching 4-96% (b) range matching min-max (c) Nyúl et al. [69] normalization

Figure 3.4: *Learning curves for WM/GM/CSF segmentation, showing mean classification errors as a function of the number of T_s samples on 13 features for three different normalization techniques. (a) equals the image in Fig. 3.2(b) and includes range matching between the 4th and 96th percentile, (b) includes range matching between the minimum and maximum value, and (c) includes the normalization of Nyúl et al. [69].*

Figure 3.4 shows the learning curves for the three types of normalization. Min-Max range matching, for which the results are shown in Figure 3.4(b) led to higher mean classification errors than 4-96th percentile range matching. Also, for min-max range matching the SVM T_s classifier performed worse than the SVM T classifier regardless of the number of T_s samples, indicating that the min-max normalization is not sufficient, even within the same source. Applying the more extensive normalization of Nyúl et al. [69], for which the result is shown in

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

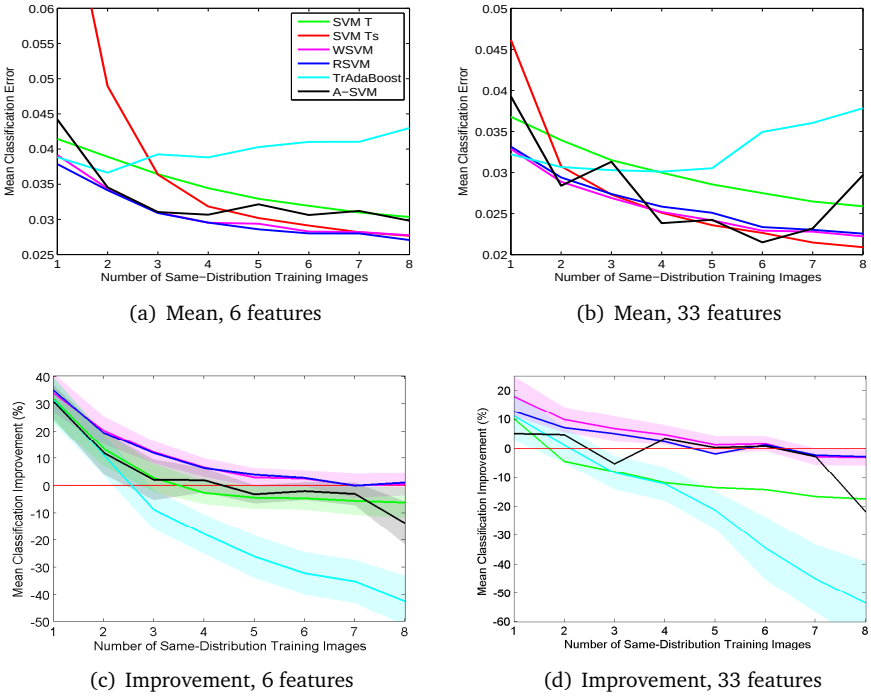
Table 3.2: *Mean Tanimoto coefficients on CSF, GM, and WM for a variety of methods on the IBSR data with 20 subjects. G+W denotes the average score on GM and WM, and C+G+W denotes average score on CSF, GM, and WM. For the SVM T_s, WSVM, RSVM, and A-SVM classifier 200 T_s samples were randomly selected from one image, for the transfer classifiers $4\,500$ T_d samples were added. Classification was performed on four features.*

Classifier	CSF	GM	WM	G+W	C+G+W
Adaptive MAP [77]	0.069	0.564	0.567	0.566	0.400
Biased MAP [77]	0.071	0.558	0.562	0.560	0.379
Fuzzy c-means [77]	0.048	0.473	0.567	0.519	0.362
MAP [77]	0.071	0.550	0.554	0.552	0.392
ML [77]	0.062	0.535	0.551	0.543	0.383
TS k-means [77]	0.049	0.477	0.571	0.524	0.366
AMS [63]	-	0.683	0.691	0.687	-
BSE-BFC-PVC [84]	-	0.595	0.664	0.630	-
C-GMM [41]	-	0.680	0.660	0.670	-
Decision Forest [119]	0.614	0.838	0.731	0.785	0.728
FC-PABIC [123]	-	0.770	0.658	0.714	-
Modified FCM [86]	-	0.750	0.724	0.737	-
MPM-MAP [62]	0.227	0.662	0.683	0.673	0.524
SV-GMM [73]	-	0.768	0.734	0.751	-
TMCD [93]	-	0.676	0.669	0.673	-
SPM8	0.107	0.650	0.684	0.667	0.480
SVM T_s	0.309	0.757	0.645	0.701	0.570
WSVM	0.266	0.754	0.648	0.701	0.556
RSVM	0.240	0.730	0.633	0.682	0.534
A-SVM	0.162	0.759	0.633	0.696	0.518

Figure 3.4(c), did not give better overall results than when 4-96th percentile range matching was applied. The performance of the SVM T_s and the transfer classifiers was similar for the two normalization techniques, but the SVM T classifier performed slightly better for the 4-96th percentile range matching. This indicates that more extensive normalization is not needed to normalize within sources, and slightly hurts the performance of classification between sources. The use of a transfer classifier improved classification of the two supervised-learning classifiers regardless of the used normalization technique.

3.5.2 MSL and WML Segmentation

3.5.2.1 Comparison of Classifiers



(a) Mean, 6 features (b) Mean, 33 features

(c) Improvement, 6 features (d) Improvement, 33 features

Figure 3.5: *Learning curves showing the mean classification error and mean classification improvement on the test sets as a function of the number of T_S images, for WML segmentation with the six classifiers: SVM on T, SVM on T_S, WSVM, RSVM, TrAdaBoost, and A-SVM. (a) and (b) show the mean learning curves over all 40 images from the three sources for 6 and 33 features respectively. (c) and (d) show the percentage reduction in mean classification error compared to SVM T_S averaged over all 40 images, for 6 and 33 features respectively. The shaded areas show 95%-CIs for the mean improvement. For (a) and (b) the CI of SVM T and RSVM were similar to the one of WSVM, and for (b) the CI of A-SVM was similar to the one of WSVM.*

We performed a similar set of cross-validation experiments for MSL and WML segmentation.

Figures 3.5(a) and (b) show the mean learning curves of the six classifiers on 6 and 33 features respectively. A very similar pattern can be seen as in the learning curves for GM/WM/CSF segmentation: for a small amount of T_s data SVM T was the best supervised-learning classifier, whereas for more T_s data SVM T_s performed better. The transfer classifiers WSVM and RSVM improved over these two supervised-learning classifiers up to a point where a relatively large number of same-distribution training images was available. At this point SVM T_s, WSVM and RSVM converged to the same error rate. All classifiers performed better on 33 features than on 6. Figures 3.5(c) and (d) show the improvement over SVM T_s for SVM T, WSVM, RSVM, TrAdaBoost and A-SVM, for 6 and 33 features. The figures include CIs for some of the classifiers. The CIs of the other classifiers were similar to that of WSVM. WSVM and RSVM overall performed best, significantly outperforming SVM T_s for up to five T_s images (three for RSVM on 33 features). WSVM seems to perform slightly worse than RSVM on 33 features, but this difference is not significant. Similar to the GM/WM/CSF experiments TrAdaBoost overall performed poorly, with a larger variance than the other classifiers. A-SVM, which overall performed best on the WM/GM/CSF experiments, did not perform well in the lesion-segmentation experiments.

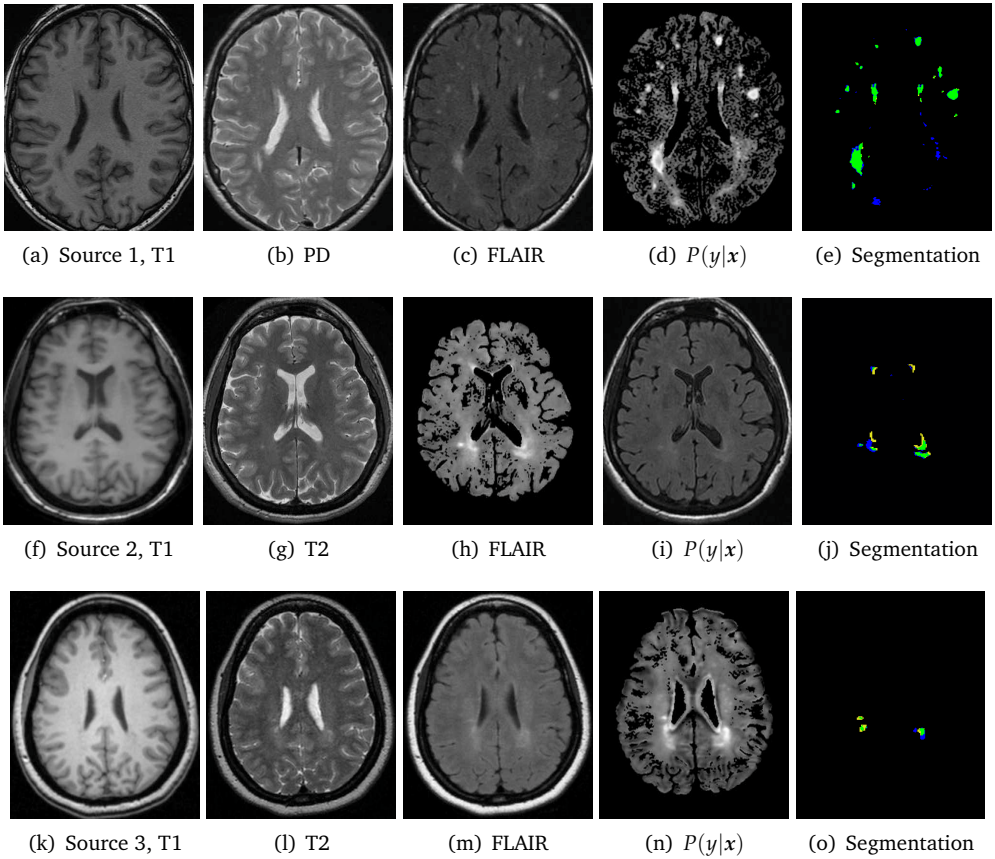
Figure 3.6 shows resulting segmentations of the RSVM classifier on 33 features with eight same-distribution images, where the threshold on the posterior probabilities was selected so that the total lesion volume equaled that in the manual segmentation.

3.5.2.2 MS Lesion Challenge

Table 3.3: *Average scores obtained on the two datasets (CHB = Children's Hospital of Boston, UNC = University of North Carolina) of the MS Lesion Challenge, for RSVM with 33 features. RAVD = Relative Absolute Volume Difference (%), ASSD = Average Symmetrical Surface Distance (mm), TPR = True Positive Rate (%), FPR = False Positive Rate (%). The score for each measure is between brackets.*

	Ground Truth: CHB Rater				Ground Truth: UNC Rater				
Dataset	RAVD	ASSD	TPR	FPR	RAVD	ASSD	TPR	FPR	Total
All CHB	112(84)	7(85)	53(81)	79(62)	115(89)	5(91)	59(84)	70(67)	80
All UNC	151(84)	12(74)	28(68)	66(70)	300(87)	13(73)	45(76)	70(67)	75
All Average	129(84)	10(80)	42(75)	73(65)	195(88)	8(83)	53(81)	70(67)	78

Table 3.3 shows the mean scores obtained on the 30 test images of the MS Lesion Challenge data. The scores were designed such that a score of 90 is comparable to expert segmentations.



(a) Source 1, T1 (b) PD (c) FLAIR (d) $P(y|\boldsymbol{x})$ (e) Segmentation

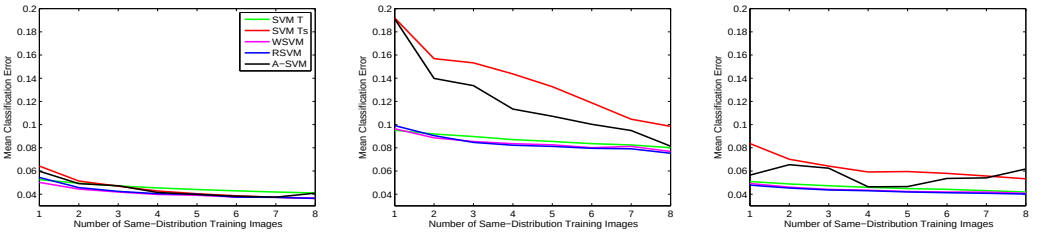
(f) Source 2, T1 (g) T2 (h) FLAIR (i) $P(y|\boldsymbol{x})$ (j) Segmentation

(k) Source 3, T1 (l) T2 (m) FLAIR (n) $P(y|\boldsymbol{x})$ (o) Segmentation

Figure 3.6: *Examples of resulting segmentations of the RSVM classifier on 33 features after training on T_s samples from eight images and 4 000 T_d samples. Figures (d),(i),(n) show the posterior outputs of the classifier, and Figures (e),(j),(o) show the final segmentation in blue, the manual segmentation in yellow, and the overlap between the two in green. The true positive rates (TPRs) and false positive rates (FPRs) for the showed slices were (e): TPR=92%, FPR=14%, (j): TPR=47%, FPR=49%, (o): TPR=48%, FPR=45%.*

Our method performed slightly better on the CHB data than on the UNC data, with scores of 80 and 75 respectively. At the moment of writing the website of the MS Lesion Challenge[3] listed the performance of 35 segmentation algorithms. With a total score of 77.9083 our method ranked second on a total of eight methods that segmented all 30 test images. The other 27 methods segmented only 23 test images (14 CHB, 9 UNC), on which our algorithm obtained a total score of 81.2174. Nine of the 27 methods had a higher score on the 23 test images than our algorithm.

3.5.2.3 Influence of Normalization



(a) range matching 4-96% (b) range matching min-max (c) Nyúl et al. [69] normalization

Figure 3.7: *Learning curves for WML/MSL segmentation, showing mean classification error as a function of the number of T_S samples on 13 features for three different normalization techniques. (a) equals the image in Fig. 3.5(b) and includes range matching between the 4th and 96th percentile, (b) includes range matching between the minimum and maximum value, and (c) includes the normalization of Nyúl et al. [69].*

Figure 3.7 shows the learning curves for the three types of normalization. Like for WM/GM/CSF segmentation, the Min-Max range matching, shown in Figure 3.4(b), led to higher mean classification errors than 4-96th percentile range matching. The more extensive normalization of Nyúl et al. [69], for which the result is shown in Figure 3.7(c), gave similar results as 4-96th percentile range matching for SVM T, WSVM, and RSVM, but not for SVM T_S and A-SVM. For all three normalization techniques a WSVM or RSVM classifier improved performance. Remarkably, the performance of the SVM T_S classifier deteriorates when the normalization of Nyúl et al. [69] is used, compared to 4-96th percentile range matching.

[3] http://www.ia.unc.edu/MSseg/results_table.php

3.6 Conclusion and Discussion

We presented a transfer-learning approach to image segmentation, which enables supervised segmentation of images acquired with different MRI scanners and/or imaging protocols. The presented transfer classifiers benefit from training data acquired with different protocols, so-called different-distribution training data (T_d), and therefore compared to a regular supervised classifier, need fewer labeled samples that are exactly representative of the target data, the so-called same-distribution training data (T_s), to obtain the same result.

The benefits of transfer learning over standard supervised learning were evaluated with experiments on WM/GM/CSF segmentation and WML and MSL segmentation on MRI brain scans obtained with various scanners and scan protocols, with varying numbers of T_s samples. The experiments showed a clear improvement in performance when transfer learning was used. Specifically, for a small number of T_s samples transfer learning greatly outperformed the supervised-learning classifiers, minimizing mean classification errors by up to 60%. Also, when the required accuracy is set, the use of a transfer classifier typically reduces the required number of T_s training samples. Ultimately, when enough T_s samples were available to reliably train a supervised classification scheme, a regular SVM on T_s and the best-performing transfer classifiers reached similar performance.

For GM/WM/CSF segmentation, a relatively easy task, a regular SVM on T_s reached the same performance as the best transfer classifiers at an earlier point than was the case for the lesion segmentation. Also, intuitively transfer learning could bring more improvement when more features are used, since higher-dimensional feature spaces generally require more training samples. This could clearly be seen in the experiments on WML/MSL segmentation. On the experiments on brain-tissue classification however, only one of the transfer classifiers gave more improvement on the larger feature set.

We presented and evaluated four transfer classifiers: Weighted SVM (WSVM), Reweighted SVM (RSVM), TrAdaBoost, and Adaptive SVM (A-SVM). WSVM showed to be the most consistent classifier of the four; for a small number of T_s samples, it outperformed the regular SVMs on all training data T and on only T_s in all learning curves. RSVM generally performed similar to the WSVM on the lesion segmentation experiments, but worse than the WSVM on the WM/GM/CSF segmentation experiments. TrAdaBoost showed the worst results, never outperforming the two baseline supervised-learning classifiers. It especially performed badly for

lesion segmentation, where classification errors increased for an increasing number of T_s samples. We think TrAdaBoost is likely to experience difficulties when there is class overlap in the T_s samples. Since the weights of misclassified T_s samples are increased, this can make the classifier focus too much on a few initially misclassified T_s samples. The performance of A-SVM was dependent on the classification task. It performed very well on the WM/GM/CSF segmentation experiments when more than 15 T_s samples were available, in most cases greatly outperforming all other classifiers, whereas it did not give an overall good performance on the lesion-segmentation experiments. A-SVM is the only classifier of the four that does not explicitly take the T_d samples into account. It therefore incorporates less knowledge of the distribution of the T_d samples, such as the amount of class overlap and the class prior probabilities, which might be disadvantageous in some cases. Further investigating which of the different transfer classifiers are best suitable for which situation might be an interesting direction for future work.

We also investigated the influence of three image normalization techniques. In our experiments min-max normalization led to larger classification errors than the 4th-96th percentile range matching, both between and within sources. The more extensive normalization method of Nyúl et al. [69] overall slightly increased classification errors for the WM/GM/CSF segmentation experiments, and slightly decreased errors for WML and MSL segmentation. This is in contrast to results by Shah et al. [83], who showed that this normalization can greatly improve performance on images from different sources. For all normalization techniques a transfer-learning classifier could still bring improvement over the regular classifiers, indicating that although a more advanced normalization procedure could reduce differences between images from different scanners, transfer learning is still beneficial.

In the experiments the SVM parameters C and γ were determined with a regular SVM on T_s, which gave the regular SVMs an advantage over the transfer classifiers. The performance of the transfer classifiers may therefore still be improved by determining the optimal C and γ for each classifier separately, for instance by grid search on the different-distribution sources. To facilitate the large number of experiments required for the learning curves however, we chose to keep these parameters fixed. The classifier-specific parameters of the transfer classifiers were tuned using cross validation on the different-distribution sources, assuming that the differences and similarities between those sources were representative of the differences and similarities that can be expected in general between T_s and T_d data. Another option would be to include T_s data when determining the transfer parameters. As we wanted to study the behavior of the transfer classifiers also for very few T_s samples, this technique was only used

in the experiments for the MS-lesion challenge. It could also be beneficial to apply different transfer parameters for each of the different-distribution sources, depending on the similarity with the target data. Exploring other ways of determining classifier parameters will be a topic of further research.

The three transfer classifiers WSVM, RSVM, and A-SVM provided good segmentations in comparison to results reported in literature. In Table 3.1 we compared our WM/GM/CSF segmentations to segmentations obtained with SPM8 [4], a state-of-the-art brain-tissue segmentation tool. On all four sources WSVM, RSVM, and A-SVM outperformed SPM8. In Table 3.2 we reported the performance of various methods on the IBSR data with 20 subjects. Our transfer classifiers outperformed 12 of the 16 methods. One of the methods that outperformed our classifiers was trained and tested in cross validation, using many more same-distribution training images than our methods, and the other three used a much more sophisticated bias-correction scheme. Using our methods as part of such a scheme could increase the performance on this dataset. Also, in the MRBrainS13[4] brain-tissue-segmentation challenge our SVM classification scheme ranked second, only to be beaten by a semi-automatic method. In the MS-lesion challenge our RSVM ranked second out of nine methods on all test data, and tenth out of 26 methods on a subset.

For MRI brain-tissue segmentation several other techniques have been developed to facilitate image segmentation across scanners. Cocosco et al. [18] used a registered probabilistic tissue atlas to automatically select "training" samples from target images, based on which a kNN classifier was trained to segment the whole image. Freesurfer [31] first automatically segments the voxels with the highest intensities (within a brain mask) as WM, after which the GM is identified by dilation of the WM tissue following the intensity gradients up until the point where a decrease in intensity indicates the boundary between GM and CSF. A different often used approach is unsupervised classification by the expectation-maximization (EM) algorithm [62, 98, 114, 115]. Here segmentation of the target data is performed by alternating between optimization of the source-specific model parameters given the segmentation of the previous step, and optimizing the segmentation given the determined model parameters. The state-of-the-art brain-tissue-segmentation method SPM is also based on such an EM-optimization [4]. All these methods do not use any labeled samples of the target data. This makes it easy to apply these techniques to new data. However, as our experiment prove superiority of transfer learning over SPM, we may conclude that a small amount of manually labeled T_s data used in a transfer-learning framework, can greatly improve the performance.

[4]http://mrbrains13.isi.uu.nl/

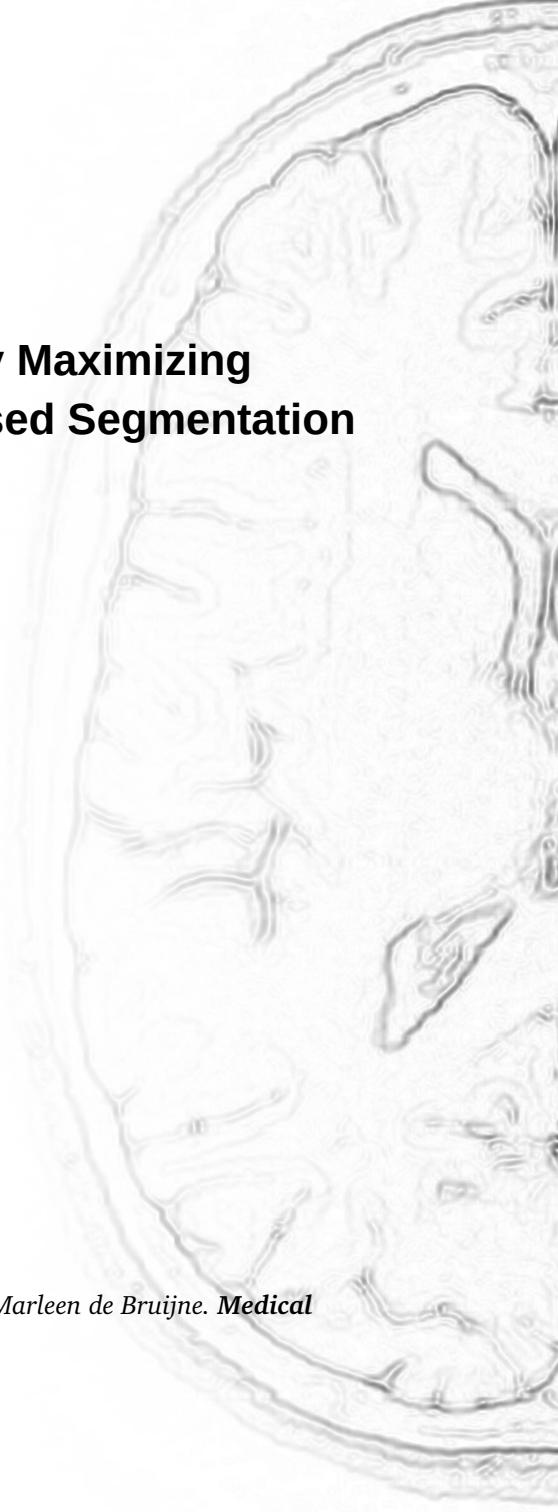
Many of the techniques mentioned above combine voxelwise classification with atlas-based prior tissue probabilities, partial-volume modeling, and/or Markov-Random-Field modeling. In this work we have restricted ourselves to voxelwise classification, to allow for a direct comparison of the different learning techniques. However, the established transfer-learning framework could also be used as the basis of a more advanced segmentation scheme, replacing the voxelwise classification step in any of the mentioned techniques.

In the experiments we have focused on MRI brain segmentation. However, the variability in imaging protocols forms a common problem across most applications. We expect that transfer learning can also improve supervised algorithms in many other segmentation and image analysis tasks.

We believe that transfer learning is a promising approach to biomedical image analysis. In applications for which data with ground truth labels is available from other studies, transfer learning can significantly decrease the amount of representative training data needed. This facilitates the application of supervised techniques in large multi-center studies and in clinical practice.

Chapter 4

Weighting Training Images by Maximizing Distribution Similarity for Supervised Segmentation Across Scanners



Annegreet van Opbroek, Meike W. Vernooij, M. Arfan Ikram, & Marleen de Bruijne. ***Medical Image Analysis, 2015, 24(1)****, 245-254.*

Many automatic segmentation methods are based on supervised machine learning. Such methods have proven to perform well, on the condition that they are trained on a sufficiently large manually labeled training set that is representative of the images to segment. However, due to differences between scanners, scanning parameters, and patients such a training set may be difficult to obtain.
We present a transfer-learning approach to segmentation by multi-feature voxelwise classification. The presented method can be trained using a heterogeneous set of training images that may be obtained with different scanners than the target image. In our approach each training image is given a weight based on the distribution of its voxels in the feature space. These image weights are chosen as to minimize the difference between the weighted probability density function (PDF) of the voxels of the training images and the PDF of the voxels of the target image. The voxels and weights of the training images are then used to train a weighted classifier.
We tested our method on three segmentation tasks: brain-tissue segmentation, skull stripping, and white-matter-lesion segmentation. For all three applications, the proposed weighted classifier significantly outperformed an unweighted classifier on all training images, reducing classification errors by up to 42%. For brain-tissue segmentation and skull stripping our method even significantly outperformed the traditional approach of training on representative training images from the same study as the target image.

4.1 Introduction

The segmentation of medical images into tissues or structures yields quantitative information that can be used to study the cause and development of a disease and to facilitate the diagnosis. Since performing such segmentations manually is very time consuming and prone to inter- and intra-observer variability, a large variety of algorithms have been developed to perform automated segmentation. Many methods for automated image segmentation are based on supervised learning [2, 33, 36, 61, 96, 103], where a segmentation model is trained on a manually annotated set of training images. To train a successful model, these supervised-learning

techniques require a training set that is both sufficiently large to capture a large variation of appearances and representative of the target data. In practice however, available training sets may not be exactly representative of the target data, since they may be obtained with a different scanner, a different imaging protocol, or may regard a different study population.

Contrary to conventional supervised-learning techniques, so-called *transfer-learning* techniques are designed to cope with training data that is not exactly representative of the target data [71]. The use of transfer-learning techniques in medical imaging is relatively new. Cheng et al. [14, 15] used transfer learning to predict which patients with mild cognitive impairment (MCI) would convert to Alzheimer's disease (AD), based on MRI brain scans. By using a transfer classifier they could improve the performance of their algorithm by training not only on MCI patients, but also on AD patients and healthy controls (CN). Guerrero et al. [43] performed AD/MCI/CN classification on 1.5T and 3T brain MRI scans. By using manifold alignment they were able to combine 1.5T and 3T scans into a single feature space by finding a low-dimensional manifold that generalizes between the two field strengths.

In medical image segmentation, several transfer-learning techniques have been proposed, which can be trained on only a small amount of training data for the target study and a larger amount of other training data. Ablavsky et al. [1] performed mitochondria segmentation in microscopy images, where they trained on a large number of images of striatum tissue and a smaller number of images of hippocampus tissue, and tested on images of hippocampus tissue. They demonstrated the superior performance of a transfer classifier over a non-transfer classifier on this dataset. Becker et al. [7] used transfer learning for mitochondria and synapse segmentation and fiber-path classification in microscopy images. Their method was trained on images from different domains (e.g. striatum tissue and hippocampus tissue) and would learn a non-linear feature mapping that would generalize between all training and target images. In previous work [100, 102] we applied transfer learning to brain-tissue and white-matter-lesion segmentation by voxelwise classification in MR brain images that originated from different studies. We compared the performance of four transfer classifiers (three that used data weighting and one other transfer classifier) with that of two non-transfer classifiers. We showed that when only a small amount of training data from the target study was available a transfer classifier could greatly improve performance over a non-transfer classifier. Van Engelen et al. [97] showed that a combination of image normalization and a transfer classifier that used data weighting can be beneficial for the segmentation of plaque components in MR images of the carotid artery.

In the transfer-learning literature there are two main categories of problems where transfer learning is used. The first and broadest category consists of problems where training and target data have different labeling functions $P(y|\boldsymbol{x})$ and might, on top of that, have different prior distributions $P(\boldsymbol{x})$, features, and even classes. Most applications fall in this category, since labeling functions often differ between training and target data. For example, if we would have training and target images that were obtained with different scanning protocols, the distribution of intensity values for a tissue type under consideration is very likely to differ between training and target images. If these intensity values are used as features, this would result in a difference in labeling functions between training and target data. In medical image segmentation, all algorithms that are proposed for problems in this category assume that a small amount of labeled target data is available on top of a much larger amount of other labeled training data. The other labeled training data is then used either in a regularization term in a classifier [1, 7, 102] or given a lower weight than the labeled target data and used in a classifier with a weighted loss function [97, 102]. Van Opbroek et al. [102] and Van Engelen et al. [97]) also presented a method that iteratively calculated a weighted classifier on all training data and reduced the weights of the misclassified labeled target samples. The rationale behind this reweighting is to give high weights only to target samples that are useful for the classification and lower weight to target samples that might be harmful.

The second category of problems is often called *covariate shift*[1] [71]. In cases with covariate shift there is a difference in covariates between training and target data. These covariates are assumed to correlate with the class label y, but not with the features. As a result, training and target samples can be assumed to have the same labeling function $P(y|\boldsymbol{x})$ (and therefore also have the same features and classes) but do not follow the same underlying distribution $P(\boldsymbol{x})$. Such cases often appear when training and test data are acquired under different circumstances, e.g. in object recognition if test data consists of objects under different poses, while training data consists of objects at frontal pose [72]. The resulting difference in underlying distributions $P(\boldsymbol{x})$ between training and target samples can be corrected for by weighting the training samples in such a way that the probability density function (PDF) of the target samples and the PDF of the weighted training samples becomes more similar. Various methods have been proposed to determine these weights [30, 46, 90, 91, 120]. Heimann et al. [45] for example, used a weighting method of Sugiyama et al. [91] to improve localization of an ultrasound transducer in X-ray images. Their method was trained on artificial training images of the transducer and applied to real-life images, which introduced a distribution difference

[1] Transfer learning for covariate shift is often referred to as *domain adaptation*. The term can however be confusing, since it is also sometimes used as an analogue for the term transfer learning.

between training and target images, yet, due to the way their training data was constructed, they could assume to have no difference in labeling functions between training and test data. By weighting the training images according to distribution similarity with unlabeled real-life images they could assign training weights that reflected the probability of the training images being observed in real life. Goetz et al. [39] used the same weighting method for brain-tumor segmentation. Their classifier was trained on some manually indicated regions in a training image. Since these regions were all drawn in unambiguous parts of the tissues (i.e. far away from the tissue edges), which resulted in a distribution difference between training and target data since the regions were all drawn in unambiguous regions of the image.

In this paper we investigated whether weighting based on differences between training and target PDFs can also be beneficial in cases where training and target labeling functions are different. Our method assigns a weight to every training image (rather than to every voxel, as done in cases with covariate shift) based on the distribution of its voxels in a feature space. This is done in such a way that the PDF of the weighted training voxels in the feature space best approximates the PDF of the voxels of a target image. This way, training images with a PDF that is more similar to that of the target image are more likely to receive a high weight than images with a very different PDF. By jointly weighting all training images, weights can be distributed among images that match different parts of the target PDF. After determining the image weights, a weighted classifier is calculated on the training voxels and the weights given to their corresponding image. We evaluated our method on training images from different studies as well as on images from the same study as the target image.

An early version of this work was presented in a workshop paper [101]. The presented paper extends the work in [101] by comparing the performance of three PDF dissimilarity measures and presenting more thorough experiments on three applications.

This paper is organized as follows: the proposed methods for image weighting and classification are presented in Section 4.2, experiments on brain-tissue segmentation, white-matter-lesion segmentation and skull stripping are presented in Section 4.3, of which the results are presented in Section 4.4. The conclusion and discussion is given in Section 4.5.

4.2 Methods

Our method consists of two parts. First, for every training image a weight is determined based on the PDF of its voxels in the feature space. These weights are chosen in an unsupervised manner, by minimizing the difference between the PDFs of the weighted training images and the PDF of the target image. In this work, we experimented with three PDF dissimilarity measures, which are presented in Sections 4.2.2.1, 4.2.2.2, and 4.2.2.3. Second, we train a weighted classifier on the samples (voxels) of the training images, where every sample is given the weight of the image it originates from. The resulting classifier is then used to classify the target voxels. As a weighted classifier we used a weighted version of the support vector machine (SVM), which is presented in Section 4.2.3. Figure 4.1 gives a schematic overview of the algorithm.

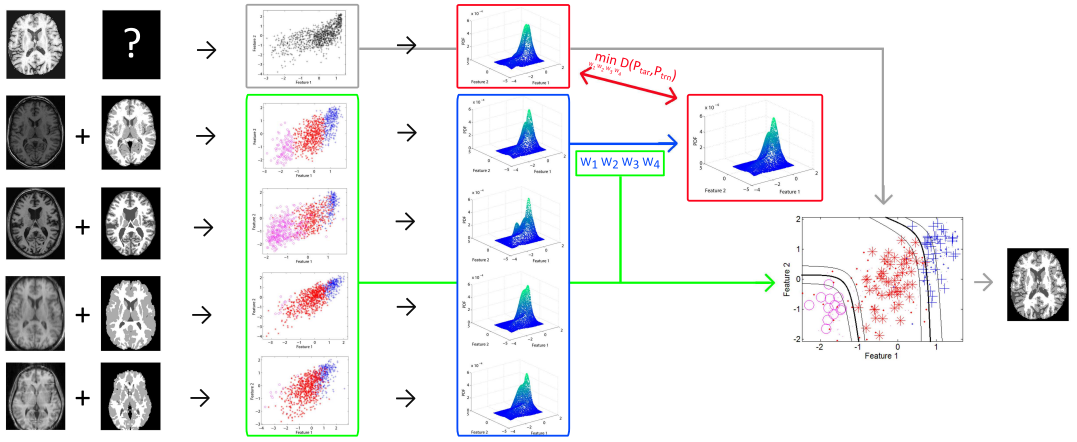


Figure 4.1: *Schematic overview of our method. From all training images and the target image the PDF of the samples in the feature space is calculated (when the KL is used the PDF of the target samples need not be calculated). Next, for each training image a weight is determined by minimizing the difference between the target PDF and the total (weighted) training PDF. The training samples and their corresponding image weights are then used in a weighted SVM classifier.*

4.2.1 Notation

Let $\boldsymbol{x}_i \in \mathbb{R}^n$ denote the n-dimensional feature vector for sample (voxel) number i. We distinguish between a target sample $\boldsymbol{x}_i^{\text{tar}}$ ($i = 1, 2, \ldots N_{\text{tar}}$) with corresponding (unknown) label y_i^{tar}, and a training sample $\boldsymbol{x}_j^{\text{trn}}$ ($j = 1, 2, \ldots N_{\text{trn}}$), with corresponding label y_j^{trn}. We will here describe the two-class case, where $y_i^{\text{tar}}, y_j^{\text{trn}} \in \{-1, 1\}$. For multi-class problems this was extended by one-vs-one classification.

Let $P_m(\boldsymbol{x})$ denote the PDF of training image m at location $\boldsymbol{x}$ in the feature space, $P_{\text{trn}}(\boldsymbol{x})$ the total training PDF of all weighted training samples together, and $P_{\text{tar}}(\boldsymbol{x})$ the PDF of the target image. In our experiments all PDFs were estimated by kernel density estimation as described in Section 4.3.5.

We propose to assign an importance weight W_m ($m = 1, 2, \ldots, M$) to every training image m, which represents the importance of the image in the training phase. The weights are non-negative and normalized such that $\sum_{m=1}^{M} W_m = 1$. This gives a total weight vector of $\boldsymbol{W} = [W_1, W_2, \ldots, W_M]$.

4.2.2 Determining the Image Weights

The total training PDF $P_{\text{trn}}(\boldsymbol{x})$, is a weighted sum of each of the M PDFs $P_m(\boldsymbol{x})$:

$$P_{\text{trn}}(\boldsymbol{x}) = \sum_{m=1}^{M} W_m P_m(\boldsymbol{x}). \tag{4.1}$$

The values for W_m can be chosen such that the difference between P_{trn} and P_{tar} is minimized. Since many measures exist to determine the difference between two PDFs [12], we experimented with three measures, each of which belongs to a different "family" of distances: the Kullback-Leibler divergence, the Bhattacharyya distance, and the squared Euclidean distance. We will briefly discuss each of the three measures and how these can be used to determine the weights.

4.2.2.1 Kullback-Leibler Divergence

One of the most common measures for the difference between two PDFs is the Kullback-Leibler divergence (KL). The KL between $P_{\text{tar}}(\boldsymbol{x})$ and $P_{\text{trn}}(\boldsymbol{x})$ is defined as

$$\text{KL}(P_{\text{tar}}||P_{\text{trn}}) = \int_{\mathcal{D}} P_{\text{tar}}(\boldsymbol{x}) \log\Big(\frac{P_{\text{tar}}(\boldsymbol{x})}{P_{\text{trn}}(\boldsymbol{x})}\Big) d\boldsymbol{x}, \tag{4.2}$$

where $\mathcal{D}$ is the domain of P_{tar} and P_{trn}. This expression can be rewritten by using the target samples to approximate $\int_{\mathcal{D}} P_{\text{tar}}(\boldsymbol{x})d\boldsymbol{x}$ by $\frac{1}{N_{\text{tar}}}\sum_{i=1}^{N_{\text{tar}}} \boldsymbol{x}_i^{\text{tar}}$. This leads to the following expression:

$$\text{KL}(P_{\text{tar}}||P_{\text{trn}}) \approx \frac{1}{N_{\text{tar}}} \sum_{i=1}^{N_{\text{tar}}} \log\Big(\frac{P_{\text{tar}}(\boldsymbol{x}_i^{\text{tar}})}{P_{\text{trn}}(\boldsymbol{x}_i^{\text{tar}})}\Big). \tag{4.3}$$

If a large number of randomly drawn target samples is used, this approximation should be fairly accurate. This expression equals

$$\begin{aligned} \text{KL}(P_{\text{tar}}||P_{\text{trn}}) \approx & \frac{1}{N_{\text{tar}}} \sum_{i=1}^{N_{\text{tar}}} \log P_{\text{tar}}(\boldsymbol{x}_i^{\text{tar}}) \\ & - \frac{1}{N_{\text{tar}}} \sum_{i=1}^{N_{\text{tar}}} \log\Big(\sum_{m=1}^{M} W_m P_m(\boldsymbol{x}_i^{\text{tar}})\Big), \end{aligned} \tag{4.4}$$

which follows from substituting P_{trn} as given in Eq. 4.1.

Note that the first term of Eq. 4.4 is independent of W_m, therefore the KL is minimized by maximizing the second term with respect to W_m.

4.2.2.2 Bhattacharyya Distance

The Bhattacharyya distance (BD) is another commonly used measure for distances between PDFs, which minimizes the sum of geometric means of two PDFs. The KL is large if P_{tar} and

P_{trn} differ at locations where P_{trn} is small, which makes it sensitive to its tails. Contrary to the KL, the BD gives a larger difference value if P_{tar} and P_{trn} differ at locations where either P_{trn} or P_{tar} is large. The BD between $P_{\text{tar}}(\boldsymbol{x})$ and $P_{\text{trn}}(\boldsymbol{x})$ is defined as

$$\text{BD}(P_{\text{tar}}, P_{\text{trn}}) = -\log \int_{\mathcal{D}} \sqrt{P_{\text{tar}}(\boldsymbol{x}) P_{\text{trn}}(\boldsymbol{x})} d\boldsymbol{x}. \tag{4.5}$$

The integral can be approximated by a large number N_r of random samples $\boldsymbol{x}_i$ from the domain $\mathcal{D}$:

$$\text{BD}(P_{\text{tar}}, P_{\text{trn}}) \approx -\log \frac{1}{N_r} \sum_{\boldsymbol{x}_i \in \mathcal{D}} \sqrt{P_{\text{tar}}(\boldsymbol{x}_i) P_{\text{trn}}(\boldsymbol{x}_i)}. \tag{4.6}$$

By substituting P_{trn} as given in Eq. 4.1 this equals

$$\text{BD}(P_{\text{tar}}, P_{\text{trn}}) \approx -\log \frac{1}{N_r} \sum_{\boldsymbol{x}_i \in \mathcal{D}} \sqrt{P_{\text{tar}}(\boldsymbol{x}_i) \sum_{m=1}^{M} W_m P_m(\boldsymbol{x}_i)}. \tag{4.7}$$

Note that to minimize the BD one needs not only the training PDFs P_m, but also the target PDF P_{tar}, whereas to minimize the KL one needs only the training PDFs. This makes the computation time slightly larger if the BD is used than if the KL is used.

4.2.2.3 Squared Euclidean Distance

Third, we investigated the Squared Euclidean distance (SE) or L_2 distance. Unlike the KL and the BD, which measure relative distances, the SE measures the absolute distance between the training and the target PDFs. This makes that the SE, unlike the KL and the BD, gives equal importance to all positions in the domain. The SE between P_{tar} and P_{trn} is defined as

$$\text{SE}(P_{\text{tar}}, P_{\text{trn}}) = \int_{\mathcal{D}} (P_{\text{tar}}(\boldsymbol{x}) - P_{\text{trn}}(\boldsymbol{x}))^2 d\boldsymbol{x}. \tag{4.8}$$

Similar to the BD, this can be approximated by

$$\mathrm{SE}(P_{\mathrm{tar}}, P_{\mathrm{trn}}) \approx \frac{1}{N_r} \sum_{\boldsymbol{x}_i \in \mathcal{D}} (P_{\mathrm{tar}}(\boldsymbol{x}_i) - P_{\mathrm{trn}}(\boldsymbol{x}_i))^2. \tag{4.9}$$

Substituting P_{trn} as given in Eq. 4.1 gives

$$\mathrm{SE}(P_{\mathrm{tar}}, P_{\mathrm{trn}}) \approx \frac{1}{N_r} \sum_{\boldsymbol{x}_i \in \mathcal{D}} (P_{\mathrm{tar}}(\boldsymbol{x}_i) - \sum_{m=1}^{M} W_m P_m(\boldsymbol{x}_i))^2. \tag{4.10}$$

As with the BD, the SE requires not only the calculation of the training PDFs P_m, but also that of the target PDF P_{tar} to determine the weights.

4.2.2.4 Optimization

The optimal weights W_m are determined with constrained optimization by minimizing the KL, BD, or SE distance criteria in Equations 4.4, 4.7, and 4.10 together with the constraints $\mathbf{0} \leq \boldsymbol{W} \leq \mathbf{1}$ and $|\boldsymbol{W}| = 1$.

Note that all three objective functions are convex, therefore a local minimum of the objective function is also the global minimum. In our experiments this minimization was performed with the interior-reflective Newton method [19] with the *fmincon* function of the Matlab optimization toolbox.

4.2.3 Weighted Support Vector Machine Classification

Once the optimal image weights W_m are determined, the training images and their corresponding weights are used to train a classifier. This is done by assigning every training sample $\boldsymbol{x}_i^{\mathrm{trn}}$ a weight w_i that equals the weight of its image, W_m. Next, a weighted SVM (WSVM) classifier [13] is trained on all training samples that received a non-zero weight w_i. First all

non-zero sample weights w_i are normalized so that they sum up to the total number of training samples N with a non-zero weight. This step is performed to facilitate comparison of the WSVM parameters to those of a regular SVM, where every sample is given a weight of one. Next, the decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \boldsymbol{x} + \boldsymbol{v_0}$ for the WSVM is calculated. After training, f can be used to classify an input sample $\boldsymbol{x_j}$, where the predicted label corresponds to the sign of $f(\boldsymbol{x_j})$. v and v_0 in the decision function are the model parameters that have to be optimized from the data by minimizing

$$\begin{aligned} \min_{v} \quad & \frac{1}{2}||\boldsymbol{v}||^2 + C\sum_{i=1}^{N} w_i\xi_i \\ \text{s.t.} \quad & y_i^{\text{trn}}(\boldsymbol{v}^T\boldsymbol{x}_i^{\text{trn}} + v_0) \geq 1 - \xi_i \\ & \xi_i \geq 0 \\ & \forall \boldsymbol{x}_i^{\text{trn}} : w_i > 0. \end{aligned} \tag{4.11}$$

This expressions equals that for the regular soft-margin Support Vector Machine (SVM) [21], except for the inclusion of the sample weights w_i. In this optimization, the term $||\boldsymbol{v}||^2$ maximizes the margin around the decision function, while the term $\sum_{i=1}^{N} w_i\xi_i$ minimizes the total weight of samples that are either misclassified or lie within the margin. ξ_i are so-called slack variables. A sample $\boldsymbol{x}_i^{\text{trn}}$ receives a value $\xi_i > 1$ if it is misclassified, a value ξ_i, $0 < \xi_i \leq 1$ if it is correctly classified but lies within the margin, and a value $\xi_i = 0$ otherwise. C is the SVM parameter that trades off between maximization of the margin $||\boldsymbol{v}||^2$ and minimization of $\sum_{i=1}^{N} w_i\xi_i$.

The WSVM in Eq. 4.11 given above can only produce linear decision functions. We used kernel learning [82] to produce non-linear decision functions. In a kernel SVM a mapping ϕ is used to map every sample $\boldsymbol{x}_i^{\text{trn}}$ into a new (possibly higher-dimensional) feature space $\phi(\boldsymbol{x}_i^{\text{trn}})$. In this new feature space a decision function $f(\boldsymbol{x}) = \boldsymbol{v} \cdot \phi(\boldsymbol{x}) + \boldsymbol{v_0}$ can be determined by optimizing the criterion in Eq. 4.11 where the first constraint is changed to $y_i^{\text{trn}}(\boldsymbol{v}^T\phi(\boldsymbol{x}_i^{\text{trn}}) + v_0) \geq 1 - \xi_i$. If a non-linear mapping is used, this new decision function corresponds to a non-linear decision function in the original feature space.

4.3 Experiments

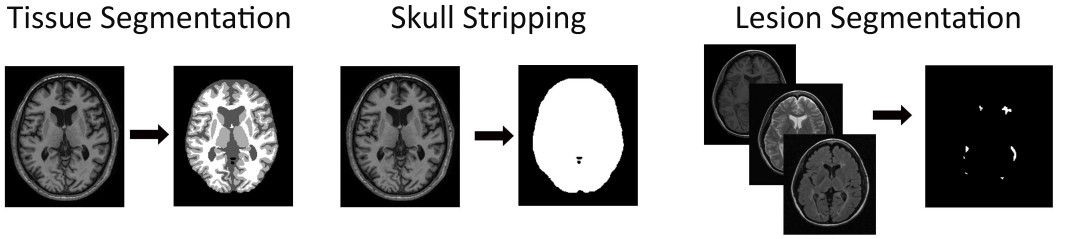


Figure 4.2: *The three applications presented in this paper: brain-tissue segmentation, skull stripping, and white-matter-lesion segmentation.*

We performed experiments on three MRI brain segmentation applications: brain-tissue segmentation, skull stripping, and white-matter-lesion (WML) segmentation. For brain-tissue segmentation we classified each voxel inside a manually segmented brain mask as either white matter (WM), gray matter (GM), or cerebrospinal fluid (CSF). For skull stripping we classified each voxel as either brain or background, where the brain included the CSF (both the ventricles and the sulcal CSF), WM, GM (including the cerebellum), and the brain stem. For lesion segmentation each voxel within the brain was classified as either WML or non-WML tissue. Figure 4.2 schematically shows the goal of these three tasks. All experiments included training data from multiple studies acquired with different scanners.

For all three applications we performed two sets of experiments. Firstly, we performed a set of leave-one-study-out cross-validation experiments where the classifiers were trained on only different-study images, i.e. images from different studies than the target image. Secondly, we performed a set of leave-one-image-out cross-validation experiments where both same- and different-study images were used for training. This way we investigated whether our algorithms would receive higher weights to same-study training images than to different-study training images, and whether our transfer-learning framework can also be beneficial if some same-study training images are available. For both types of experiments mean classification errors were obtained by selecting every image as target image once. Additionally, we performed a set of experiments to study the influence of the number of features used to determine the PDFs and therefore the image weights.

4.3.1 Data

4.3.1.1 Brain-Tissue Segmentation

For the experiments on brain-tissue segmentation MR images with corresponding manual segmentations from five studies were used:

1. 6 T1-weighted images from the Rotterdam Scan Study [52], acquired with a 1.5T GE scanner with $0.49 \times 0.49 \times 0.80$ mm^3 voxel size
2. 5 T1-weighted images from the MRBrainS13 challenge[2], acquired at the UMC Utrecht, the Netherlands, with a 3T Philips scanner with $0.958 \times 0.958 \times 3.0$ mm^3 voxel size
3. 18 T1-weighted images from the Internet Brain Segmentation Repository (IBSR) [116], acquired with an unknown number and type of scanners, with voxel sizes between $0.84 \times 0.84 \times 1.5$ mm^3 and $1 \times 1 \times 1.5$ mm^3
4. 20 T1-weighted images from the IBSR [116], of which 10 were acquired with a 1.5T Siemens scanner and 10 were acquired with a 1.5T GE scanner, all with $1 \times 3.1 \times 1$ mm^3 voxel size (which image was obtained with which scanner is not known)
5. 12 half-Fourier acquisition single-shot turbo spin echo (HASTE) images, scanned with a HASTE-Odd protocol (inversion time = 4400 ms, TR = 2800 ms, TE = 29 ms) from the Rotterdam Scan Study [52], acquired with a 1.5T Siemens scanner with $1.25 \times 1 \times 1$ mm^3 voxel size. These HASTE-Odd images have image contrast comparable to inverted T1 intensity.

All five studies used different scanners and different scanning parameters, and for all images a manual skull strip and tissue segmentation was available. Images from Study 3 and 4 have the cerebellum included in the tissue segmentation, whereas images from Studies 1, 2, 5 did not.

The intensities of the 12 HASTE-Odd images were inverted prior to calculation of the features since these images have inverted intensities compared to the T1-weighted images.

[2] http://mrbrains13.isi.uu.nl

4.3.1.2 Skull Stripping

For the experiments on skull stripping MR images with manual skull strips from Studies 1-4 of the brain-tissue experiments were used. Study 5 was not included in these experiments because we did not have manual skull strips that included the cerebellum and the brain stem.

4.3.1.3 White-Matter-Lesion Segmentation

For the experiments on lesion segmentation images with manual lesion segmentations from three studies were used:

1. 20 healthy elderly subjects from the Rotterdam Scan Study [52], scanned with three sequences: T1, PD, and FLAIR, with $0.49 \times 0.49 \times 0.80$ mm^3 voxel size
2. 10 MS patients from the MS Lesion Challenge [89], scanned at the Children's Hospital of Boston with three sequences: T1, T2, and FLAIR, with $0.5 \times 0.5 \times 0.5$ mm^3 voxel size
3. 10 MS patients from the MS Lesion Challenge [89], scanned at the University of North Carolina with three sequences: T1, T2, and FLAIR, with $0.5 \times 0.5 \times 0.5$ mm^3 voxel size

All three studies used different scanners and scanning parameters. As the PD images of Study 1 appear similar to the T2-weighted images of Study 2 and 3, we decided to treat these modalities to be the same.

4.3.2 Preprocessing

The x, y, and z axes of all images were oriented to be the same (for all images orienting the noses, left-right, and top of the head in the same direction) and normalized for brain size by scaling between zero and one in every direction. All images were corrected for non-uniformity with the N4 method [94]. Subsequently the image intensities were normalized by a range-matching procedure that scaled the voxels within a mask such that the voxels between the 4th and 96th percentage in intensity were mapped between zero and one. This range matching was performed within the mask used for segmentation.

4.3.2.1 Skull Stripping

Since the four studies for the skull stripping show a large variability in field of view, which would influence the images' PDFs, we performed an initial skull stripping by running SPM8 [4] with default parameters (but without bias-field correction). This overall resulted in relatively large brain masks. To ensure that all manually selected brain voxels were included, we performed an additional dilation with a spherical structuring element with a radius of 30mm.

4.3.2.2 White-Matter-Lesion Segmentation

For the experiments on lesion segmentation the brain-extraction tool [88] was used to perform skull stripping. Since lesions appear bright on FLAIR images, only voxels with a FLAIR intensity above a pre-selected threshold were used to determine the image weights and perform the classification. This threshold was manually set to 0.75 on the normalized FLAIR intensity, excluding almost all CSF and some of the gray-matter voxels.

4.3.3 Features

Classification was performed on 1) the normalized voxel intensities, 2) the normalized intensities after convolution with a Gaussian kernel at different scales, 3) the gradient magnitude of the normalized intensities after convolution with a Gaussian kernel at different scales. 4) the Laplacian of the normalized image intensities after convolution with a Gaussian kernel at different scales. The used scales were 1, 2, and 4 mm^3 for tissue segmentation, 1, 2, 4, and 8 mm^3 for skull stripping, and 0.5, 1, and 2 mm^3 for lesion segmentation. For lesion segmentation smaller scales were used because the images used for this application overall had smaller voxels than for the other two applications.

For tissue segmentation and skull stripping the cylindrical coordinates of the voxels, R, θ, and z were added as spatial features. For every slice z the center of the brain C was determined as the point (0.5, 0.5), and the front of the brain F was determined as the point (0, 0.5). For every voxel we calculated: 1) the distance R between the voxel and C, 2) the angle $\theta \in [0, \pi]$ between 1. the axis that passes through F and C and 2. the axis that passes through the voxel and C, 3) the normalized slice height z. By taking $\theta \in [0, \pi]$ we avoided a discontinuity at

$\theta = 0$ or $\theta = \pi$. No spatial features were included in the lesion segmentation since these can be misleading if lesions in a target image appear at different locations than in the training images.

For tissue segmentation and skull stripping only T1/inverted HOdd images were available, which resulted in a total of 13 features for tissue segmentation and 16 features for skull stripping. For lesion segmentation features were extracted from the T1, T2/PD, and FLAIR images, which resulted in a total of 30 features. Within each study all features were normalized to zero mean and unit standard deviation.

The PDFs were determined on only seven features, because PDF estimation on all features could computationally be very expensive. The choice of seven features was validated in a set of experiments where the features were ordered and consecutively added by forward feature selection as described in Sect. 4.3.5.

4.3.4 Classifiers

In all experiments we compared the performance of multiple classifiers: 1) a regular (unweighted) SVM on samples from all training images (SVM_A) as a baseline; a WSVM with weights determined by 2) the KL (WSVM_KL); 3) BD (WSVM_BD); 4) SE (WSVM_SE); a regular SVM trained on the single best image according to 5) the KL (SVM_KL_Best); 6) BD (SVM_BD_best); 7) SE (SVM_SE_best). For the second set of experiments we also trained 8) a regular SVM on the same-study training images only (SVM_S).

For all eight classifiers a Gaussian kernel was used, of which the kernel parameter was determined as described in Section 4.3.5. For calculation of the SVM and the WSVM classifiers an implementation in LIBSVM [13] was used.

4.3.4.1 White-Matter-Lesion Segmentation

For the experiments on lesion segmentation the SVM classifiers were trained and tested on datasets of voxels with a FLAIR intensity above 0.75. After discarding all voxels below this threshold the lesion voxels constituted still only a very small fraction of voxels, namely 1.61%

for Study 1, 1.31% for Study 2, and 0.22% for Study 3.[3] To train a more accurate classifier, lesion voxels were given a 10 times as high chance of being selected in the datasets as non-lesion voxels. Note that these datasets were only used for training and testing of the classifier, the weights were determined on a random subset of all voxels with a FLAIR intensity above 0.75.

4.3.5 Feature Selection and Parameter Settings

We optimized the SVM parameter C, the SVM kernel parameter γ, and the feature ordering by three-dimensional grid search on the training images with a non-weighted SVM. We used forward feature selection, where consecutively the best feature was added based on accuracy. The determined C and γ were used for all classifiers.

For the set of experiments with same- and different-study training data we performed leave-one-image-out grid search. For the set of experiments with only different-study training data we performed leave-one-study-out grid search. Leave-one-study-out grid search is especially designed for multiple-source cross validation, and generally selects variables that best generalize on images from previously unseen studies [35].

All classifiers were trained on a total of 50 000 training samples, regardless of the number of training images used to train the classifier. Mean classification errors were reported on 50 000 random samples per target image.

The PDFs were estimated on 10 000 random samples per image by kernel density estimation. A multivariate Gaussian kernel was used with a $d \times d$ covariance matrix S, where d denotes the number of features. We used $S = \sigma_S I$, where I denotes the $d \times d$ identity matrix, and σ_S denotes the kernel width, which is a common choice in kernel density estimation. The value σ_S was determined with Silverman's rule [85]:

$$\sigma_S = \left(\frac{4}{d+2}\right)^{\frac{1}{d+4}} N_m^{\frac{-1}{d+4}} \sigma. \tag{4.12}$$

[3] The images of Study 3 contained fewer lesion voxels than images from the other two studies, since these subjects have fewer lesions, but also because these lesions were more conservatively segmented.

Here N_m denotes the number of samples from image m (10 000) and σ the standard deviation of these samples (1 because of the feature normalization). This expression for σ_S is proved by Silverman [85] to minimize the Mean Integrated Square Error (MISE) between the actual and the estimated PDF for a multivariate Gaussian kernel.

The difference between the training and target PDF according to the KL, BD, and SE was evaluated on 10 000 points. For the KL these were the same 10 000 points used to determine the target PDF, for the BD and SE these were 10 000 uniformly distributed samples between the minimum and maximum value for each of the features attained for the target samples.

4.4 Results

4.4.1 Different-Study Training Data

Table 4.1: *Classification results for the experiments with only different-study training data. For each application the mean classification error is shown of which the lowest and all errors that are not significantly higher are shown in bold. "*" is added for classifiers that significantly outperformed SVM_A. Significances were determined with a two-sided paired t-test with the significance threshold at $P = 0.05$.*

Classifier	Classification Error		
	Tissue	Skull	Lesion
SVM_A	20.01%	**7.45%**	10.45%
WSVM_KL	**15.25%***	**7.48%**	**7.26%***
WSVM_BD	**15.43%***	**7.43%**	8.28%*
WSVM_SE	**15.44%***	**7.54%**	8.59%
SVM_KL_Best	**15.75%***	8.91%	**7.58%***
SVM_BD_Best	**16.55%***	8.31%	**8.00%**
SVM_SE_Best	20.28%	8.83%	11.24%

Table 4.1 gives for all three applications the mean classification errors when the training set contains only different-study images. P-values for a pairwise comparison between the classifiers can be found in Table 4.3 the appendix. For two out of three applications weighting gave a significant improvement over not weighting. For brain-tissue segmentation all three WSVMs and the SVM_KL_Best and SVM_BD_Best all performed significantly better than the

baseline SVM_A, but not significantly different from each other. For the skull stripping the three WSVMs and SVM_A performed significantly better than the three classifiers trained on the single best image, but not significantly different from each other. For lesion segmentation WSVM_KL, WSVM_BD, and SVM_KL_Best performed significantly better than SVM_A, of which WSVM_KL performed significantly better than WSVM_BD.

Fig. 4.3 shows the contribution of each of the training studies to the total weight. For all three applications the difference between the three weighting methods were quite small and the methods generally agreed which studies should receive a high or low weight.

4.4.2 Same- and Different-study Training Data

Table 4.2: *Classification results for the experiments with same- and different-study training data. For each application the mean classification error is shown of which the lowest and all errors that are not significantly higher are shown in bold. "*" is added for classifiers that significantly outperformed SVM_A and "†" for classifiers that significantly outperformed SVM_S. Significances were determined with a two-sided paired t-test with the significance threshold at* $P = 0.05$.

Classifier	Classification Error		
	Tissue	Skull	Lesion
SVM_A	13.88%	5.43%	3.50%
SVM_S	8.67%*	4.33%*	**2.61%***
WSVM_KL	8.45%*	4.23%*	3.00%*
WSVM_BD	**8.02%***†	**4.01%***†	2.85%*
WSVM_SE	8.44%*	4.35%*	3.01%*
SVM_KL_Best	8.68%*	4.76%*	7.61%
SVM_BD_Best	8.44%*	4.79%*	4.25%
SVM_SE_Best	9.64%*	4.81%*	4.88%

Table 4.2 shows the results with both same- and different-study training data for the three applications. P-values for a pairwise comparison between the classifiers can be found in Table 4.4 the appendix. For all three applications weighting significantly improved over not weighting. For the tissue segmentation and skull stripping the baseline SVM (SVM_A) was significantly outperformed by all other classifiers. For both applications the best results were obtained with WSVM_BD, which significantly outperformed all other classifiers. The performances of

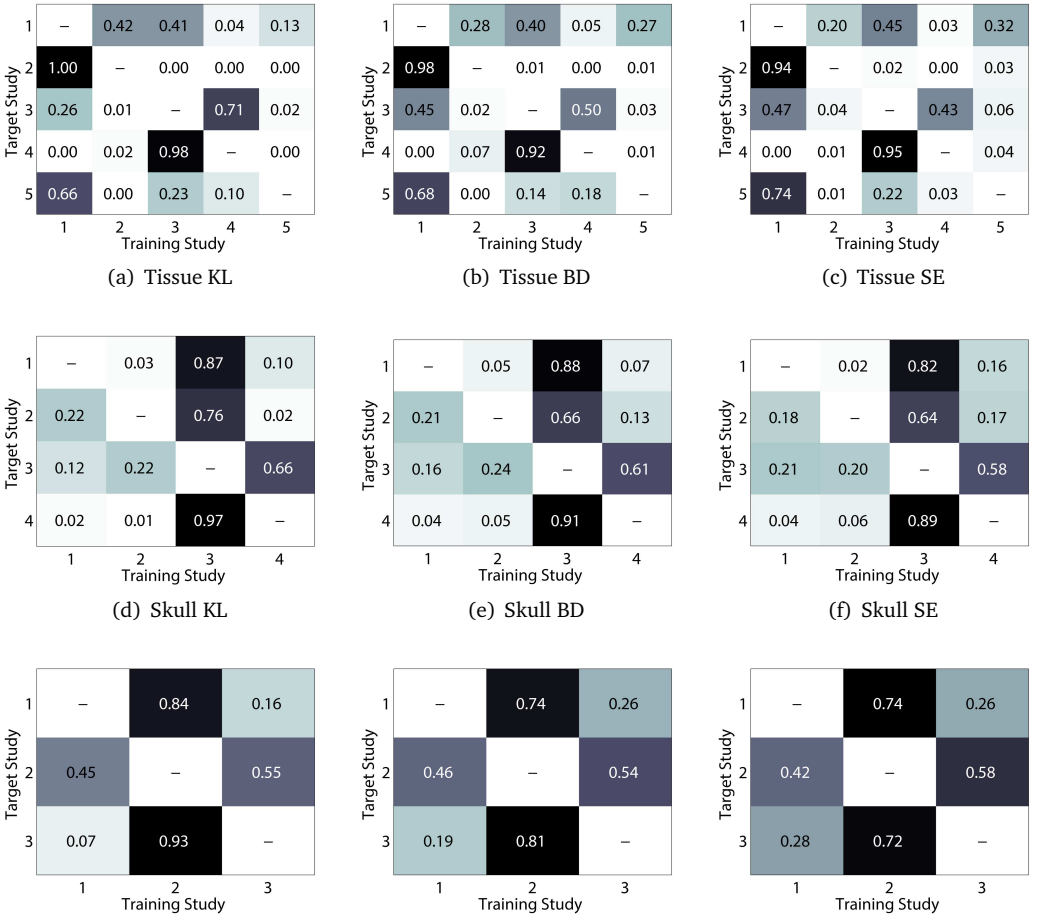


Figure 4.3: *Matrices showing the mean total weight per study for WSVM_KL, WSVM_BD, and WSVM_SE, for the experiments with only different-study training data. Each row in the matrices represents the experiments with target data from the shown study. The columns give the total weight given to each of the training studies, averaged over the classification of all target images.*

WSVM_KL and WSVM_SE were not significantly different from each other. For the tissue segmentation the three _Best classifiers did not perform significantly worse than WSVM_KL and WSVM_SE, whereas for skull stripping and lesion segmentation the three _Best classifiers performed significantly worse than all WSVMs. This indicates that jointly weighting all training images can be beneficial compared to training only on the single best training image.

For tissue segmentation and skull stripping the best WSVM, WSVM_BD significantly outperformed not only SVM_A, but also SVM_S. This means that even if same-study training images are available, adding different-study training images and weighting all training images can bring significant improvement over a non-weighted classifier on the same-study images.

For lesion segmentation the WSVMs performed significantly better than SVM_A, but significantly worse than SVM_S. Apparently assigning image weights by maximizing distribution similarity outperforms weighting all images equally, but if one knows beforehand which of the training images are same-study training images, it is beneficial to train on these images only. Note that in this application we are dealing with a much larger imbalance between class priors than in the other two applications. Even after discarding all voxels with a normalized FLAIR intensity below 0.75, the mean number of lesions voxels was only between 1 and 2% of the total number of voxels. As a result, the weights were much more determined on the similarity of white- and gray-matter voxels than on the similarity of lesion voxels. Since the three WSVMs outperformed SVM_A, our weighting scheme seems to find some useful information on the similarity between training and target images, but not enough to obtain similar performance as SVM_S.

The mean contribution of the various training studies can be found in Fig. 4.4. For all three applications the training images from the target study received the highest total weight. In all but one case there was some weight given to the different-study images. As with the experiments on only different-study training images, the figure shows only a small difference in weight distribution over the sources between the three weighting techniques.

4.4.3 Number of Features Used to Determine Weights

We also studied the influence of the number of features used to determine the PDFs and the image weights. Fig. 4.5 shows the mean improvement over SVM_A as a function of the number of features, averaged over the segmentation of all target images and all three applications.

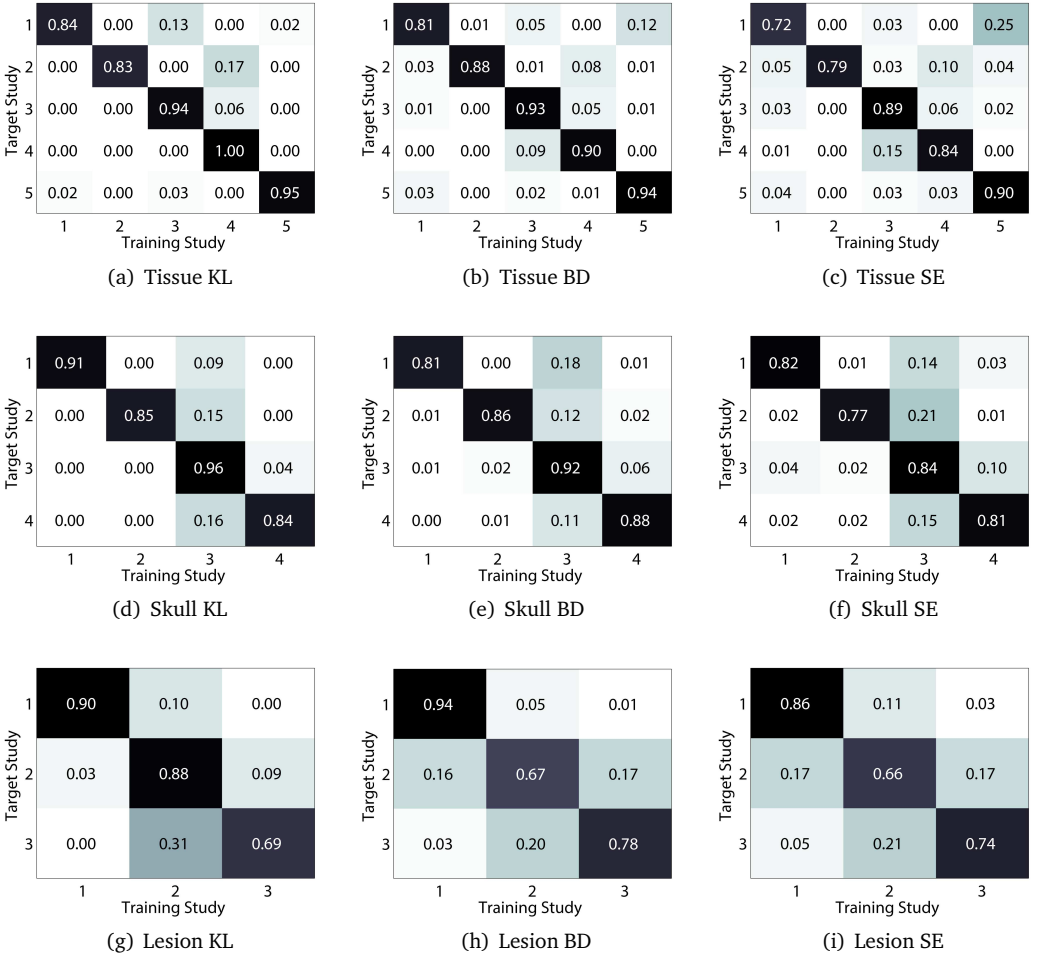
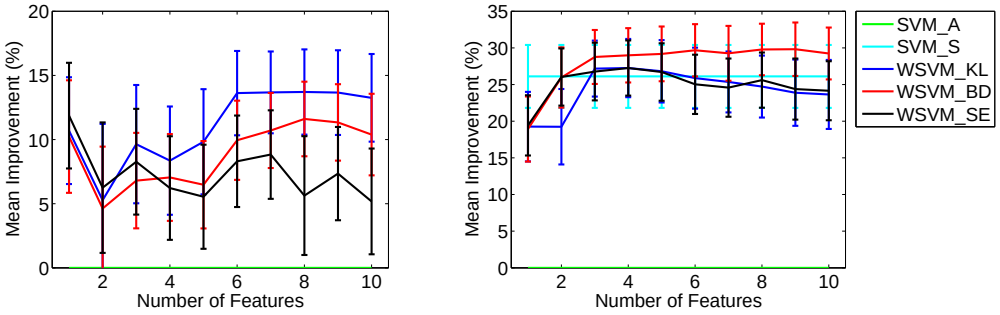


Figure 4.4: *Matrices showing the mean total weight per study for WSVM_KL, WSVM_BD, and WSVM_SE, for the experiments with same- and different-study training images. Each row in the matrices represents the experiments with target data from the shown study. The columns give the total weight given to each of the training studies, averaged over the classification of all target images.*



(a) Different (b) Same + Different

Figure 4.5: *Mean improvement (in %) over the baseline SVM_A classifier as a function of the number of features used to determine the image weights. The results are averaged over all target images and over all three applications. The errorbars give the 95%-confidence interval for the mean improvement over SVM_A. (a) gives the results for different-study training data only, (b) gives the results for same- and different-study training data.*

Fig. 4.5(a) shows the results for only different-study training data, Fig. 4.5(b) shows the results for same- and different-study training data. Note that the 95%-confidence intervals can be used to determine which classifier's mean performance was significantly ($P < 0.05$) different from that of SVM_A, but they cannot be used to determine whether the three WSVMs and SVM_S were significantly different from each other. We conclude that regardless of the number of features used to determine the weights, all three WSVMs significanty outperformed SVM_A. For same- and different-study training data, WSVM_BD had lower average classification error than SVM_S for 3 to 10 features, but this difference was not significant, although it was significant for tissue segmentation and lesion segmentation.

In our other experiments we used seven features to determine the image weights. For only different-study training data (Fig. 4.5(a)) for WSVM_KL and WSVM_BD using 6 to 10 features gave no significantly different result from using seven features. For WSVM_SE using six to nine features gave no significantly different result from using seven features For same- and different-study training data (Fig. 4.5(b)) for WSVM_KL and WSVM_SE the optimal number of features was only three, four, or five features, whereas for WSVM_BD the optimal number ranged from 3 to 10 features.

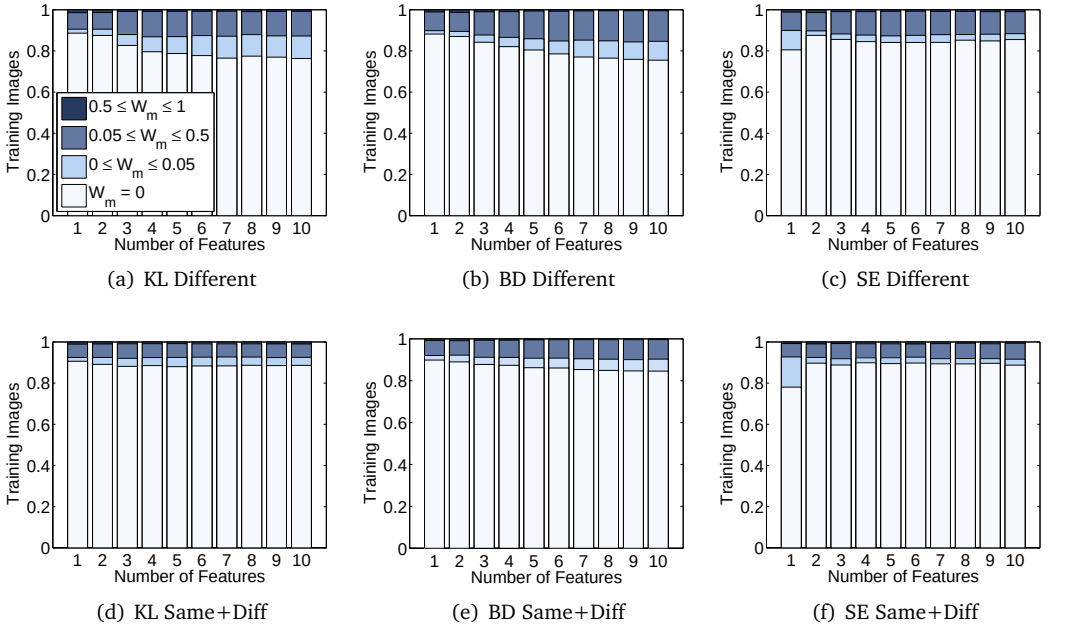


Figure 4.6: *Cumulative plots showing the distribution of image weights for WSVM_KL, WSVM_BD, and WSVM_SE as a function of the number of features used to determine the weights. The y-axis shows the percentage of training images that obtained a certain weight, averaged over all three applications. (a)-(c) show the results for different-study training data, (d)-(f) show the results for same- and different-study training data.*

Fig. 4.6 shows how the number of features influenced the distribution of weights over the training images. In all cases the majority of training images was given a weight of zero, quite some images were given a weight between 0 and 0.5, and a few images were given a weight between 0.5 and 1. The distribution of weights for WSVM_KL and WSVM_BD was very similar, but for WSVM_SE the distribution of weights was slightly different. For all plots the number of features did not seem to influence the number of images that received a weight above 0.05, but for WSVM_KL and WSVM_BD (but not for WSVM_SE) the number of images with a weight between 0 and 0.05 increased with the number of features. This means that for WSVM_KL and WSVM_BD an increase in features resulted in an increase of the number of images that were

used to train the classifier (i.e. all images with a weight above 0). For WSVM_SE the number of images used to train the classifier was highest at one feature, and almost constant for 2 to 10 features.

Comparing Fig.4.6(a)-(c) to Fig.4.6(d)-(f) shows that in the experiments with only different-study training data more images were given a weight between 0 and 0.5 and fewer images were given a weight of 0 than on same- and different-study training data. This is as expected: if same- and different-study training images are available we expect many different-study training images to receive a weight of 0, since same-study training images are assumed to be more similar to the target image than different-study training images. If only different-study training images are available, we also expect some training images to receive a weight of 0, but how many depends on whether the target image is similar to images from only one study, or from multiple studies. That both scenarios happened in our experiments can be seen in the matrices in Fig. 4.3.

4.5 Conclusion and Discussion

We presented an image-segmentation algorithm that can be trained on images from different studies (images that were made with different scanners and scanning protocols) than the target image. Our method uses supervised voxelwise classification with a weighted SVM (WSVM) classifier, where weights are based on Probability Density Function (PDF) similarity between the weighted training data and the target data. We experimented with three measures for this similarity: the Kullback-Leibler divergence (KL), the Bhattacharyya distance (BD), and the Squared Euclidean distance (SE).

We performed a set of experiments on brain-tissue segmentation, skull stripping, and white-matter-lesion segmentation. The experiments showed that our algorithm could be used when only labeled training data from other studies is available as well as when some training data from the target study is available. In five out of six experiments our weighting scheme significantly outperformed not weighting. For the experiments where only training data from other studies was available, weighting significantly outperformed not weighting for tissue segmentation and lesion segmentation. For skull stripping, weighting and not weighting performed similar. For the experiments where some training data from the target study was available weighting significantly outperformed not weighting for all three applications. Training on the

single best training image according to the KL, BD, or SE performed significantly worse than weighting all training images in four out of six cases and similar in the other two cases.

For tissue segmentation and skull stripping, the best weighting method also significantly outperformed SVM_S, an SVM trained on all other training images from the target study. This shows that even when some training images from the same study are available adding training images from other studies and weighting all training images can be beneficial. Adding training images from other studies can be beneficial when few training images from the target study are available. Weighting training images from the target study can especially be beneficial if images are heterogeneous because of e.g. scanner differences, artifacts, or large patient differences. This was observed for the IBSR data (Studies 3 and 4 for the tissue segmentation and skull stripping), where images in the same study originated from multiple scanners and had considerable artifacts. For both studies we saw a significant improvement of the WSVM_BD classifier over SVM_S.

Overall, weighting with the BD performed better than weighting with the KL or SE. This might be because the BD focuses more on matching the PDFs at high-density areas than on matching low-density areas. This might be beneficial for applications where the low-density areas are not very informative for the classifier. For one of the two experiments on lesion segmentation however, weighting with the KL outperformed weighting with the BD and SE. This is probably because contrary to the BD, the KL tends to give a larger weight to low-density areas than to high-density areas. This may be useful in lesion segmentation since lesion voxels are often in the tail of the distribution. A weighting method that focuses on high-density areas, like the BD, therefore matches the other tissues (mainly white and gray matter), whilst a weighting method that focuses more on low-density areas weights more according to lesion resemblance. The SE, the weighting method that gives the same importance to all parts of the PDF, performed on average slightly worse than the BD and the KL. This might indicate that depending on the application certain parts of the PDFs can be more informative for weighting than others.

Experiments with feature sets of different sizes indicated that the WSVM_BD is not very sensitive to the number of features in the dataset. It could accurately estimate the PDFs for up to 10 features. The performance of the WSVM_KL and WSVM_SE on the other hand seemed to drop slightly when many features were used. Using too many features compared to the number of samples will increase the kernel size and thus make all PDFs look similar, which causes the WSVMs to converge to an ordinary unweighted SVM on all training samples. Although our experiments indicated no problems with determining the PDFs, in theory using many features

can deteriorate the performance. We would therefore advise to perform feature selection for datasets with many (e.g. more than 10) features. Also, it might be beneficial to determine the weighting only on the features that are most useful for classification.

In our experiments we slightly favored the SVM on all training samples (SVM_A) by using this classifier to select the optimal SVM and kernel parameter, C and γ. Selecting these parameters separately for each of the classifiers under consideration could slightly improve the performance of the transfer classifiers. However, we believe this would not have a big influence on the conclusions drawn in this paper.

The method presented in this paper was inspired by weighting methods for covariate shift, and especially by the Kullback-Leibler Importance Estimation Procedure (KLIEP) [90]. Many methods for covariate shift aim to weight training samples by the ratio between the target PDF and the training PDF [71]. KLIEP uses the Kullback-Leibler divergence to calculate this ratio without explicitly calculating the training and target PDF. In this paper we showed that weighting according to distribution similarity could be used for image weighting in cases where we are not dealing with covariate shift. Since our method assigns weights to images instead of samples it can handle the problem of differences in distributions and labeling functions fairly well. We also showed that although by using the KL the target PDF need not be calculated, using a different measure, such as the BD, might further improve performance.

In previous work [102] we presented four transfer classifiers that required some training data from the target study and a larger amount of training data from other studies. In these classifiers all training data from the other studies was given the same weight. The results of this paper show that it can be beneficial to weight training images differently depending on their similarity to the target image. In the workshop paper that preceded this paper [101] we used the same data, features, and classifier as for the tissue segmentation in Van Opbroek et al. [102], which allows for a direct comparison. Giving equal weights to all training samples from other studies than the target image (together with 3 training voxels from target study) gave a mean classification error of 20% [102]. Image weighting (without training voxels from the target study) decreased the mean classification error to 15% [101]. As this paper shows, this result might be improved even further by using the BD (Van Opbroek et al. [101] used the KL) and applying feature selection.

Our results convincingly show that our weighting method can improve performance both on training images from different studies than the target image and on training images from

the same study. We therefore believe that weighting according to voxel distributions can be beneficial not only for transfer classifiers such as the ones presented by [1, 97, 102], but also for traditional classifiers that use only training data from the target study.

In this paper we focused on MRI brain segmentation. However, the presented transfer-learning approach could also be used on other medical-image-segmentation tasks. A requirement for our approach is that a sufficiently large number of samples is available, so that the image PDFs could be estimated accurately. This means our approach could be used for segmentation by voxelwise classification, since one image provides many voxels, but classification of super voxels for example is also possible, given that an image provides enough training samples. For applications with highly unbalanced classes, such as lesion segmentation, our experiments indicate that although the presented weighting method might not be optimal, it can still give a large improvement over regular unweighted classification.

Our approach significantly outperformed regular supervised learning on a variety of applications and for a variety of available training sets. It facilitates the use of supervised image segmentation in situations where a representative training set (from the target scanner and patient group) is not available, which is often the case in clinical practice and in e.g. multicenter studies. Also, as the results suggest, even when such a training set is available, our weighting method might still be beneficial. We believe that by reducing the need for manually labeled same-study training images and enabling the segmentation of images regardless of the originating scanner, our method can be very valuable in research as well as clinical practice.

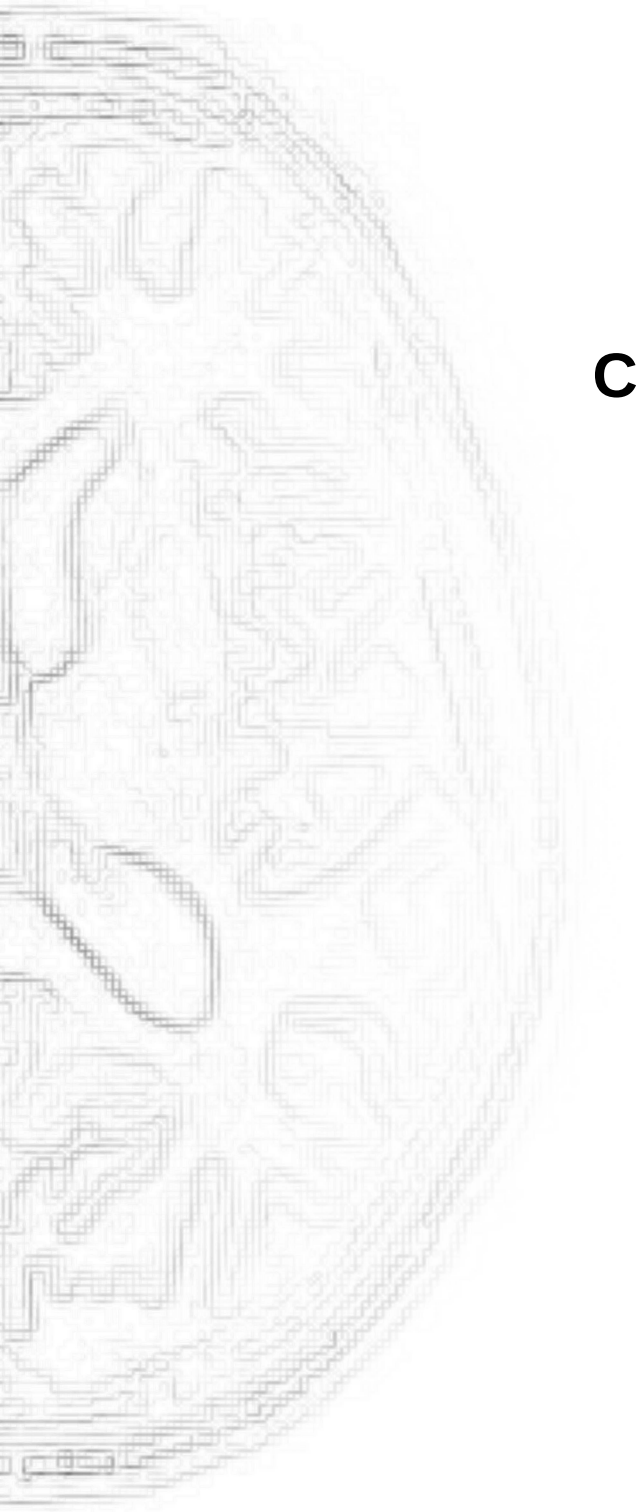
4.6 Appendix

Table 4.3: Classification results for the experiments with only different-study training data. For each application the mean classification error is shown of which the lowest and all errors that are not significantly higher are shown in bold. The table also shows a comparison of the performance of each classifier to that of the SVM_A, WSVM_KL, WSVM_BD, and WSVM_SE classifier. Significant results are in bold, significantly better results are denoted with a "*". Significance was determined with a two-sided paired t-test with the significance threshold at $P = 0.05$.

Classifier	Error	P-value of classifier when compared with			
		SVM_A	WSVM_KL	WSVM_BD	WSVM_SE
Tissue					
SVM_A	20.01%	-	**$1 \cdot 10^{-18}$**	**$2 \cdot 10^{-14}$**	**$5 \cdot 10^{-12}$**
WSVM_KL	**15.25%**	***$1 \cdot 10^{-18}$**	-	0.4	0.6
WSVM_BD	**15.43%**	***$2 \cdot 10^{-14}$**	0.4	-	1.0
WSVM_SE	**15.44%**	***$5 \cdot 10^{-12}$**	0.6	1.0	-
SVM_KL_Best	**15.75%**	***$2 \cdot 10^{-9}$**	0.3	0.4	0.5
SVM_BD_Best	**16.55%**	***$4 \cdot 10^{-3}$**	0.2	0.3	0.3
SVM_SE_Best	20.28%	0.9	**$4 \cdot 10^{-3}$**	**$4 \cdot 10^{-3}$**	**$3 \cdot 10^{-3}$**
Skull					
SVM_A	**7.45%**	-	0.8	0.9	0.6
WSVM_KL	**7.48%**	0.8	-	0.5	0.7
WSVM_BD	**7.43%**	0.9	0.5	-	0.2
WSVM_SE	**7.54%**	0.6	0.7	0.2	-
SVM_KL_Best	8.91%	**$5 \cdot 10^{-5}$**	**$2 \cdot 10^{-5}$**	**$5 \cdot 10^{-6}$**	**$1 \cdot 10^{-4}$**
SVM_BC_Best	8.31%	**$5 \cdot 10^{-4}$**	**$4 \cdot 10^{-4}$**	**$9 \cdot 10^{-5}$**	**$1 \cdot 10^{-3}$**
SVM_SE_Best	8.83%	**$1 \cdot 10^{-5}$**	**$6 \cdot 10^{-6}$**	**$3 \cdot 10^{-6}$**	**$1 \cdot 10^{-5}$**
Lesion					
SVM_A	10.45%	-	**$7 \cdot 10^{-3}$**	**$3 \cdot 10^{-2}$**	$5 \cdot 10^{-2}$
WSVM_KL	**7.26%**	***$7 \cdot 10^{-3}$**	-	***$5 \cdot 10^{-3}$**	***$1 \cdot 10^{-3}$**
WSVM_BD	8.28%	***$3 \cdot 10^{-2}$**	**$5 \cdot 10^{-3}$**	-	0.1
WSVM_SE	8.59%	$5 \cdot 10^{-2}$	**$1 \cdot 10^{-3}$**	0.1	-
SVM_KL_Best	**7.58%**	***$3 \cdot 10^{-2}$**	0.5	0.2	$6 \cdot 10^{-2}$
SVM_BD_Best	**8.00%**	0.1	0.1	0.7	0.4
SVM_SE_Best	11.24%	0.1	**$5 \cdot 10^{-3}$**	**$1 \cdot 10^{-2}$**	**$2 \cdot 10^{-2}$**

Table 4.4: Classification results for the experiments with same- and different-study training data. Mean classification errors are shown and the lowest error is shown in bold. The table also shows a comparison of the performance of each classifier to that of the SVM_A, SVM_S, WSVM_KL, WSVM_BD, and WSVM_-SE classifier. Significant results are in bold, significantly better results are denoted with a "*". Significance was determined with a two-sided paired t-test with the significance threshold at $P = 0.05$.

Classifier	Error	P-value of classifier when compared with				
		SVM_A	SVM_S	WSVM_KL	WSVM_BD	WSVM_SE
Tissue						
SVM_A	13.88%	-	**$4 \cdot 10^{-12}$**	**$4 \cdot 10^{-12}$**	**$2 \cdot 10^{-13}$**	**$4 \cdot 10^{-12}$**
SVM_S	8.67%	***$4 \cdot 10^{-12}$**	-	0.4	**$3 \cdot 10^{-3}$**	0.4
WSVM_KL	8.45%	***$4 \cdot 10^{-12}$**	0.4	-	**$1 \cdot 10^{-6}$**	0.9
WSVM_BD	**8.02%**	***$2 \cdot 10^{-13}$**	***$3 \cdot 10^{-3}$**	***$1 \cdot 10^{-6}$**	-	***$2 \cdot 10^{-3}$**
WSVM_SE	8.44%	***$4 \cdot 10^{-12}$**	0.4	0.9	**$2 \cdot 10^{-3}$**	-
SVM_KL_Best	8.68%	***$5 \cdot 10^{-11}$**	1.0	0.2	**$9 \cdot 10^{-5}$**	0.3
SVM_BD_Best	8.44%	***$2 \cdot 10^{-12}$**	0.3	0.9	**$1 \cdot 10^{-2}$**	1.0
SVM_SE_Best	9.64%	***$9 \cdot 10^{-6}$**	0.2	0.1	**$3 \cdot 10^{-2}$**	0.1
Skull						
SVM_A	5.43%	-	**$1 \cdot 10^{-6}$**	**$4 \cdot 10^{-8}$**	**$4 \cdot 10^{-10}$**	**$3 \cdot 10^{-7}$**
SVM_S	4.33%	***$1 \cdot 10^{-6}$**	-	0.5	**$1 \cdot 10^{-2}$**	0.9
WSVM_KL	4.23%	***$4 \cdot 10^{-8}$**	0.5	-	**$1 \cdot 10^{-3}$**	0.2
WSVM_BD	**4.01%**	***$4 \cdot 10^{-10}$**	***$1 \cdot 10^{-2}$**	***$1 \cdot 10^{-3}$**	-	***$5 \cdot 10^{-4}$**
WSVM_SE	4.35%	***$3 \cdot 10^{-7}$**	0.9	0.2	**$5 \cdot 10^{-4}$**	-
SVM_KL_Best	4.76%	***$5 \cdot 10^{-3}$**	**$2 \cdot 10^{-2}$**	**$6 \cdot 10^{-5}$**	**$8 \cdot 10^{-7}$**	**$1 \cdot 10^{-2}$**
SVM_BD_Best	4.79%	***$2 \cdot 10^{-2}$**	**$3 \cdot 10^{-2}$**	**$1 \cdot 10^{-3}$**	**$2 \cdot 10^{-5}$**	**$3 \cdot 10^{-2}$**
SVM_SE_Best	4.81%	***$2 \cdot 10^{-2}$**	**$1 \cdot 10^{-2}$**	**$4 \cdot 10^{-4}$**	**$3 \cdot 10^{-6}$**	**$8 \cdot 10^{-3}$**
Lesion						
SVM_A	3.50%	-	**$6 \cdot 10^{-5}$**	**$3 \cdot 10^{-2}$**	**$3 \cdot 10^{-3}$**	**$5 \cdot 10^{-2}$**
SVM_S	**2.61%**	***$6 \cdot 10^{-5}$**	-	***$4 \cdot 10^{-2}$**	***$3 \cdot 10^{-2}$**	***$6 \cdot 10^{-3}$**
WSVM_KL	3.00%	***$3 \cdot 10^{-2}$**	**$4 \cdot 10^{-2}$**	-	0.2	1.0
WSVM_BD	2.85%	***$3 \cdot 10^{-3}$**	**$3 \cdot 10^{-2}$**	0.2	-	0.2
WSVM_SE	3.01%	***$5 \cdot 10^{-2}$**	**$6 \cdot 10^{-3}$**	1.0	0.2	-
SVM_KL_Best	7.61%	**$1 \cdot 10^{-3}$**	**$1 \cdot 10^{-4}$**	**$7 \cdot 10^{-4}$**	**$3 \cdot 10^{-4}$**	**$2 \cdot 10^{-4}$**
SVM_BD_Best	4.25%	0.1	**$8 \cdot 10^{-4}$**	**$7 \cdot 10^{-3}$**	**$2 \cdot 10^{-3}$**	**$2 \cdot 10^{-3}$**
SVM_SE_Best	4.88%	**$3 \cdot 10^{-2}$**	**$9 \cdot 10^{-4}$**	**$5 \cdot 10^{-3}$**	**$2 \cdot 10^{-3}$**	**$2 \cdot 10^{-3}$**



Chapter 5

Transfer Learning by Feature-Space Transformation: A Method for Hippocampus Segmentation Across Scanners

Annegreet van Opbroek, Hakim C. Achterberg, Meike W. Vernooij, M. Arfan Ikram, & Marleen de Bruijne. ***Submitted****.*

Many successful approaches in MR brain segmentation use supervised voxel classification, which requires manually labeled training images that are representative of the test images to segment. However, the performance of such methods often deteriorates if training and test images are acquired with different scanners or scanning parameters, since this leads to differences in feature representations between training and test data.
In this paper we propose a feature-space transformation (FST) to overcome such differences in feature representations. The proposed FST is derived from unlabeled images of a subject that was scanned with both the source and the target scan protocol. After an affine registration, these images give a mapping between source and target voxels in the feature space. This mapping is then used to map all training samples to the feature representation of the test samples.
We evaluated the benefit of the proposed FST on hippocampus segmentation. Experiments were performed on two datasets: one with relatively small differences between training and test images and one with large differences. In both cases, the FST significantly improved the performance compared to using only image normalization. Additionally, we showed that our FST can be used to improve the performance of a state-of-the-art patch-based-atlas-fusion technique.

5.1 Introduction

The segmentation of medical images gives quantitative information about the tissues and structures of interest, which can aid both research and clinical diagnosis. Compared to manual segmentation, automatic segmentation can save large amounts of time and eliminate the problem of inter- and intra-observer variability. A widely used and successful method to perform such segmentations is by voxelwise classification based on supervised learning. Here, a manually annotated training set is used to extract features and train a classification system in the determined feature space. Then, the same features are determined for the test voxels and the trained classifier is used to make a decision on which label they should receive. Supervised-learning methods are used for a variety of segmentations tasks, such as whole

brain (also called skull stripping) ([50]), brain tissue ([64]), white matter lesion ([25, 36]), and, combined with atlas registration, for segmentation of brain structures such as the hippocampus and cerebellum ([28, 95]).

Supervised-learning methods can perform very well if they are provided with a large enough training set that is representative of the test dataset. However, performance often deteriorates if training and test datasets have differences in appearance, which can lead to differences in sample distributions in the feature space that is used for the classification. These problems often happen because of differences between scanners or scanning parameters, for example in multi-center datasets. The most common way to deal with such differences between training and test data is by intensity normalization. Many methods in neuro-image segmentation use range matching, matching the mean and standard deviation of the datasets, or more extensive normalization techniques. Such extensive normalization techniques can roughly be separated into two groups, where the first group of methods first identify a tissue or multiple tissues of interest (such as white matter, grey matter, CSF, or background) in both the source and the target images and then match the peaks of these tissues in the intensity distributions ([17, 60, 78, 81]). The second group of intensity-normalization techniques aim to match the intensity distributions of training and test images as a whole, without information of the imaged tissues ([44, 54, 69, 113]). The method of [69] is most widely used in neuro-image segmentation. It is shown to improve performance on e.g. brain-tissue and white-matter-lesion segmentation, both on same-scanner images ([124]) and between scanners ([83]).

However, image normalization techniques have the disadvantage that they aim at normalizing the image intensity only, while classification methods are often also based on other image-derived features. On the other hand, extracting derived features such as Gaussian-scale-space features from intensity-normalized images may still lead to different representations between scan protocols. Images are normalized by different mappings, which propagate differently in the derived features. In this paper, we propose a method that maps not only the intensity of training and test images, but also all the other features used for the classification, all at the same time. We will call this mapping a *feature-space transformation* (FST), since it maps the entire feature space of a training image to that of a test image. Our method learns the feature-space transformation from images of subjects that were scanned with both the training and the test scan protocol. Since our method involves learning, it can be called a *transfer-learning* technique ([71]). Transfer learning (sometimes also called *domain adaptation*) is recently gaining attention in medical image segmentation, since it aims to build a robust classification system by somehow compensating for differences between the distributions of training and test

data. [102] showed that transfer-learning techniques can improve segmentation performance across scan protocols over intensity-normalization techniques such as range matching and the method of [69] in brain-tissue segmentation and white-matter-lesion segmentation.

A few papers have been published that apply transfer-learning techniques to neuro-image segmentation ([38, 56, 102, 104, 106]). Most methods aim to compensate for the difference between training and test data in the classifier, for example with a weighted classifier that weights training samples according to resemblance to the test data. These methods show to improve performance compared to traditional, unweighted, classifiers when training and test data are from different scanners or scan protocols. However, these methods have the disadvantage that they only select samples as is, rather than learning how to transform the distribution of the training samples in the feature space as to better match the distribution of test samples. Some deep-learning methods take a different approach to transfer learning by learning a representation that is shared between data different scanners or scan protocols ([56, 106]). This way, these methods learn a feature representation that is dataset invariant. We propose an approach for non-deep learning that, rather than learning a shared representation, maps the feature distribution of training samples directly to that of test samples. Our method learns an FST based on pairs of unlabeled images of one or multiple subjects that were scanned with both the source and target protocol. After transformation, a regular (non-transfer) classifier can be trained on the transformed features of the training data.

We performed a set of experiments on hippocampus segmentation in two heterogeneous datasets to show the added value of our FST over standard intensity normalization. Hippocampus segmentation is known to be a challenging task, since the gray levels of the hippocampus are very similar to those of neighboring structures such as the amygdala, thalamus, and caudate nucleus ([32]). Most hippocampus-segmentation methods are based on multi-atlas registration, where a set of training images (called *atlases*) are registered to the test image. The registered training images are then combined to obtain a final segmentation by an atlas-combination method such as majority voting or STAPLE ([112]). Performance can be greatly improved by combining registered atlases with appearance information such as voxel intensities in a supervised classifier (e.g. [22, 75, 95, 110, 121]). However, incorporating such appearance information is likely to lead to problems when training and test images are obtained with different scan protocols. To decrease difference between training and test data, [95] used intensity normalization to zero mean, unit norm and [22] used the technique of [69]. In this paper, we investigate whether the use of an FST in such algorithms could improve performance over intensity-normalization techniques.

A preliminary version of this work has been published as a workshop paper ([99]). The presented paper extends this workshop paper by thorough experiments on a new, enlarged version of the dataset presented in the workshop paper, one additional dataset, and comparison to STAPLE ([112]) and the patch-based-atlas-fusion method of [110].

5.2 Material and Methods

This section describes the proposed method and the data used in the experiments. The presented feature-space transformation is described in Section 5.2.1; Section 5.2.2 describes how the feature-space transformation is used in a voxel classifier; Section 5.2.3 presents the two datasets used in the experiments; and Section 5.2.4 describes the setup of the experiments.

5.2.1 Feature-Space Transformation

We determine a feature-space transformation (FST) based on unlabeled images of subjects scanned with both source and target scanner. These images are used to obtain correspondences in the feature space, which are used to transform the feature values of the training voxels. Figure 5.1 shows a schematic picture of the working of the FST.

5.2.1.1 Notation

We distinguish two groups of samples. The first group consists of training and test samples, where the labels of the training samples are used in a classifier to label the test samples. The second group consists of unlabeled source samples and target samples. These source and target samples originate from images of one or multiple subjects that have been scanned with both the source and the target scanner.

We first define the training and test samples. Let $\boldsymbol{x}_i^{s_m} \in \mathbb{R}^{d_{s_m}}$ denote a d_{s_m}-dimensional feature vector at voxel i from source scanner s_m and $y_i^{s_m} \in \mathbb{Z}$ its label. Similarly, $\boldsymbol{x}_i^t \in \mathbb{R}^{d_t}$ denotes a d_t-dimensional feature vector in an image from the target scanner t and $y_i^t \in \mathbb{Z}$ its label. Note that images from different scanners need not have the same number of features.

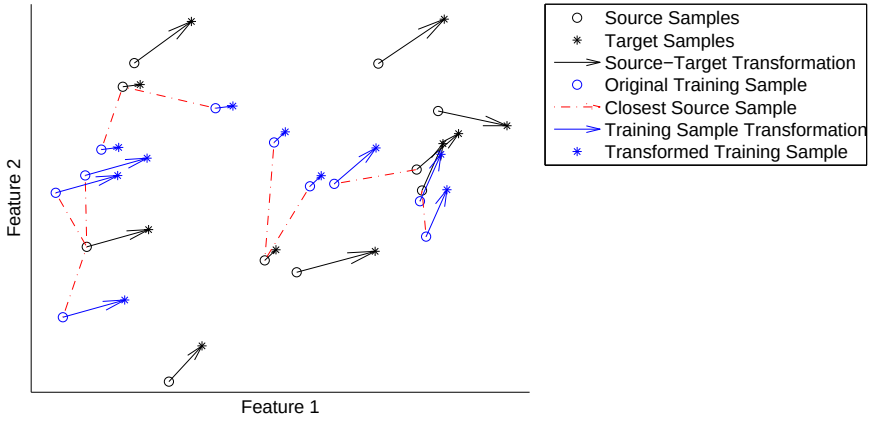


Figure 5.1: *Schematic picture of the presented FST. Unlabeled source-target samples (shown in black) are generated from images of a subject scanned with both the source and target scanner. Labeled training samples (shown in blue) are then linked to their closest k source samples (here, $k = 1$, shown in red) and given the median transformation of these k source samples, which results in transformed training samples.*

The distribution of samples from a certain scanner (i.e. all images scanned with that scanner) in the feature space is denoted with $F(\boldsymbol{x})$. The underlying feature distribution of the voxels from source scanner s_m is denoted with $F_{s_m}(\boldsymbol{x})$ and that of all voxels from the target scanner t with $F_t(\boldsymbol{x})$. Samples that originate from different scanners may have different distributions (i.e. $F_{s_n}(\boldsymbol{x}) \neq F_{s_m}(\boldsymbol{x})$ for $n \neq m$ and $F_{s_n}(\boldsymbol{x}) \neq F_t(\boldsymbol{x})$). The goal of our method is to determine an FST from s_m to t: $f_{s_m \to t} : F_{s_m}(\boldsymbol{x}) \to F_t(\boldsymbol{x})$, which transforms the training samples $\boldsymbol{x}_i^{s_m}$ from $F_{s_m}(\boldsymbol{x})$ to $F_t(\boldsymbol{x})$ by setting them to

$$\tilde{\boldsymbol{x}}_i^{s_m} = f_{s_m \to t}(\boldsymbol{x}_i^{s_m}). \tag{5.1}$$

The FST between a source scanner s_m and the target scanner t is learned from source and target samples: unlabeled voxels from images of N subjects that were scanned with both s_m and t. We call two images from the same subject obtained with s_m and t a *source-target pair*. These pairs should be acquired within a short time interval, so that the subject's anatomy

can be assumed unchanged. Let $\boldsymbol{z}_i^{s_m} \in \mathbb{R}^{d_{s_m}}$ denote sample i from the source image of the source-target pair and $\boldsymbol{z}_j^t \in \mathbb{R}^{d_t}$ sample j from the target image of the source-target pair.

5.2.1.2 FST Determination

The target images of every source-target pair are affinely registered to their corresponding source images. A nearest-neighbor interpolation of the target images then provides a voxelwise correspondence for every sample $\boldsymbol{z}_i^{s_m}$ to a sample $\boldsymbol{z}_\ell^t$:

$$\forall i: \ \exists\, \ell : \boldsymbol{z}_i^{s_m} \rightarrow \boldsymbol{z}_\ell^t \,. \tag{5.2}$$

For each training sample $\boldsymbol{x}_i^{s_m}$ inside the brain mask, we determine the closest k source samples $\{\boldsymbol{z}_{c_1^i}^{s_m}, \boldsymbol{z}_{c_2^i}^{s_m}, \ldots, \boldsymbol{z}_{c_k^i}^{s_m}\}$ in feature space, where c_k^i denotes the kth closest sample to training sample number i. The FST of $\boldsymbol{x}_i^{s_m}$ equals the transformation to the robust median[1] target sample of these k source samples:

$$f_{s_j \rightarrow t}(\boldsymbol{x}_i^{s_m}) = \boldsymbol{x}_i^{s_m} + \text{median}(\boldsymbol{z}_{\ell_1}^t - \boldsymbol{z}_{c_1}^{s_m}, \ldots, \boldsymbol{z}_{\ell_k}^t - \boldsymbol{z}_{c_k}^{s_m}), \tag{5.3}$$

where $\boldsymbol{z}_{\ell_n}^t$ $(n = 1, 2, \ldots, k)$ is the paired voxel of $\boldsymbol{z}_{c_n}^{s_m}$ as defined in Equation 5.2. We used the median (rather than the mean) to assure that the chosen transformation is one that is observed in the correspondence in Equation 5.2.

Say that $\boldsymbol{z}_{\ell_p}^t - \boldsymbol{z}_{c_p}^{s_m}$ is the median transformation for $\boldsymbol{x}_i^{s_m}$. Note that $\boldsymbol{x}_i^{s_m} - \boldsymbol{z}_{c_p}^{s_m}$ is supposed to be small, since these points are close in feature space. Therefore $f_{s_j \rightarrow t}(\boldsymbol{x}_i^{s_m}) \approx \boldsymbol{z}_{\ell_p}^t$, so $f_{s_j \rightarrow t}(\boldsymbol{x}_i^{s_m})$ is approximately distributed by the same distribution as $\boldsymbol{z}_{\ell_p}^t$, i.e. $F_t(\boldsymbol{x})$, the target distribution. A schematic picture of the working of the FST is displayed in Figure 5.1.

Higher k increases the regularization, which results in a smoother transformation. In our experiments where we compared the presented FST with other methods, we used $k = 1$ as

[1] The robust median gives the transformation that has minimal total distance to all k transformations.

default. In some extra experiments we showed the effect of increasing k.

5.2.2 Hippocampus Segmentation

The hippocampus segmentation is performed by voxelwise classification within a region of interest (ROI) around the hippocampus.

5.2.2.1 Multi-Atlas Probability

A set of atlases (where the hippocampus has value one and non-hippocampus has value zero) is used to determine a probability per training and test voxel of it being hippocampus. All atlases are non-rigidly registered to the training and test images as described in Section 5.2.3.4. The multi-atlas probability per voxel is then determined by averaging the values of all registered atlases.

The ROI is determined as all voxels with a multi-atlas prior probability of at least 10%. This threshold was chosen manually in a trade-off between accuracy and speed as to exclude as many non-hippocampus voxels and a few hippocampus voxels as possible.

5.2.2.2 Features

The multi-atlas probability is used as a feature in the classifier. Additionally, 10 local image-appearance features were used:

- the voxel intensity
- the intensity after a Gaussian smoothing at $\sigma = 1, 2.2,$ and 5 mm^3
- the gradient magnitude after a Gaussian smoothing at $\sigma = 1, 2.2,$ and 5 mm^3
- the Laplacian after a Gaussian smoothing at $\sigma = 1, 2.2,$ and 5 mm^3

These features are a subset, consisting of all rotationally invariant features of those used by [95] for hippocampus segmentation. Only the rotationally invariant features are chosen in order to cope with differences in patient orientation.

5.2.2.3 Classification

The segmentation is obtained by voxelwise classification with a support vector machine (SVM) ([21]) with a Gaussian kernel ([82]). It is trained on a uniformly randomly selected subset of samples inside the ROI of the training images. After training, the SVM is applied to all test samples within the ROI of the test images.

5.2.3 Data

We present results on two datasets. The first dataset consists of ADNI ([68]) data, which has been acquired with various scanners with similar scanning protocols. The second dataset consists of Rotterdam Scan Study ([51]) data, which has been acquired with two scanners with different scanning protocols. As a result, in the first dataset the differences in appearance between images from different scanners are much smaller than is the case in the second dataset.

5.2.3.1 Dataset1: Harmonized Protocol

The first dataset consists of Harmonized Protocol (HarP) data[2]. This dataset consists of 135 Alzheimer's Disease Neuroimaging Initiative (ADNI) images ([68])[3] with manual hippocampus segmentations ([8]). These 135 images were scanned at 34 sites, of which 12 sites scanned subjects with both a 1.5T and a 3T scanner. For 8 of these 12 sites we found at least four pairs

[2]http://www.hippocampal-protocol.net/

[3]The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). For up-to-date information, see http://www.adni-info.org.

of (unlabeled) images in the ADNI database[4] of subjects that were scanned with both the 1.5T and the 3T scanner within a month from each other. The 45 HarP images of these 8 sites were used as training and test data in cross validation. A maximum of four pairs of the unlabeled ADNI images per site were selected to be used as source-target pairs to determine the FST between the scanners. Table 5.1 gives the number of images per site and per scanner in the HarP dataset. Figure 5.2(a),(g) give an impression of the difference between a 1.5T and a 3T scan from the same site and Figure 5.2(b), (h) of their manual segmentation.

Table 5.1: *Subjects per site in the HarP datasets that were included in the training and test sets.*

	Number of images	
Site number	1.5T	3T
002	7	7
005	3	2
007	2	2
013	3	2
016	3	3
020	1	1
126	1	2
127	2	4
Total	22	23

5.2.3.2 Dataset 2: Rotterdam Scan Study

The second dataset consists of images of healthy elderly volunteers from the Rotterdam Scan Study (RSS) ([51]) with manual hippocampus segmentations. 20 images were obtained with a 1.5T Siemens scanner with a Haste-Odd protocol (inversion time = 4400 ms, TR = 2800 ms, TE = 29 ms)([53]); 18 images were obtained with a 1.5T GE scanner with a T1 protocol([51]). The datasets were segmented by different observers.

As source-target pairs we used rescan images of 9 subjects that were scanned with both scanners within a short time interval from each other. Figure 5.2(m),(s) show an example Haste-

[4]http://www.adni.loni.usc.edu

Odd image and T1 image and Figure 5.2(n),(t) show their manual hippocampus segmentations.

5.2.3.3 Preprocessing

All images were rigidly registered to MNI152 space with $1 \times 1 \times 1$ mm^3 voxel size as described in Section 5.2.3.4 and corrected for MRI bias field with the N4 method ([94]). Next, a brain mask was determined as follows. For the HarP dataset, the brain extraction tool (BET) ([88]) was run with default parameters on all images. Since this gave variable results, a second step was applied. Here, the BET segmentations of all images were non-rigidly registered to each other and per image a majority vote was performed. For the RSS data, a slightly different approach was used since BET gave bad results for the Haste-Odd images. Here, BET was run with default parameters on the 9 T1 rescan images. These masks were then transformed to the Haste-Odd rescan images by an affine registration of each T1 rescan image to its corresponding Haste-Odd rescan image. The final brain masks for all images (also the rescan images) were obtained by non-rigid registration of the 9 rescan images of the same scanner followed by a majority vote.

Before calculation of the appearance features, all images were normalized for intensity by a 4th-96th percentile range matching procedure within the brain mask. The appearance features were normalized per scanner to zero mean, unit variance within the brain mask. The multi-atlas probability was normalized to zero mean, unit variance based on the samples within the ROI around the hippocampus, since this feature is mostly zero outside the ROI.

5.2.3.4 Registration

All registrations were performed with the Elastix registration toolbox ([58]) based on maximizing normalized mutual information. We used the registration settings of [9], which were visually optimized for ADNI data.

The source-target pairs were registered to each other by a rigid registration followed by an affine registration, to compensate for possible distortion. The brain masks were registered by consecutively running a rigid, affine, and non-rigid registration. The multi-atlas probabilities were obtained by an initial rigid registration of the brain masks, followed by a rigid, affine,

and non-rigid registration of the images, where only voxels inside the brain masks contributed to the similarity measure.

5.2.4 Experimental Setup

5.2.4.1 Experiments

On the HarP dataset, we performed segmentation of all 45 images in Table 5.1 in cross validation, by training on images scanned with the other scanner at same site as the test image. We will refer to this dataset as HarP 1. As can be seen from the table, this meant that only between 1 and 7 training images could be used. Unfortunately, images from other sites could not be used, since no source-target pairs were available between sites. However, a possible way to improve the performance, which was also investigated, is by determining the multi-atlas-probability feature on all images from different scanners than the test image, which results in a total of 128 to 134 atlases. This data will be referred to as HarP 2.

On the RSS dataset, we performed segmentation of all 38 images by training on all images from the other scanner in cross validation, which resulted in either 18 or 20 training images. Additionally, we performed an experiment where we segmented the two rescan images of all 9 subjects in cross validation, by training the FST on one of the other rescan images. Here, both scanners were once used as training scanner to segment all rescan images; where we compared the difference in segmented volume between the two rescan images of all 9 subjects. This experiment was performed to study the influence of our method for the reproducibility of segmentations across scanners.

For both datasets, we also studied the influence of the number of source-target images, N and the number of neighbors, k, that are used in the FST. We also compared the performance of our SVM with FST to that of two established hippocampus-segmentation methods: STAPLE ([112]) and the multi-atlas label-fusion method of [111].

The performance of the various methods is measured in terms of Dice overlap ([27]) between the resulting segmentations and the manual segmentations, averaged for left and right hippocampus. Significance of differences is determined with a Wilcoxon signed-rank test per subject, with the significance threshold at $P = 0.05$. The repeatability in *RSS Rescan* is shown

in a Bland-Altman plot, which shows the difference in volume between the outputs of two methods as a function of the average volume of the two outputs.

5.2.4.2 Compared Methods

We compared the performance of the following methods:

Atlas MV: Majority vote, where the segmentation was obtained by thresholding the multi-atlas probability feature at 0.5.

SVM Atlas: The SVM classifier on just the multi-atlas probability feature. This method was added to determine how much of the difference in performance between the SVM method and the Atlas MV method can be explained by the probability feature.

SVM: The SVM classifier on the multi-atlas probability and the appearance features, without the feature-space transformation.

SVM FST: The SVM classifier on the multi-atlas probability and the appearance features with the feature-space transformation.

SVM FST Intensity: Similar to SVM FST, but with the FST applied to the intensity feature only. The other features were calculated from the transformed intensity image. This method was added to show the added value of transforming all features at the same time over transforming intensity alone.

Additionally, we compared the performance with that of two state-of-the-art hippocampus-segmentation methods; one that uses only atlas information and one that incorporates atlas and appearance information:

STAPLE: Here, atlases are combined with the Simultaneous Truth And Performance Level Estimation (STAPLE) algorithm ([112]).

Fusion: The multi-atlas-label-fusion method of [111], without corrective learning ([109]). This method won third place at the MICCAI 2012 Grand Challenge and Workshop on Multi-Atlas Labeling on hippocampus segmentation.

We also investigated a combination of our FST and *Fusion*:

Fusion FST Intensity: Here, the FST was determined in the feature space used for *SVM FST* in order to transform image intensities of the training data to those observed in the test data. The transformed intensity images were subsequently used for the patch-based fusion. Here, only the intensity was transformed, contrary to all features, because we used a readily available implementation of Fusion, which does not allow the transformation of all features. Note that *Fusion FST Intensity* could not be applied to the HarP 2 dataset, since this would require every training image to be transformed, which is not possible because source-target pairs are not available between all training and test images.

5.2.4.3 Implementation and Parameters

For all SVM classifiers, we used LIBSVM ([13]). For STAPLE the CRKIT[5] was used. For *Fusion* we used the implementation of the authors of the paper ([111])[6]. For both *STAPLE* and *Fusion* the default parameters were used.

All SVM classifiers were trained on 10 000 training samples. This number was chosen on a subset, as a trade-off between accuracy and computation time. The SVM slack parameter C and the kernel parameter γ were determined in cross validation on the training set. Here, for the HarP dataset leave-one-site-out cross validation was used. This way, the parameters were optimized to cope with differences between images from different sites. For the RSS dataset, leave-one-image-out cross validation was used, since there were not enough different sites (or scanners) in this dataset to perform leave-one-site-out cross validation.

Separate classifiers were trained for the left and right hippocampus, which improved performance compared to training a single classifier. Probably, this is because the left and right hippocampus has slightly different appearance.

The sample correspondence for the FST in Equation 5.2 was determined on all voxels within the intersection of the source and target brain masks. For the FST, we used $k = 1$ number of neighbors and $N = \max$ ($N = 4$ for HarP data and $N = 9$ for RSS data) number of

[5] http://crl.med.harvard.edu/software/CRKIT/index.php

[6] We used version 1.3, without corrective learning. The implementation is available at http://www.nitrc.org/frs/?group_id=634.

source-target image pairs. In Section 5.3.3, we investigate the influence of k and N.

5.3 Results

5.3.1 Comparison of Appearance Features With and Without FST

Table 5.2: *Mean Dice overlap of the various methods on 1) the HarP dataset trained on images from the same site other scanner than the test image, 2) the HarP dataset with multi-atlas probabilities determined from all HarP images except for the ones from the test scanner, 3) the RSS dataset. The best result and the results that are not statistically significantly worse, are shown in bold.*

Method	HarP 1	HarP 2	RSS
Atlas MV	0.725	0.793	0.791
SVM Atlas	0.729	0.827	0.797
SVM	0.743	0.786	0.726
SVM FST	**0.753**	**0.840**	**0.804**
SVM FST Intensity	0.690	0.727	0.411

Table 5.2 shows the mean performance for *Atlas MV*, *SVM Atlas*, *SVM*, *SVM FST*, and *SVM FST Intensity* on 1) the HarP dataset, 2) the HarP dataset with all different-scanner images used as atlas, and 3) the RSS dataset. For all three cases, training an SVM on only the multi-atlas probability (*SVM Atlas*) improved the performance over setting the multi-atlas threshold at 0.5 (*Atlas MV*). Adding appearance features (without FST), as in the *SVM* method, improved performance only in HarP 1. This overall decrease in performance is because appearance differs between the scanners and is therefore misleading for classification. In HarP 2, the performance decreased by adding appearance features without FST, probably because here better atlas information is available than in HarP 1. For the RSS dataset, where appearance differs much more between training and test images than in the HarP dataset, the appearance features harmed performance most. Adding appearance features with FST, as in the *SVM FST* method, significantly improved the performance over using only multi-atlas information (*SVM Atlas*) and using appearance features without FST (*SVM*) in all three cases. Applying the FST only on the intensity feature, as in *SVM FST Intensity*, performed much worse than applying the FST to all features in all three experiments.

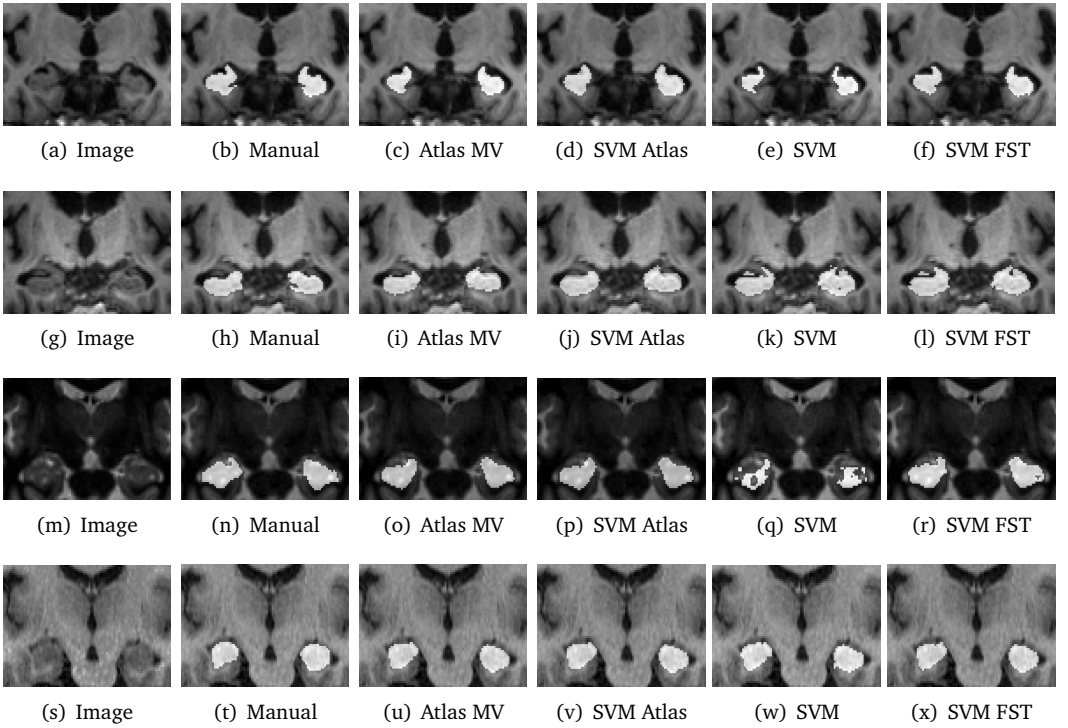


Figure 5.2: *Example hippocampus segmentations for the various methods overlaid on the bias-field corrected images. (a)-(f): HarP dataset with all different-scanner images used as atlas, 1.5T images segmented by training on 3T images and (g)-(l): 3T images segmented by training on 1.5T images. (m)-(r): RSS dataset, Haste-Odd images segmented by training on T1 images and (s)-(x): T1 images segmented by training on Haste-Odd images. Examples were chosen to have Dice overlap as close as possible to the mean Dice overlap on all images.*

Figure 5.2 shows example segmentations for *Atlas MV*, *SVM Atlas*, *SVM*, and *SVM FST* on the HarP 2 dataset and on the RSS dataset. For all four images, the methods that use only atlas information, *Atlas MV* and *SVM Atlas*, produced segmentations that are too smooth compared to the manual segmentations. The methods that combine atlas information and appearance information, *SVM* and *SVM FST* gave more detailed segmentations, where *SVM FST* gave the

best segmentations. In Figure 5.2(e) and 5.2(q) we can see that *SVM* produced an under segmentation because of the difference in appearance between training and test data. As can be seen from Figure 5.2(f) and 5.2(r) this problem was solved by using the FST.

The example segmentations also show a disadvantage of adding appearance features: it increases the chance of obtaining a segmentation with incorrect topology (i.e. an unconnected segmentation or a segmentation with a hole). This may happen in voxel classifiers that incorporate appearance features because non-neighboring voxels in the image might be close to each other in the feature space, while neighboring voxels in the image are not necessarily close in the feature space. This can easily be solved by a post-processing step such as morphological opening, taking the biggest connected component, smoothing of posterior outputs in the image space, or a graph cut ([95]).

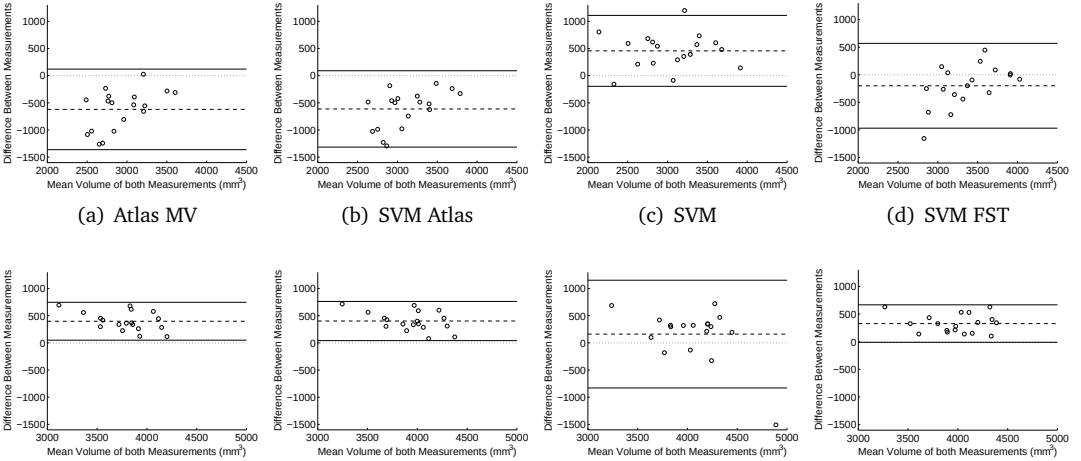
5.3.2 RSS Rescan Segmentation

Figure 5.3 shows Bland-Altman plots for *Atlas MV*, *SVM Atlas*, *SVM*, and *SVM FST* on segmenting the RSS rescan images. When training on the T1 images (Figure 5.3(a)-(d)), *SVM FST* showed a much smaller bias than the other methods and similar variance, indicating more consistent segmentation results across scanners. When training on the Haste-Odd images, *Atlas MV*, *SVM Atlas*, and *SVM FST* showed similar bias and variance. *SVM* gave a smaller bias, but a much larger variance than the other methods.

Note that the mean volume was much larger when training on the Haste-Odd images than on the T1 images (4000 versus 3000 mm^3). This is partly a result of the manual segmentations, which are on average about 15% bigger in the Haste-Odd images than in the T1 images and partly a result of the larger voxel sizes for the Haste-Odd images.

5.3.3 Influence of k and N

Figure 5.4 shows the influence of k and N on the performance of our FST for the two experiments on the HarP data and the experiment on the RSS data. As can be seen from the figure, the influence of both k and N on the performance is very small; the difference in average Dice between the worst and the best k and N is only 0.6%. $N = 1$, $k = 10$ seems the overall



(a) Atlas MV (b) SVM Atlas (c) SVM (d) SVM FST

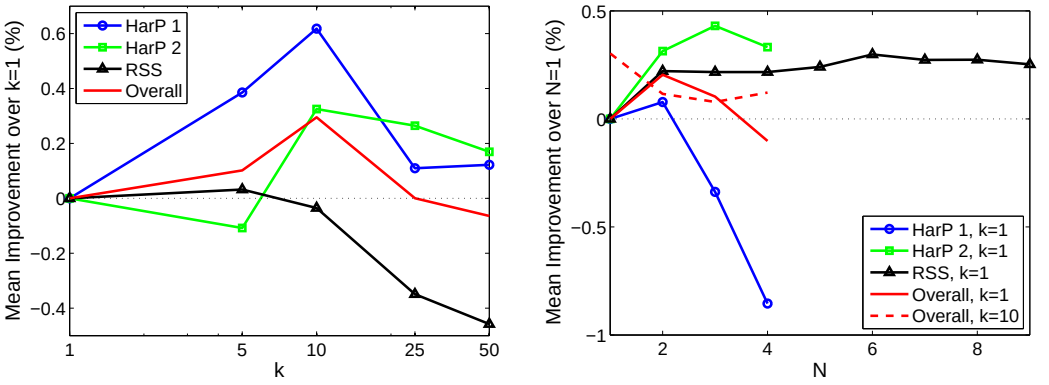
(e) Atlas MV (f) SVM Atlas (g) SVM (h) SVM FST

Figure 5.3: *Bland-Altman plots for Atlas MV, SVM Atlas, SVM, and SVM FST ($N = 8$) in RSS Rescan: reproducibility on the hippocampus segmentation of 9 rescan images in the RSS dataset. (a)-(d): trained on T1 images, (e)-(h): trained on Haste-Odd images. Each sample is one hippocampus (left or right).*

best choice, performing significantly better than the other values for k and non-significantly different from other values for N.

5.3.4 Comparison with State-of-the-Art Methods

Table 5.3 shows the performance of *STAPLE*, *Fusion*, and *Fusion FST Intensity* on the HarP1, HarP2, and RSS dataset. *STAPLE*, which uses only atlas information, performed similar to the other two methods that use only atlas information, *Atlas MV* and *SVM Atlas*. When appearance differences are small (the HarP datasets), *Fusion* greatly improved performance compared to both *STAPLE* and *SVM*. *Fusion* therefore seems to use a framework that is better capable of handling small appearance differences than the baseline *SVM*. When differences are large however (the RSS dataset), performance of *Fusion* dropped dramatically (much more than *SVM*). Using an FST to transform image intensity before using Fusion, as in *Fusion FST In-*



(a) Influence of k (b) Influence of k

Figure 5.4: *Influence on the performance of the number of neighbours, k and the number of source-target images, N. Figure (a) shows the influence of k for $N = 1$; Figure (b) shows the influence of N for $k = 1$ and $k = 10$ (only average over the three datasets). All results are shown as improvement in Dice over the performance with $k = 1$, $N = 1$.*

Table 5.3: *Mean Dice overlap of our method compared to state-of-the-art methods on 1) the HarP dataset trained on images from the same site other scanner than the test image, 2) the HarP dataset with multi-atlas probabilities determined from all HarP images except for the ones from the test scanner, 3) the RSS dataset. The best result and the results that are not statistically significantly worse, are shown in bold. N.a. = not available; this method is not available since no source-target pairs are available between sites.*

Method	HarP 1	HarP 2	RSS
STAPLE	0.718	0.827	0.799
Fusion	**0.798**	**0.884**	0.336
Fusion FST Intensity	0.773	n.a.	**0.816**

tensity greatly improved the performance in case of large differences. For small differences however, the intensity FST decreased performance. We argue that this is probably because transforming only the intensity is a suboptimal solution, as was also shown in Table 5.2 for *SVM FST Intensity*.

5.4 Conclusion and Discussion

We presented a feature-space transformation (FST) to decrease appearance differences between training and test datasets caused by the use of different scanners or scanning parameters. Our method uses unlabeled images of one or multiple subjects that have been scanned with both the training and the test scan protocol. These images, which we call source-target pairs, give a correspondence between the source and target feature spaces. Training samples are then mapped from the source feature space into the target feature space by applying the median transformation of the k closest source voxels in the feature space.

Extensive experiments on hippocampus segmentation in two very different datasets, showed the added value of appearance features over using atlas information alone. When training and test data was obtained with different scanners and protocols, our FST showed to be beneficial compared to standard intensity normalization by range matching. In the first dataset, where differences between training and test images were only small, the FST improved the performance of an SVM classifier on atlas and appearance features from a mean Dice of 0.74 to 0.75 when few atlases were used and from 0.79 to 0.84 when many atlases were used. In the second dataset, where differences between training and test images were relatively big, our FST improved the mean Dice from 0.73 to 0.80. On this dataset, we also showed that the FST can improve the reproducibility across scanners, by decreasing the bias between segmentations of images from different scanners.

We also compared with multiple other methods for hippocampus segmentation: a multi-atlas registration with majority vote, atlas fusion by STAPLE ([112]), and the patch-based atlas fusion method of [110]. Note that majority vote and STAPLE make a decision based only on atlas registration, while patch-based fusion methods, just like the used SVM classifier, incorporate appearance information. Patch-based atlas fusion clearly outperformed the baseline SVM (without FST) in case of small differences between train and test data, but decreased performance much more in case of large differences. Overall, we think patch-based atlas fusion is a better framework for atlas-based hippocampus segmentation than the baseline SVM, in case of training and test data from the same scanner, or when differences between images from different scanners are small. We think that the patch-based fusion makes better use of the atlas information than the SVM by combining appearance information of every training sample (voxel) with the atlas prior of its image. The SVM on the other hand, combines the appearance information of training samples with the atlas prior of all images together.

The SVM therefore gives all atlases the same weight, while the patch-based fusion gives large weights only to the atlases with most similar appearance. Weighting training images according to similarity in appearance with the test image can give great increase in performance when training and test data is heterogeneous ([104]). However, when appearance information is misleading, as in datasets with large differences between training and test data, patch-based fusion deteriorates more, because of this effect. We think that the presented FST can solve this problem, by transforming the representation of training samples to that of test samples. We also experimented with a poor man's implementation of such an FST for patch-based fusion, by transforming all training voxels in the feature space of the SVM FST, generating a transformed intensity image, and feeding this image into the patch-based fusion. This procedure greatly improved performance of patch-based fusion in the RSS dataset, but decreased performance on the HarP dataset. However, we think that performing an FST in the patch feature space used in patch-based fusion, rather than transforming only the intensity, would solve this problem. We namely showed for the SVM that transforming all features with the FST works much better than transforming only intensity.

Our experiments indicated that an FST based on a single source-target pair performed as good as an FST based on multiple pairs. Thus, to allow transferring of models to a new scanner, only a single subject needs to be scanned with both scanners. We also studied the influence of the FST parameter k, the number of source neighbors in the feature space. The FST seems to be relatively insensitive to the chosen value for k. $k = 10$ overall slightly outperformed smaller and larger values. If the training data consists of images from multiple scanners, the optimal value for k can be chosen in cross validation. Otherwise, since differences were small, we would advise to choose k small (e.g. $k = 1$), since it is computationally advantageous.

Our approach is inspired by the patch-based image-synthesis techniques of e.g. [49, 79], which aim to make source images look as if they were imaged with a different scanning protocol. These methods extract source-target patches from source and target scans of the same subject and then adapt the intensities of a new image by splitting it up into patches and determining the closest source patch. In contrast, our FST performs a transformation in the higher-dimensional feature space that is used for the classification. The added value of transforming all features was shown by comparing to a method that used our FST on the intensity feature only, after which the other features were calculated from the transformed intensity image. Transforming intensity alone has the disadvantage that derived features in the training data can appear very different from the (non-transformed) features in the test data, due to e.g. noise and discontinuities in the mapping function that normalizes the image. Directly calcu-

lating the transformation in the feature space, as in our FST, produces a transformation that is smooth in feature space, which results in a more robust transformation and better classification than calculating a transformation on image intensity alone. This is reflected by the much better performance of the FST on all features over the FST on intensity alone in all experiments.

The experiments in this paper indicate that a single image pair is sufficient to determine an FST. In single-site studies such as the RSS ([51]), rescans are often already made in order to check for reproducibility and to eliminate scanning problems. For multi-site studies however, rescan data may not available. However, applying the presented method to source-target images of different subjects is unlikely to work, since the subjects' anatomy will be too different to obtain correspondence from registration. Investigating how to obtain an FST from images of different subjects would be an interesting direction for further research.

In the presented experiments, we focused on rotationally invariant Gaussian-scale-space features and on cases where the same features are extracted for source and target data. However, the presented FST is more widely applicable. We believe our method has two restrictions here. Firstly, the unlabeled source-target images should be representative for the training data. Problems may arise if, for example, training data contains structures that are not observed in source-target data (such as tumors), for it would not be possible to learn the proper transformation. Secondly, there should be a one-to-one mapping between classes in source and target data. If voxels from two different classes have the same appearance in source images, but have different appearances in target images, the FST will not be able to learn a proper mapping. In other cases however, we believe the presented FST should be able to learn the proper transformation from the source-target images.

We believe that the presented method is very useful for machine-learning based segmentation of medical images that have been obtained with different scanners or scanning protocols. The experiments in this paper were all on hippocampus segmentation. However, the presented FST can be used for all supervised image segmentation tasks where the registration of two images of the same subject can give an approximate correspondence, such as brain-tissue segmentation, white-matter lesion segmentation, and segmentation of other brain structures than the hippocampus. This way, we think that our method can aid the applicability of many supervised-segmentation methods to different datasets and eliminate the requirement of same-scanner labeled training data.

Chapter 6

Transfer Learning for Image Segmentation by Combining Image Weighting and Kernel Learning

Annegreet van Opbroek, Hakim C. Achterberg, Meike W. Vernooij, & Marleen de Bruijne.
Submitted.

Many medical-image segmentation methods are based on supervised classification of voxels. Such methods generally perform well when provided with a training set that is representative of the test images to segment. However, problems may arise when training and test data follow different distributions, for example due to differences in scanners, scanning protocols, or patient groups. Under such conditions, weighting training images according to distribution similarity has been shown to greatly improve performance. However, this assumes that part of the training data is representative of the test data; it does not make unrepresentative data more similar.
We therefore investigate kernel learning as a way to reduce differences between training and test data and explore the added value of kernel learning for image weighting. We also propose a new image-weighting method that minimizes maximum mean discrepancy (MMD) between training and test data, which enables the joint optimization of image weights and kernel. Experiments on brain-tissue, white-matter-lesion, and hippocampus segmentation show that both kernel learning and image weighting, when used separately, greatly improve performance on heterogeneous data. Here, MMD weighting obtains similar performance to previously proposed image-weighting methods. Combining image weighting and kernel learning, optimized either individually or jointly, can give an additional improvement in performance.

6.1 Introduction

The segmentation of biomedical images into the various tissues and structures forms a crucial step for both medical research and clinical practice. Automatic segmentation is important because manually segmenting three-dimensional images is very time consuming and prone to inter- and intra-observer variability. Many automatic segmentation methods are based on voxelwise supervised classification. Here, per voxel a decision is made as to the class it belongs to (which tissue or structure) by a classifier trained on a manually annotated training set. In brain MRI, such methods have for example been applied to whole-brain segmentation [50, 57], brain-tissue segmentation [24, 64, 67], white-matter-lesion segmentation [10, 25, 36], and brain-structure segmentation [28, 67, 95].

However, a disadvantage of supervised-learning methods is a deterioration in performance in case of certain differences between training and test data, such as use of different scanners, imaging protocols, or differences in patient groups. Human observers can easily adapt to such differences. Supervised-learning methods however, can struggle with differences between images, because they often result in a difference between the distributions of training and test samples in the feature space. Transfer-learning[1] methods are designed to handle certain differences between training and test data, including differences in sample distributions [71]. Many transfer-learning methods that have so far been presented in medical-image segmentation are based on weighting training samples. This can be done by weighting individual samples (voxels) [39, 97, 102] or complete training images [16, 104]. Compared to sample weighting, image weighting has the advantage of requiring no labeled training data from the test scanner to handle differences between scanners.

These previously presented weighting methods only weight training samples *as is*, no steps are taken to reduce differences in the representation (the feature values) of training and test samples. An image-weighting method can only either use a training image as is (give it a positive weight), or not use it (give it a weight of zero). We therefore propose to combine image weighting with a feature-representation transfer step that 1) makes the data distributions more similar between training and test data and 2) separates the different classes as well as possible.

So far, few works have investigated explicit feature-representation transfer for medical-image segmentation. In machine learning and computer vision however, various methods have been developed. Many of them seek a linear transformation that minimizes distribution differences between training and test samples [71]. Since linear transformations are often not enough to overcome differences between datasets, transformations are often performed in a high- (possibly infinitely-) dimensional kernel space (see for example the metric-learning methods of [59, 80]). Alternatively, one may directly learn a kernel that reduces distribution differences. Pan et al. [70] for example, propose an unsupervised framework to learn a kernel that makes training and test distributions more similar by minimizing maximum mean discrepancy (MMD) between training and test samples. However, since the method is unsupervised, the learned kernel is not optimized for classification. Duan et al. [29] present a supervised method that learns a kernel that minimizes MMD between training and test samples while simultaneously minimizing a notion of classification error on some labeled test samples in the learned kernel space. Unfortunately, these two methods, as well as most kernel-learning and metric-

[1] sometimes called *domain adaptation*, although some give different definitions for the two terms

learning methods developed so far, are computationally difficult to train on many samples (e.g. more than 1 000). In voxelwise classification however, many more samples are available for training and using them is likely to result in a better performance. The image-weighting method of [104] for example, uses 50 000 samples *per image*.

In this paper, we propose two efficient kernel-learning methods that can be used in conjunction with image weighting. Firstly, we discuss a multiple-kernel-learning (MKL) method that minimizes within-class distances and maximizes between-class distances. Secondly, we use this MKL method in the framework of Duan et al. [29], which adds an MMD term. This results in a kernel space that is suitable for classification and additionally makes training and test distributions more similar. Contrary to the methods of Pan et al. [70] and Duan et al. [29], the proposed methods can be trained on ten thousands of training samples per image. We investigate whether these kernel-learning methods result in a better segmentation compared to using a standard Gaussian kernel. We also investigate whether these learned kernels improve performance when using image weighting. Further, we show that the MMD measure can be used to determine not only the kernel, but also the image weights. This way, image weights and kernel can be optimized jointly, which facilitates the optimization.

First, in Section 6.2.2, we describe three image-weighting methods: two weighting methods of [104], which minimize the Kullback-Leibler (KL) divergence and Bhatacharryya Distance (BD), and a new image-weighting method that minimizes MMD. In Section 6.2.3, we discuss the two MKL methods used: centered kernel alignment [20], and the MMD MKL framework of Duan et al. [29]. We also show in Section 6.2.3.2 how MMD image weighting and MMD MKL can be integrated in a joint optimization framework. We investigated the performance of image weighting, MKL, and the combination when training and testing on data from different datasets, which were acquired with different scanners and scanning protocols. Three medical-image-segmentation tasks were studied: brain-tissue segmentation, white-matter-lesion segmentation, and hippocampus segmentation.

6.2 Methods

Image weighting is performed similar to Van Opbroek et al. [104], where each training image is given a weight based on minimizing a distance measure between the probability density functions (PDFs) of the training images and the test image to segment. Next, a training set

is assembled based on the training images and their weights and a support-vector-machine (SVM) classifier [21] is trained. Kernel learning is performed for each test image individually and followed by classification with a kernel SVM [82].

6.2.1 Notation

We denote column vectors with bold small letters, e.g. $\boldsymbol{x}$, matrices with capital letters, e.g. X, and scalar values with small letters, e.g. m, θ. An exception is made for ϕ, which denotes a mapping into kernel space and K, which denotes a kernel ($K(\boldsymbol{x},\boldsymbol{y}) = \phi(\boldsymbol{x})^T\phi(\boldsymbol{y})$) by means of the kernel trick). Indices of vectors and matrices are denoted with subscripts, e.g. $\boldsymbol{x}_i$, $M_{p,q}$. Superscripts are used for naming, e.g. n^{te} for the number of test samples and n^{tr} for the number of training samples per image.

A total of m training images are used, which may have different scanning characteristics. Every training image m_i provides n^{tr} randomly sampled training samples (voxels) $\boldsymbol{x}_j^{m_i}$ ($j = 1,2,\ldots,n^{\text{tr}}$), where $\boldsymbol{x}_j^{m_i} \in \mathbb{R}^n$ denotes a vector containing a value for each of the n features. The $m \cdot n^{\text{tr}} \times n$ matrix of all training samples is denoted by X^{tr}. Each sample $\boldsymbol{x}_j^{m_i}$ has a label $y_j^{m_i} \in \mathbb{N}$. n^{te} test samples (voxels) $\boldsymbol{x}_j^{\text{te}}$, ($j = 1,2,\ldots,n^{\text{te}}$) are acquired from the test image, for which we predict the label y_j^{te}. The $n^{\text{te}} \times n$ matrix of test samples is denoted with X^{te}. The training samples from image m_i follow the distribution $P_{m_i}(\boldsymbol{x})$ and the test samples follow distribution $P^{\text{te}}(\boldsymbol{x})$. Based on these distributions (or PDFs), m_i is given a weight w_{m_i}, resulting in a weight vector for all training images, $\boldsymbol{w} = [w_1, w_2, \ldots, w_m]^T$. These weights are non-negative and normalized such that $\sum_{m_i=1}^{m} w_{m_i} = |\boldsymbol{w}| = 1$.

6.2.2 Image Weighting

Giving each training image m_i a weight w_i results in a total training PDF that is a weighted sum of each of the individual training PDFs:

$$P^{\text{tr}}(\boldsymbol{x}) = \sum_{m_i}^{m} w_{m_i} P_{m_i}(\boldsymbol{x}). \tag{6.1}$$

The optimal weights $\boldsymbol{w}^*$ are chosen by minimizing a convex distance criterion between this total training PDF and the test PDF:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \mathrm{DIST}_{\boldsymbol{w}}(P^{\mathrm{tr}}, P^{\mathrm{te}}). \tag{6.2}$$

6.2.2.1 PDF Weighting

The method presented by Van Opbroek et al. [104] first explicitly estimates the PDFs P_{m_i} (and, depending on the distance function, also P^{te}) by kernel density estimation and then uses a distance function on the estimated PDFs. These PDFs are approximated from the samples by kernel density estimation with a Gaussian kernel where the kernel parameter σ^S is determined with Silverman's rule [85]:

$$\sigma^S = \left(\frac{4}{n+2}\right)^{\frac{1}{n+4}} n^{\mathrm{tr}\,\frac{-1}{n+4}} \sigma^{\mathrm{tr}}. \tag{6.3}$$

Here, σ^{tr} equals the standard deviation of the training samples, averaged over all features. The chosen σ^S minimizes the Mean Integrated Square Error between the actual and the estimated PDF for a multivariate Gaussian kernel [85].

There are various methods to measure the distance between training and test PDFs. One of them, as presented in [104], is the Kullback-Leibler divergence (KL):

$$\begin{aligned} \mathrm{KL}(P^{\mathrm{te}}||P^{\mathrm{tr}}) &= \int_{\mathcal{D}} P^{\mathrm{te}}(\boldsymbol{x}) \log\left(\frac{P^{\mathrm{te}}(\boldsymbol{x})}{P^{\mathrm{tr}}(\boldsymbol{x})}\right) d\boldsymbol{x} && (6.4)\\ &\approx \frac{1}{n^{\mathrm{te}}} \sum_{j=1}^{n^{\mathrm{te}}} \log P^{\mathrm{te}}(\boldsymbol{x}_j^{\mathrm{te}}) && (6.5)\\ &\quad -\frac{1}{n^{\mathrm{te}}} \sum_{j=1}^{n^{\mathrm{te}}} \log\Big(\sum_{m_i=1}^{m} w_{m_i} P_{m_i}(\boldsymbol{x}_j^{\mathrm{te}})\Big), \end{aligned}$$

where $\mathcal{D}$ is the domain of P^{te} and P^{tr}.

Note that the first term of Equation 6.5 does not depend on $\boldsymbol{w}$. So the optimal weights $\boldsymbol{w}$ are determined by maximizing $\sum_{j=1}^{n^{\text{te}}} \log\left(\sum_{m_i=1}^{m} w_{m_i} P_{m_i}(\boldsymbol{x}_j^{\text{te}})\right)$ under the constraints $\boldsymbol{w} \geq \mathbf{0}$ and $|\boldsymbol{w}| = 1$.

A second method to measure distances between PDFs that performed well in [104] is the Bhattacharyya distance (BD):

$$\begin{aligned} \text{BD}(P^{\text{te}}, P^{\text{tr}}) &= -\log \int_{\mathcal{D}} \sqrt{P^{\text{te}}(\boldsymbol{x}) P^{\text{tr}}(\boldsymbol{x})}\, d\boldsymbol{x} && (6.6) \\ &\approx -\log\Big(\frac{1}{n^{\text{tr}}} \sum_{\boldsymbol{x}_i \in \mathcal{D}} \sqrt{P^{\text{te}}(\boldsymbol{x}_i)} \\ &\qquad \sqrt{\sum_{m_i=1}^{m} w_{m_i} P_{m_i}(\boldsymbol{x}_i)}\Big), && (6.7) \end{aligned}$$

where the $\boldsymbol{x}_i$ are randomly drawn from $\mathcal{D}$. Similar to the KL, the criterion in Equation 6.7 is minimized for $\boldsymbol{w}$ subject to the constraints $\boldsymbol{w} \geq \mathbf{0}$ and $|\boldsymbol{w}| = 1$.

6.2.2.2 MMD Weighting

We propose a new image weighting method based on maximum mean discrepancy (MMD) minimization. This method can determine the image weights immediately from the samples and therefore does not require a PDF estimation step. MMD image weighting is similar to the MMD sample-weighting method of Huang et al. [47], but weights groups of samples (images) instead of separate samples. The image weights $\boldsymbol{w}$ are determined by minimizing the MMD between all training samples X^{tr} and test samples X^{te}. The MMD between two datasets X and Y is defined as the distance between the means of the samples $\boldsymbol{x} \in X$ and $\boldsymbol{y} \in Y$ in a kernel space ϕ:

$$\text{MMD}(X, Y) = ||\frac{1}{n^X}\sum_{i=1}^{n^X}\phi(\boldsymbol{x}_i) - \frac{1}{n^Y}\sum_{j=1}^{n^Y}\phi(\boldsymbol{y}_j)||^2, \tag{6.8}$$

where n^X and n^Y are the number of samples $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively.

The image weights are determined by minimizing the MMD between the training set X^{tr}, which are given a weight per image and the test set X^{te}:

$$\begin{aligned}\text{MMD}(X^{\text{tr}}, X^{\text{te}}) = \\ = ||\frac{1}{n^{\text{tr}}}\sum_{m_i=1}^{m} w_{m_i}\sum_{j=1}^{n^{\text{tr}}}\phi(\boldsymbol{x}_j^{m_i}) - \frac{1}{n^{\text{te}}}\sum_{i=1}^{n^{\text{te}}}\phi(\boldsymbol{x}_i^{\text{te}})||^2 \\ = \frac{1}{n^{\text{tr}2}}\boldsymbol{w}^T K^{\text{tr,tr}}\boldsymbol{w} - \frac{2}{n^{\text{tr}}n^{\text{te}}}\boldsymbol{w}^T\boldsymbol{k}^{\text{tr,te}} + \frac{1}{n^{\text{te}2}}k^{\text{te,te}}.\end{aligned} \tag{6.9}$$

Here, $K^{\text{tr,tr}}$ is an $m \times m$ matrix of inner products between training images:

$$K_{p,q}^{\text{tr,tr}} = \sum_{i=1}^{n^{\text{tr}}}\sum_{j=1}^{n^{\text{tr}}}\phi(\boldsymbol{x}_i^p)^T\phi(\boldsymbol{x}_j^q) = \sum_{i=1}^{n^{\text{tr}}}\sum_{j=1}^{n^{\text{tr}}}K(\boldsymbol{x}_i^p, \boldsymbol{x}_j^q). \tag{6.10}$$

Similarly, $\boldsymbol{k}^{\text{tr,te}}$ denotes a vector of length m of inner products between each of the training images and the test image:

$$\boldsymbol{k}_p^{\text{tr,te}} = \sum_{i=1}^{n^{\text{tr}}}\sum_{j=1}^{n^{\text{te}}}\phi(\boldsymbol{x}_i^p)^T\phi(\boldsymbol{x}_j^{\text{te}}) = \sum_{i=1}^{n^{\text{tr}}}\sum_{j=1}^{n^{\text{te}}}K(\boldsymbol{x}_i^p, \boldsymbol{x}_j^{\text{te}}). \tag{6.11}$$

Lastly,

$$k^{\text{te,te}} = \sum_{i=1}^{n^{\text{te}}}\sum_{j=1}^{n^{\text{te}}}\phi(\boldsymbol{x}_i^{\text{te}})^T\phi(\boldsymbol{x}_j^{\text{te}}) = \sum_{i=1}^{n^{\text{te}}}\sum_{j=1}^{n^{\text{te}}}K(\boldsymbol{x}_i^{\text{te}}, \boldsymbol{x}_j^{\text{te}}) \tag{6.12}$$

is a single value giving the inner product of the test samples.

Note that Equation 6.9 can be written as

$$\mathrm{MMD}(X^{\mathrm{tr}}, X^{\mathrm{te}}) = \boldsymbol{\omega}^T M \boldsymbol{\omega}, \tag{6.13}$$

where $\boldsymbol{\omega}^T = [\boldsymbol{w}^T, 1]$ and

$$M = \begin{bmatrix} \frac{1}{n^{\mathrm{tr}2}} K^{\mathrm{tr,tr}} & -\frac{1}{n^{\mathrm{tr}} n^{\mathrm{te}}} \boldsymbol{k}^{\mathrm{tr,te}} \\ -\frac{1}{n^{\mathrm{tr}} n^{\mathrm{te}}} \boldsymbol{k}^{\mathrm{tr,te}T} & \frac{1}{n^{\mathrm{te}2}} k^{\mathrm{te,te}} \end{bmatrix}. \tag{6.14}$$

Given a kernel K, the optimal image weights can be found by minimizing Equation 6.13 for $\boldsymbol{\omega}$ under the constraints $\boldsymbol{w} \geq \mathbf{0}$, $|\boldsymbol{w}| = 1$.

6.2.3 Kernel Learning

We aim to find a kernel space that reduces differences between datasets. When searching for a kernel matrix, we would need to pose constraints to make sure that the found matrix is symmetric and positive semidefinite, in order for it to be a kernel (because of Mercer's theorem). A relatively easy way to find a kernel matrix is with multiple kernel learning (MKL), where we start with a set of n^k pre-defined base kernels K_k^{b} ($k = 1, 2, \ldots, n^k$). The searched kernel is determined as the optimal linear combination of the base kernels:

$$K = \sum_{k=1}^{n^k} v_k K_k^{\mathrm{b}}, \tag{6.15}$$

where the kernel weights $\boldsymbol{v} = [v_1, v_2, \ldots, v_{n_k}]^T \geq \mathbf{0}$. Since K is a linear combination of kernels, it is guaranteed to be a kernel as well.

We present two kernel-learning methods: Centered Kernel Alignment [20] in Section 6.2.3.1 and the MMD kernel-learning method that is similar to the method of Duan et al. [29] in Section 6.2.3.2. This method can be efficiently combined with MMD image weighting so that image and kernel weights can be jointly optimized.

6.2.3.1 Centered Kernel Alignment

Cortes et al. [20] presented a kernel-learning method that optimizes the kernel weights v_k in Equation 6.15 by maximizing centered kernel alignment (CKA) between the learned kernel K and the *ideal kernel* K^{I}. The ideal kernel equals 1 if two samples have the same class label and -1 if they have different class labels. This corresponds to a situation in the kernel space ϕ where all samples with the same class label are at the same location and samples with different class labels are always a distance of 2 apart.[2]

First, the CKA method centers the distribution of samples in kernel space for all base kernels, which sets the expectation of the samples in the kernel space to zero. After obtaining the centered base kernels K_k^{bc} $(k = 1, 2, \ldots, n^k)$, the weights $\boldsymbol{v}$ are chosen as to maximize the alignment between K and K^{I}:

$$\boldsymbol{v}^* = \arg\max_{\boldsymbol{v}} \frac{\langle \sum_{k=1}^{n^k} v_k K_k^{\mathrm{bc}}, K^{\mathrm{I}} \rangle_F}{|| \sum_{k=1}^{n^k} v_k K_k^{\mathrm{bc}} ||_F}, \tag{6.16}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius product and $|| \cdot ||_F$ denotes the Frobenius norm[3]. The expression in Equation 6.16 can be seen as the cosine of the angle between the learned kernel K and the ideal kernel K^{I} and equals one if and only if the two kernels are the same[40].

6.2.3.2 MMD Kernel Learning

Duan et al. [29] proposes to find a kernel K that minimizes 1) the distance between a (possibly weighted) training distribution and a test distribution, so that in the learned kernel space training samples appear similar to test samples, 2) some notion of classification error on the training samples, so that the learned kernel is suitable for classification. This results in the following optimization criterion:

[2] This can be seen by calculating $\mathrm{dist}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \sqrt{||\phi(\boldsymbol{x_i}) - \phi(\boldsymbol{x_j})||^2} = \sqrt{K_{i,i} - 2K_{i,j} + K_{j,j}}$.

[3] $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_F = \mathrm{Trace}(\boldsymbol{x}^T \boldsymbol{y})$ and $||\boldsymbol{x}||_F = \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle_F}$

$$K^* = \arg\min_K \theta \, \mathrm{DIST}_K(P^{\mathrm{tr}}, P^{\mathrm{te}}) + f_K(X^{\mathrm{tr}}, \boldsymbol{y}^{\mathrm{tr}}), \tag{6.17}$$

where DIST_K is a PDF distance function in the kernel space of K, f_K measures classification error in the kernel space of K, and θ is a trade-off parameter between the two terms.

For f_K, Duan et al. [29] uses either structural risk functional (used in support vector regression) or the Hinge loss (used in SVM), both of which are computationally very expensive if many training samples are used. CKA on the other hand, can be calculated very efficiently for a large number of training samples that consist of weighted subsets. We therefore use the CKA value, multiplied by -1, since CKA is maximized but Equation 6.17 is minimized. Similar to Duan et al. [29], we use the MMD as the distance function DIST_K in Equation 6.17.

From Equation 6.13 we can see how the MMD between training and test distributions can be calculated when the training images are weighted (by setting $\boldsymbol{\omega} = [\boldsymbol{w}^T, 1]^T$) or when no image weights are used (by setting $\boldsymbol{\omega} = [\frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m}, 1]^T$). Since the searched kernel K consists of a linear combination of base kernels, as in Equation 6.15, we construct a matrix M per base kernel, M_k, so that

$$\mathrm{MMD}(X^{\mathrm{tr}}, X^{\mathrm{te}}) = \sum_{k=1}^{n^k} v_k \boldsymbol{\omega}^T M_k \boldsymbol{\omega}. \tag{6.18}$$

When combining MMD kernel learning and MMD image weighting in this framework, we optimize

$$[\boldsymbol{w}^*, \boldsymbol{v}^*] = \arg\min_{\boldsymbol{w},\boldsymbol{v}} \theta \, \mathrm{MMD}_{\boldsymbol{w},\boldsymbol{v}}(P^{\mathrm{tr}}, P^{\mathrm{te}}) - \mathrm{CKA}_{\boldsymbol{v}}(X^{\mathrm{tr}}, \boldsymbol{y}^{\mathrm{tr}}). \tag{6.19}$$

For MMD kernel learning without MMD image weighting, Equation 6.19 is optimized for the kernel weights $\boldsymbol{v}$ only.

6.2.3.3 Implementation

Three issues could come up when using the MMD term for MKL. First of all, one may encounter numerical problems when calculating the MMD for certain base kernels where $K^{\mathrm{b}}_{p,q} \approx 1$ for all p, q, which would (wrongly) lead to an MMD of zero (for example, when K^{b} is a Gaussian kernel with a very small value for the kernel parameter γ[4]).

In order to solve this issue, we subtract from each M_k the $(m+1) \times (m+1)$ matrix

$$a_k \begin{bmatrix} \mathbf{1}\mathbf{1}^T & -\mathbf{1} \\ -\mathbf{1}^T & 1 \end{bmatrix}, \tag{6.20}$$

where we chose a_k as the average off-diagonal value of K^{b}. Note that, since $|\boldsymbol{w}| = 1$,

$$[\boldsymbol{w}^T, 1] \begin{bmatrix} \mathbf{1}\mathbf{1}^T & -\mathbf{1} \\ -\mathbf{1}^T & 1, \end{bmatrix} [\boldsymbol{w}^T, 1]^T = [1, 1] \begin{bmatrix} 1 & -1 \\ -1 & 1, \end{bmatrix} [1, 1]^T = 0, \tag{6.21}$$

so the subtraction of Equation 6.20 does not change the optimum of Equation 6.19 for any value a_k.

Secondly, when learning the kernel weights v that minimize Equation 6.17, the optimization might be biased towards base kernels that shrink all distances between samples, since this would decrease the MMD term. To overcome this problem, we added a constraint to the optimization of Equations 6.17 (only kernel learning) and 6.19 (joint kernel learning and image weighting) that normalizes the training samples to unit variance in the learned kernel space ϕ.

The variance of the training samples in kernel space ϕ equals

$$v_k \quad = \quad \frac{1}{m} \sum_{m_i=1}^{m} \frac{1}{n^{\mathrm{tr}}} \sum_{j=1}^{n^{\mathrm{tr}}} [\phi(\boldsymbol{x}_j^{m_i}) - \frac{1}{n^{\mathrm{tr}}} \sum_{\ell=1}^{n^{\mathrm{tr}}} \phi(\boldsymbol{x}_\ell^{m_i})]^2 \tag{6.22}$$

[4] $K(x,y) = \exp\{-\gamma||x-y||^2\}$

$$= \frac{1}{m}\sum_{m_i=1}^{m}[\frac{1}{n^{\text{tr}}}\text{diag}(K) - \frac{1}{n^{\text{tr}2}}\sum_{j,\ell=1}^{n^{\text{tr}}} K(\boldsymbol{x}_j^{m_i}, \boldsymbol{x}_\ell^{m_i})], \tag{6.23}$$

where $\text{diag}(K)$ denotes the diagonal of K. Calculating $\boldsymbol{\nu} = [\nu_1, \nu_2, \ldots, \nu_k]^T$ and adding the constraint

$$\boldsymbol{v}^T\boldsymbol{\nu} = 1 \tag{6.24}$$

guarantees the distribution of training samples to have variance 1 in all directions in the learned kernel space. A very similar constraint was used in the transfer kernel-learning method TCA [70][5].

Thirdly, there might be an optimization problem when determining the optimal value for θ in Equation 6.19 (both with and without incorporated image weighting), since it trades off between two methods with very different magnitudes of outcomes, $\text{MMD}(X^{\text{tr}}, X^{\text{te}})$ and $\text{CKA}(X^{\text{tr}}, \boldsymbol{y}^{\text{tr}})$. To solve this, θ was determined as a multiplication factor times the MMD and CKA values for the initial setting where all images and all base kernels are weighted equally:

$$\theta = \tilde{\theta}\frac{\text{CKA}_{\boldsymbol{w}_0,v_0}(X^{\text{tr}}, \boldsymbol{y}^{\text{tr}})}{\text{MMD}_{\boldsymbol{w}_0,v_0}(X^{\text{tr}}, X^{\text{te}})}, \tag{6.25}$$

where $\boldsymbol{w}_0$ provides equals weights for all training images,v_0 provides equal weights for all base kernels, and $\tilde{\theta}$ is a multiplication factor.

6.3 Experiments

We performed experiments on voxelwise classification on three MRI brain segmentation tasks: brain-tissue segmentation, white-matter-lesion (WML) segmentation, and hippocampus segmentation. For brain-tissue segmentation, each voxel inside a manually annotated brain mask was classified as either white matter (WM), gray matter (GM), or cerebrospinal fluid (CSF).

[5] with the difference that the TCA constraint enforces not only normalization, but also a covariance of zero between different dimensions in ϕ

For WML segmentation, each voxel inside an automatically generated brain mask was classified as either WML or non-WML. For hippocampus segmentation, a region of interest (ROI) around the hippocampus was determined using multi-atlas registration, of which every voxel was classified as hippocampus or non hippocampus.

For all three applications, we used data from different datasets, which were acquired with different scanners and scanning parameters. Each image was segmented once in leave-one-dataset-out cross validation, where the training data consists of all images from different datasets than the test image.

This section describes the data, used features, and experimental setup.

6.3.1 Data

Table 6.1: *Overview of the datasets used for brain-tissue (BT), WML, and hippocampus (HC) segmentation. For hippocampus segmentation, a total of 135 scans were used, separated into 46 datasets of images that were scanned with the same scanner. The table gives the number of images in the dataset (#img), used image sequences, field strength (FS), scanner, and voxel size (in mm^3).*

Dataset	Source	#Img	Sequences	FS	Scanner	Voxel size
BT1	RSS [51]	6	T1	1.5T	GE	$0.5\times0.5\times0.8$
BT2	RSS [53]	12	HASTE-Odd	1.5T	Siemens	$1.3\times1\times1$
BT3	MRBrainS [64]	5	T1	3T	Philips	$1\times1\times3$
BT4	IBSR [116]	18	T1	1.5T	Unknown	$0.8\times0.8\times1.5$ to $1\times1\times1.5$
BT5	IBSR [116]	20	T1	1.5T	10 Siemens 10 GE	$1\times3.1\times1$
WML1	RSS [51]	20	T1,PD,FLAIR	1.5T	GE	$0.5\times0.5\times0.8$
WML2	MSL Ch. [89]	20	T1,T2,FLAIR	3T	Siemens	$0.5\times0.5\times0.5$
WML3	MSL Ch. [89]	20	T1,T2,FLAIR	3T	Siemens Allegra	$0.5\times0.5\times0.5$
HC1-27	ADNI	1-10	T1	1.5T	Various	$1\times1\times1$
HC28-46	ADNI	1-10	T1	3T	Various	$1\times1\times1$

All data used for the experiments is summarized in Table 6.1.

For brain-tissue segmentation (BT), we used a total of 61 images from 5 datasets with corresponding manual segmentations. Two datasets from the Rotterdam Scan Study (RSS) [51, 53], one from the MRBrainS Challenge [64], and two from the Internet Brain Segmentation Repository (IBSR) [116]. The images of datasets BT1, BT2, and BT3 were acquired with a single scanner, BT4 and BT5 were acquired with multiple scanners (but unknown to us which images are from same scanner). All images have a manual segmentation of the brain mask and tissues. Images from BT3 and BT4 have the cerebellum included in the brain mask, images from BT1, BT2, and BT5 do not. The images from BT2, which are Haste-Odd images, have inverted intensities compared to the T1 images from the other studies. Therefore, the voxel values in the images in BT2 were inverted prior to calculation of the features.

For WML segmentation, a total of 40 images with manual segmentations from 3 datasets were used. One from the RSS [51] and two from the MS Lesion Segmentation Challenge of MICCAI 2008 [89]. The brain extraction tool (BET) [88] was used to generate brain masks. For the calculation of features, we treated the PD images of WML1 to be the same modality as the T2 images of WML2 and WML3, since they are visually similar.

For the hippocampus experiments, we used MR images from the Harmonized Protocol (HarP)[6]. This dataset consists of 135 T1-weighted images of the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset [68][7] with manually annotated hippocampi [8]. These 135 images were scanned at 34 sites, with a total of 46 different scanners. We split these images up into datasets of images that were scanned with the same scanner. All images were rigidly registered to MNI space with $1 \times 1 \times 1$ mm^3 voxel size. We used the brain extraction tool (BET) [88] to generate brain masks.

6.3.1.1 Preprocessing

All images were corrected for intensity non-uniformity with the N4 method [94]. Next, all image intensities were normalized by rescaling the 4-96th percentile to the 0-1 interval inside the brain mask.

[6] http://www.hippocampal-protocol.net/

[7] The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). For up-to-date information, see http://www.adni-info.org.

6.3.2 Features

For each sequence and at each voxel, the following features were calculated:

- Intensity
- Intensity after convolution with a Gaussian kernel with varying σ_F
- The gradient magnitude of the intensity after convolution with a Gaussian kernel with varying σ_F
- The Laplacian of the intensity after convolution with a Gaussian kernel with varying σ_F
- Only for brain tissue: spatial features

For brain-tissue segmentation $\sigma_F = 1, 2, 4\text{mm}^3$ was chosen, for WML $\sigma_F = 0.5, 1, 2\text{mm}^3$ was chosen because of the overall smaller voxel sizes, and for hippocampus segmentation $\sigma_F = 1, 2.2,$ and 5 mm^3 was chosen, similar to [95].

For brain-tissue segmentation, spatial features were added. For each voxel, cylindrical coordinates (R, θ_F, and z) were calculated in the following way: first, every brain mask was scaled to $[-1, 1]$ in each direction and the origin O was set as the middle of the brain mask. Here, R equals the distance of the voxel to O, z indicates its position in the cranial-caudal direction, and θ_F equals the positive angle ($\in [0, \pi]$) between the line through the voxel and O and the anterior-posterior axis. For WML, no spatial features were used since lesions can appear at different locations between training and test images. For hippocampus segmentation, no spatial features were used because spatial information is encoded in the selected ROI.

This resulted in a total of 13 features for the brain-tissue experiments, 30 features for the WML experiments, and 10 features for the hippocampus experiments. These features were used for both the kernel learning and the image weighting.

For all applications, all features were normalized to zero mean, unit variance per image. No feature reduction was used.

Table 6.2: *Overview of compared methods. "Kernel" indicates the used kernel-learning method (Gaussian or MKL with CKA maximization or MMD minimization). "Weighting" indicates the used image weighting method (equal weighting or weighting determined by KL, BD, or MMD minimization). "Optimization" indicates what was optimized (the kernel K and/or the image weights $\boldsymbol{w}$) and, for methods that use both kernel learning and image weighting, whether the optimization was performed separately or jointly.*

Cat.	Method	Kernel	Weighting	Optimization
0	K_0 W_0	Gaussian	Equal	-
1	K_0 W_{KL}	Gaussian	KL	$\boldsymbol{w}$
1	K_0 W_{BD}	Gaussian	BD	$\boldsymbol{w}$
1	K_0 W_{MMD}	Gaussian	MMD	$\boldsymbol{w}$
2	K_{CKA} W_0	CKA MKL	Equal	K
2	K_{MMD} W_0	MMD MKL	Equal	K
3	K_{CKA} W_{KL}	CKA MKL	KL	K, $\boldsymbol{w}$ separately
3	K_{CKA} W_{BD}	CKA MKL	BD	K, $\boldsymbol{w}$ separately
3	K_{CKA} W_{MMD}	CKA MKL	MMD	K, $\boldsymbol{w}$ separately
3	KW_{MMD}	MMD MKL	MMD	K, $\boldsymbol{w}$ jointly

6.3.3 Experimental Setup

6.3.3.1 Compared Methods

We compared the performance of 10 methods, which can be separated into four categories: 0) a baseline method that uses neither image weighting nor kernel learning; 1) three image weighting methods, KL, BD, and MMD; 2) two multiple kernel learning (MKL) methods, CKA, which corresponds to Equation 6.17 with $\theta = 0$ and MMD, which corresponds to Equation 6.17 with $\theta \neq 0$; 3) four methods that combine MKL and image weighting: CKA combined with KL, BD, and MMD image weighting, and MMD kernel learning combined with MMD image weighting, where the kernel and weight optimization was performed jointly, as in Equation 6.19. All compared methods are summarized in Table 6.2. When no MKL was used, a Gaussian kernel was used with kernel parameter γ_G determined with cross validation.

For MKL, we used a total of 60 base kernels, consisting of the 4 types of kernels used by Duan et al. [29]: Gaussian kernel, Laplacian kernel, inverse square distance kernel, and inverse

distance kernel. For each kernel type, we used 15 different values for the kernel parameter $\gamma = 10\hat{}[-8, -7, \ldots, 5, 6]$.

Performance of all classifiers was measured in terms of classification error for the brain-tissue and WML experiments. For the hippocampus experiments, performance was measured in Dice overlap [27]. In all three applications, significance of differences between two methods was measured with a paired t-test with the significance threshold at $P = 0.05$.

For the hippocampus experiments, we applied separate classifiers for the left and right hippocampus, which outperformed a joint classifier. The performance on the hippocampus segmentation is averaged over left and right hippocampus.

For all classifiers, we used an implementation in LibSVM [13]. The optimization of the kernel and image weights was performed with the interior-reflective Newton method [19]. After the optimization, kernel weights and image weights below 0.01 were set to zero.

6.3.3.2 Training and Test sets

For brain-tissue segmentation, training and test sets were composed by uniform random selection of voxels within the brain mask.

For WML segmentation, only voxels with a normalized FLAIR intensity above 0.75 were selected for training and testing, since WMLs appear bright on the FLAIR images. The threshold of 0.75 was chosen in a trade-off to discard non-WML voxels while maintaining WML voxels. As a result, most CSF voxels and some GM voxels were excluded, while almost all WM and WML voxels were maintained. Next, WML and non-WML voxels were sampled unevenly into the training and test sets, so that per dataset, 20% of the selected voxels consisted of WML voxels. This adaptation was chosen since the prior probability of a voxel being WML is so low (1.61% for WML1, 1.31% for WML2, and 0.22% for WML3) that classifiers are likely to choose to classify all voxels as non-WML voxels. Contrary to the classification, the kernel learning and image weights were determined on an evenly sampled dataset.

For hippocampus segmentation, all training images were non-rigidly registered to the test image. To select test samples, the atlases of the training images were transformed accordingly and an ROI was constructed consisting of all voxels for which at least 10% of the atlases vote it

to be hippocampus. Similarly, training samples are selected by non-rigidly registering all training images to each other and creating an ROI, of which the voxels are used as training voxels. The registrations were performed with the Elastix registration toolbox [58] by maximizing normalized mutual information, using the registration settings of [9].

For all applications, the training sets for MKL and image weighting consist of 1 000 randomly sampled voxels per image (for both the training images and the test image).

The training sets for the classifiers were constructed by first setting all weights $w < 0.01$ to zero, followed by randomly sampling 10 000 voxels from all training images together, according to the weight vector $\boldsymbol{w}$. For the unweighted classifiers, all training images were weighted equally. The test sets consisted of 10 000 randomly selected voxels per test image.

6.3.3.3 Parameter Settings

For the methods in category 0 and 2 (no MKL), the kernel parameter γ_G and the SVM parameter C needed to be set. For the CKA kernel-learning methods (both with and without image weighting), only the SVM parameter C was needed. For the MMD MKL methods (K_0W_{MMD} and KW_{MMD}) the MMD trade-off parameter θ and the SVM parameter C were needed. These parameters were optimized with leave-one-dataset-out grid search, where the optimal parameters were found by segmenting each training image by training on all other training images that were from different datasets. For computational reasons, all three parameters were tuned without image weighting.

6.4 Results

Table 6.3 shows the performance of all 10 methods on brain-tissue, WML, and hippocampus segmentation. Note that for hippocampus segmentation, the differences between the methods are much smaller than for the other two applications. This is probably because the images are much more homogeneous between datasets than for the other two applications, since ADNI aims to acquire data at different sites with similar protocols.

We will first discuss the results on image weighting (Cat. 1), followed by kernel learning

Table 6.3: *Performance of all methods on the three applications, averaged over all images of all datasets. Performance on Tissue and WML segmentation is in classification error (%), performance on hippocampus segmentation is in Dice overlap (%). The best method and all methods that are not significantly worse, are in bold.* $*$ *indicates a significant difference between* K_{CKA} W_{MMD} *and* KW_{MMD}.

Cat.	Method	Brain Tissue	WML	Hippocampus
0	K_0 W_0	19.03	8.40	84.0
1	K_0 W_{KL}	13.36	**7.01**	83.8
1	K_0 W_{BD}	14.99	9.84	**84.6**
1	K_0 W_{MMD}	14.09	8.53	84.4
2	K_{CKA} W_0	17.31	**7.58**	83.4
2	K_{MMD} W_0	17.28	**7.62**	83.5
3	K_{CKA} W_{KL}	**12.86**	**6.84**	83.9
3	K_{CKA} W_{BD}	14.80	9.92	84.1
3	K_{CKA} W_{MMD}	13.95*	8.42	84.1
3	KW_{MMD}	14.12	8.13	**84.6***

(Cat. 2), and the combination of the two (Cat. 3).

6.4.1 Image Weighting

For all three applications, the baseline method was significantly outperformed by a weighting method. Which weighting method performed best differs between applications. For WML segmentation, KL weighting performed considerably better than other weighting methods, as was also observed and discussed in [104]. This is likely caused by the very small percentage of WML voxels, which causes the weighting method to weight according to overall image similarity rather than WML similarity. Here, KL weighting probably outperforms BD and MMD weighting since it focuses more on parts of the distribution with low $P(\boldsymbol{x})$. For brain-tissue and hippocampus segmentation, the differences were smaller between the three weighting methods, where KL performed best for brain-tissue segmentation and BD for hippocampus segmentation. MMD was second best in both applications.

Figure 6.1 shows per weighting method and per application the average distribution of weights

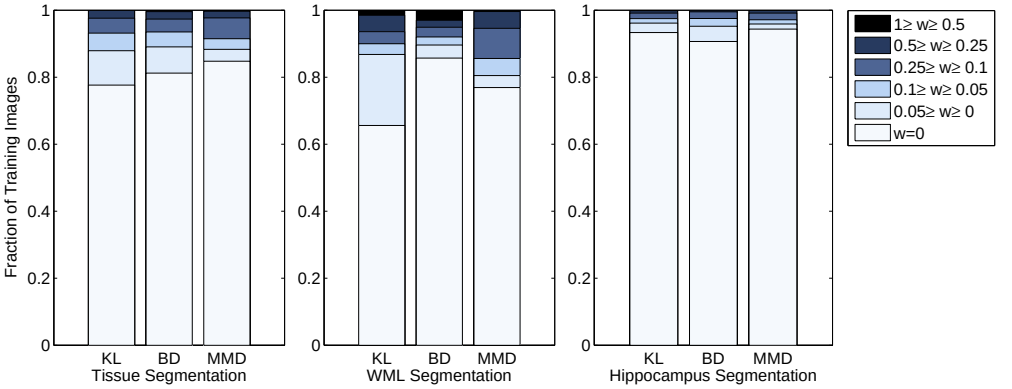


Figure 6.1: *Average distribution of weights over all training images on each of the three applications. KL, BD, and MMD respectively give the weight distributions for weighting according to W_{KL}), W_{BD}, and W_{MMD}.*

over the training images. By all three methods, the majority of training images were given a weight of zero, i.e. were not used in the training of the classifier. For the hippocampus experiments, the percentage of non-zero training images was much lower than for the other two applications, but the number of non-zero weights was similar, since many more training images (128 to 134) were used. For brain-tissue and hippocampus segmentation, the KL and BD weighting methods seem to give a bit more non-zero weights than the MMD method. For the hippocampus experiments, giving a high weight (≥ 0.3), as is sometimes done by all three weighting methods, can be unfavorable since the ROI usually consists of around 3000-5500 voxels, so no more than 30-55% of the 10 000 training samples can be obtained from a single image. For WML segmentation, there seems to be a negative correlation between the percentage of non-zero weights and the classification error. This could be explained by the fact that lesions are only a tiny percentage of the number of voxels, hence can hardly be distinguished in the PDFs. Therefore, more regularization in the form of small (non-zero) weights can be beneficial.

Figure 6.2 shows the distribution of weights for the three weighting methods when testing on the six images of the BT1 dataset. Overall, weight distributions were quite similar. All three weighting methods gave most weight to the training images of BT2, but the specific images that were given a non-zero weight differ per test image and per weighting method.

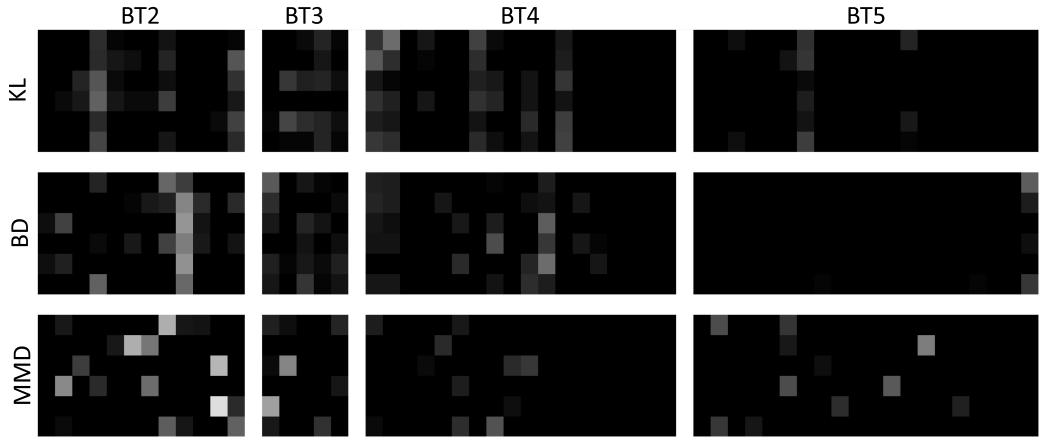
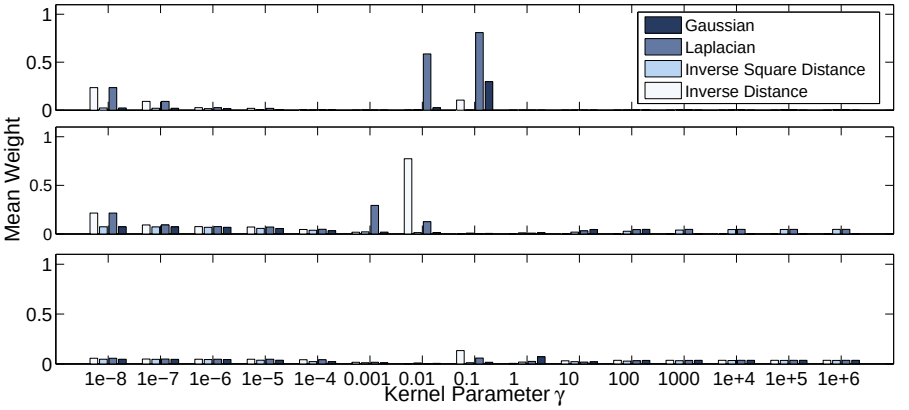


Figure 6.2: *Distribution of weights for KL, BD and MMD weighting on the six images of BT1 by training on BT2-5. Rows indicate the 6 test images, columns indicate the 55 training images, ordered by dataset. Lighter pixels indicate higher weights (highest weight is 0.43).*

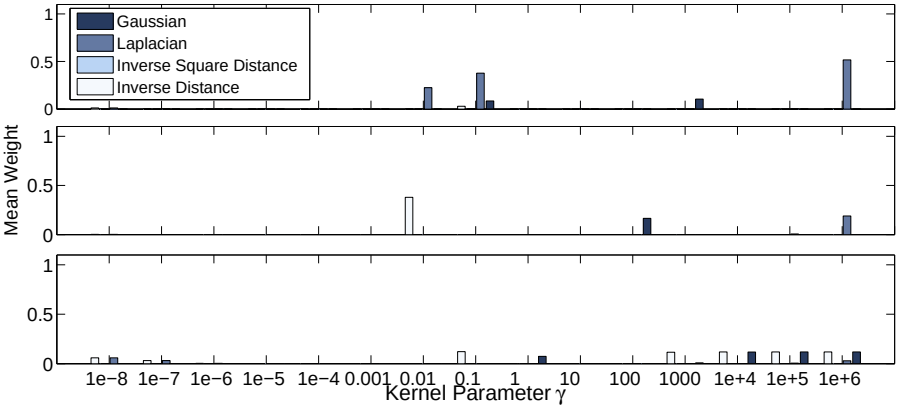
The KL and BD weighting method both find one or more training images that were given a high weight for all six test images, resulting in similar weight distributions between test images. The MMD weighting method on the other hand, gives weight distributions that differ much more between test images. This behavior difference between KL/BD and MMD was also observed in the other experiments and could indicate that MMD weights more according to specific image characteristics (such as differences between subjects), whereas KL and BD weights more according to appearance differences between scanners.

6.4.2 Kernel Learning

In brain-tissue and WML segmentation, kernel learning significantly outperformed the baseline method. For hippocampus segmentation, it performed similar to the baseline. The difference between the two kernel-learning methods (CKA and MMD) was quite small and not significantly different.



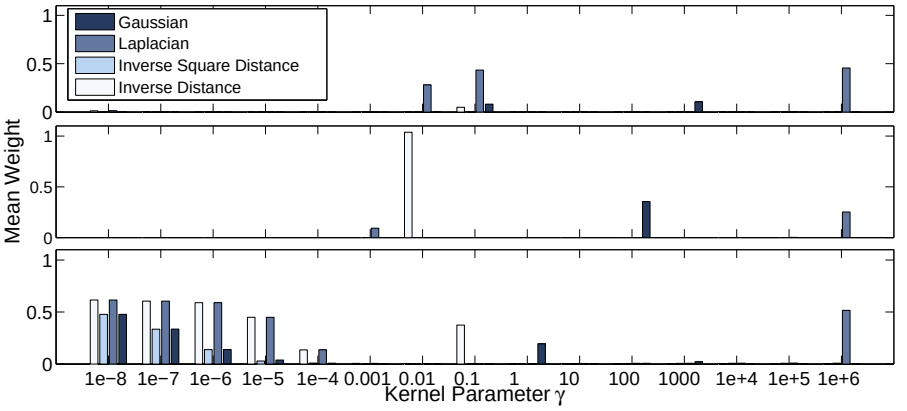
(a) K_{CKA}



(b) K_{MMD}

Figure 6.3: *Average distribution of kernel weights on the three applications (top: tissue, middle: WML, bottom: hippocampus).*

Figures 6.3(a) and (b) show the kernels that were selected by the CKA and MMD kernel-learning method respectively for each of the three applications. Although performance is similar, the distribution of kernel weights is quite different. Most kernels that are given a high



(c) KW_{MMD}

Figure 6.3: *Average distribution of kernel weights on the three applications (top: tissue, middle: WML, bottom: hippocampus) (Continued).*

weight in K_{CKA} are also given a high weight in K_{MMD}. K_{MMD} additionally gives high weights to kernels with the highest kernel parameter (10^6), i.e. kernels that focus on differences between samples that are close by in feature space. K_{CKA} on the other hand, gives an additional low weight to almost all kernels. This might be because of a premature stopping of the optimization. Although these small weights hardly influence the outcome, we set all kernel weights below 0.01 to zero to speed up the calculation of the kernels used for classification.

6.4.3 Combining Kernel Learning and Image Weighting

Image weighting generally outperforms kernel learning when used individually, especially for brain-tissue segmentation. Combining the two approaches overall gives a small additional improvement in performance, except for hippocampus segmentation, where performance was similar between image weighting with and without kernel learning.

Jointly weighting kernels and images (KW_{MMD}) gives no clear difference in performance compared to individual MMD weighting (K_{MMD}). Figures 6.3(b) and (c) show that for brain-tissue

and WML segmentation, both methods select the same base kernels, although the weights are slightly different for WML segmentation. For the hippocampus experiments, KW_{MMD} gives high weights to low-parameter kernels (i.e. focuses on differences between all samples). However, this behavior appeared only in about one third of segmented images and did not appear to significantly influence the performance. We also studied the distribution of W_{MMD} image weights when combining image weighting and MKL; these were very similar to those calculated without MKL for all three applications.

6.5 Discussion and Conclusion

Image weighting significantly improved performance over weighting all training images equally for all three studied applications: voxelwise brain-tissue, white-matter-lesion (WML), and hippocampus segmentation. This convincing benefit of image weighting corresponds to previous findings [16, 104]. For WML segmentation, which has highly unbalanced classes, Kullback-Leibler (KL) weighting performed much better than other weighting methods, which was also observed in [104]. Here, KL weighting presumably outperforms other weighting methods since it gives more importance parts of the distribution with small prior distribution $P(\boldsymbol{x})$. For brain-tissue and hippocampus segmentation, the best methods differs and differences between the three weighting methods were relatively small.

The new maximum mean discrepancy (MMD) image weighting has the advantage over KL and Bhattacharyya distance (BD) weighting that no intermediate PDF estimation is required. This is especially beneficial when many features are used, as the complexity of PDF estimation scales quadratically with the number of features. On the other hand, in the current implementation of MMD image weighting, calculation time scales quadratically ($\mathcal{O}(n^2)$) with the number of training samples used for the optimization (because of calculation of the MMD matrix M in Equation 6.14), whereas KL and BD weighting are $\mathcal{O}(n)$. It might therefore be worthwhile to investigate a method to speed up MMD calculation, such as [122], which is $\mathcal{O}(n)$.

MKL with any of the two presented methods significantly improved performance over the baseline (a Gaussian kernel) for brain-tissue and WML segmentation and performed similar to the baseline for hippocampus segmentation. For all three applications, image weighting resulted in a larger improvement than MKL, although this increase was not significant for

WML segmentation. The proposed combination of image weighting and MKL overall gave an additional improvement over image weighting alone, except for hippocampus segmentation, where it performed similarly.

We found no clear difference in performance between MKL with only the centered kernel alignment (CKA) method of Cortes et al. [20] and using and additional MMD term. Note that CKA MKL is determined on training data only, hence, finds a kernel space that discriminates between classes and generalizes well between the training datasets. Adding the MMD term should give a kernel space that additionally reduces differences between training and test distributions. We showed that the two MKL methods do indeed generate a different kernel space by selecting different kernel weights, but classification performance using the two methods was very similar. Note that for both methods there is no guarantee that the resulting kernel space and image weights are suitable for classification of the test samples, since no labeled samples from the test distribution were available for training. The additional MMD term might therefore give a kernel space that reduces distribution differences between datasets compared to CKA MKL, but has more class overlap. Adding the MMD term does have the advantage of enabling joint optimization of kernel and image weights, by using MMD image weighting. Besides being more efficient, this has the advantage of resulting in a joint optimum for kernel and image weights. On the other hand, the MMD term would require to additionally set the trade-off parameter θ (Equation 6.19). But it might be possible to use a set value (e.g. $\tilde{\theta} = 1$ in Equation 6.25), since we found performance to be stable over a large range of values for θ.

Overall, our results on a variety of different tasks convincingly show that combining kernel learning and image weighting is beneficial for across-scanner segmentation. The proposed method requires a set of training images (we used 20 to 134 training images) with different characteristics, so the image-weighting method can select the most useful images. We experimented with voxelwise classification, but the proposed methods might as well be applied to classification of super voxels or patches. It can also be applied to image-classification problems such as computer-aided diagnosis by determining image weights based on the voxel distributions. Our method is easily applicable in practice, for example in multi-center studies, since it requires no labeled data from the test distribution. As shown in [104], image weighting can be beneficial even if some manually annotated images from the test scanner are available. Although this setting was not tested in this paper, we believe the combination of kernel learning and image weighting can also be beneficial here. In this case, the labeled test data can additionally be used to ensure that the learned kernel is suitable for classification of test samples, by maximizing CKA on the test samples rather than the training samples.

To conclude, the combination of kernel learning and image weighting appears to be a promising method for supervised segmentation using heterogeneous training data different from the test dataset. The good results on different tasks indicates that the proposed methods can be applied to a wide range of medical-image-segmentation tasks.

Chapter 7

Summary and General Discussion

7.1 Summary

The aim of this thesis is to study whether transfer-learning techniques can aid automated supervised neuro-image segmentation on MRI scans with different characteristics. Many neuro-image-segmentation techniques are based on supervised learning, which assumes training and test data follow the same distribution in feature space. However, this assumption is usually violated in cases where training and test data are somewhat different, for example because of differences in scanner hardware, scan-sequence parameters, or differences between patient groups. Transfer learning comprises techniques that can cope with certain differences between training and test data. In this thesis, we studied different approaches to transfer learning and their value for supervised segmentation of MR brain images acquired in different settings.

The transfer-learning methods studied in this thesis can be divided into two categories: transfer classification and feature representation transfer. Transfer classification consist of methods that handle the differences between training and test data in the classifier. Feature-representation transfer on the other hand, consists of methods that handle these differences in the feature space, by finding a feature representation that is more similar between training and test data. As presented in this thesis, it is also possible to simultaneously use feature representation transfer and transfer classification.

7.1.1 Non-Transfer Baseline

Chapter 2 presented a regular (non-transfer) supervised classification technique for the automatic segmentation of MR brain images into cerebrospinal fluid (CSF), gray matter (GM), and white matter (WM). First, multi-atlas registration was used to segment the brain (CSF+GM+WM) from the rest of the image. Next, the different brain tissues were segmented using voxel-wise classification with a support vector machine (SVM). This classifier was trained on five labeled training images, i.e. images with expert manual annotations. For each voxel in the brain, we used three types of features: 1) intensity features from three structural MRI sequences; 2) Gaussian-scale-space features derived from these sequences; and 3) the spatial coordinates of the voxel in the image space. Gaussian-scale-space features can be seen as filtered versions of the image that incorporate information on nearby voxels. These features were added to support the classifier with information on larger structures and spatial relations within the brain. The method was applied to the dataset of the MRBrainS13 segmentation challenge, which

consists of images acquired on a single scanner. It overall gave good segmentations that were generally smooth in image space because of the addition of the Gaussian-scale-space features. Some voxels were misclassified because of a slight oversegmentation of the brain mask, due to misregistrations. The good performance of the method was supported by its second place in the MRBrainS13 challenge.

7.1.2 Transfer Classification

The non-transfer technique of Chapter 2 was used as starting point for our study of different transfer-learning techniques. Firstly, in **Chapter 3**, we applied four transfer classifiers based on multi-feature SVM classification. These classifiers all use a large amount of labeled source samples (voxels), i.e. samples from images that are scanned on different scanners or with different scan-sequence parameters than the test image to be segmented. On top of these source samples, they use a small amount of labeled target samples, i.e. samples from images that are scanned on the same scanner and with the same scanning parameters as the test image. Three of the four transfer classifiers are based on weighting source and target samples differently, where two of these classifiers additionally iterate between classification and reweighting, in different ways. The fourth classifier is based on using a classifier trained on the source samples as regularization for a classifier trained on the target samples. The performance of these transfer classifiers was evaluated on brain-tissue segmentation and white-matter-lesion segmentation. Both experiments included datasets with images obtained with different scanners and different scanning parameters. In both applications, we showed that the transfer classifiers yielded a significantly higher performance than the best non-transfer classifier when too little labeled target data is available to build a good non-transfer classifier. Additionally, we showed that different intensity-normalization techniques are helpful to decrease differences in feature distributions between scanners. Still, transfer classifiers were beneficial independent of the used normalization technique.

Chapter 4 presented a technique to weight training images without requiring labeled target data, contrary to the method of Chapter 3. This technique uses a set of training images from different studies, which are acquired with different MRI scanners and different scanning parameters than the test image. Each of the training images is given a weight and used to train a weighted SVM classifier. Here, the weight of an image is determined based on the probability density function (PDF) of its voxels in feature space. All image weights are determined jointly by minimizing the difference between the total weighted training PDF and the PDF of the test

image. We studied three different metrics for the distance between these PDFs: the Kullback-Leibler divergence (KL), the Bhattacharyya distance (BD), and the Squared Euclidean distance (SE). These three metrics resulted in slightly different image weights. The method was evaluated on three applications with data from different studies: brain-tissue segmentation, whole-brain segmentation, and white-matter-lesion segmentation. We experimented with two settings: 1) having only source images and labels available for training and 2) having labeled source and target images. For the first setting, image weighting yielded significantly better results than not weighting, for all three applications. For brain-tissue and whole-brain segmentation, weighting even significantly improved performance for the second setting. This finding indicates that image weighting can be beneficial even if labeled target images are available for training. BD weighting overall slightly outperformed KL and SE weighting for brain-tissue and whole-brain segmentation, whereas KL weighting performed best for white-matter-lesion segmentation.

7.1.3 Feature-Representation Transfer

Chapters 3 and 4 presented transfer-learning methods that account for differences between training and test datasets in the classifier, while leaving the features untouched. **Chapter 5** presented a method that minimizes differences in the feature space used for classification. It aims to learn a *feature-space transformation* (FST), a mapping from the source feature space to the target feature space, based on unlabeled images of subjects scanned with both source and target scanner. After image registration, these images provide information that can be used to derive a mapping from samples that follow the source distribution to samples with the target distribution. This mapping is then applied to samples of the labeled training images to make their distribution more similar to samples of the test image. After transformation, the training samples are used to train an SVM classifier. The proposed method was evaluated on two hippocampus-segmentation datasets, one with relatively small differences between training and test data and one with large differences. On both datasets, the proposed FST significantly outperformed a baseline method using only intensity-based normalization. We also showed that the FST could be incorporated into a patch-based atlas-fusion technique to improve its performance across scanners.

Lastly, **Chapter 6** investigated the combination of transfer classification and feature-space transfer by combining image weighting and kernel learning. For image weighting, the KL and BD proposed in Chapter 4 were used. Additionally, we proposed a new image-weighting

method based on maximum mean discrepancy (MMD) minimization that enables the joint optimization of the kernel and the image weights. For kernel learning, we proposed two methods. One method searches a kernel that improves classification between source and target data, the other method aims to find a kernel space where source and target distributions are more similar. We studied the additional value of kernel learning and image weighting separately and combined. Three applications were investigated: brain-tissue segmentation, white-matter-lesion segmentation, and hippocampus segmentation. When used individually, both image weighting and kernel learning significantly improved performance compared to the baseline non-transfer classifier. Here, the segmentation accuracy of image weighting based on MMD minimization was similar to that of the two methods from Chapter 4. When image weighting and kernel learning were combined, a small additional improvement in performance could be obtained. Joint optimization of the kernel and image weights yielded similar performance as individual optimization.

7.2 Conclusion

Table 7.1: *Overview of transfer-learning methods presented in this thesis. The table indicates for each method whether it 1) requires labeled source data, 2) requires labeled target data, 3) requires unlabeled rescan images (same subjects on source and target scanner). It also gives an indication of 4) computational load, 5) how much difference the method can handle between source and target data, and 6) performance in our experiments (*=low, **=middle, ***=high).*

Method	Chapter	Source Data	Target Data	Rescan Images	Computation Load	Data Differences	Performance
WSVM, A-SVM	3	x	x		*	*	**
KL, BD Image Weighting	4	x			**	*	***
FST	5	x		x	***	***	**
MMD Image Weighting	6	x			***	*	***
Transfer Kernel Learning	6	x			**	**	*

The results presented in this thesis show that transfer-learning methods can improve performance of supervised automated medical-image-segmentation techniques in case of training and test images from different scanners, scanning protocols, or patient groups. We showed that transfer learning can be used both in the classification step, by using a transfer classifier or image weighting, and by feature-representation transfer, by using a feature-space transformation or kernel learning. We showed that combining transfer classification and feature-representation transfer is also possible and might give an extra improvement in performance. By coping with differences between datasets, transfer learning can extend the applicability of automated supervised medical-image-segmentation methods on images with different characteristics, with potential application in both the clinic and in research.

Table 7.1 provides an overview of the transfer-learning techniques that have been presented and evaluated in the various chapters. It also summarizes the required data and computational requirements of each of the methods and the performance in our experiments.

7.3 Discussion

In this section, I discuss the main findings of our work, the assumptions and limitations, possible future directions of research, and the practical benefits of transfer learning for medical image segmentation. I will first discuss the presented methods, followed by my vision on the practical benefit of transfer learning. The first part uses the same distinction as in the rest of the thesis: transfer classification, feature-representation transfer, and the combination. I added an extra category that was not separately discussed in any of the chapters, but can be found throughout the thesis: parameters that are optimized to be robust between scanners. Section 7.3.2 on transfer classification is split up into two categories: parameter transfer and instance transfer. These are two slightly different approaches, which are distinguished in the transfer-learning overview paper of Pan and Yang [71]. The second part of the discussion, in Section 7.4 elaborates on future directions for transfer-learning methods (such as deep learning), the computational load of presented methods, and when and how to use transfer learning.

Although all experiments were conducted on MR neuro-image segmentation, most of the presented transfer-learning methods are applicable to a broader range of medical-image-segmentation tasks and even medical-image-analysis tasks such as computer-aided diagnosis.

Where applicable, I will discuss the limitations of the methods when applying them to other domains.

7.3.1 Robust Parameters

First of all, I would like to discuss presented methods that can improve classification between scanners by choosing parameters that are more robust for differences between scanners. I would not call these methods transfer-learning methods, since they use no (labeled or unlabeled) target data to adapt the framework to the test data. Nonetheless can they improve performance on images from different datasets. We presented methods that perform feature selection, feature extraction, or parameter optimization in leave-one-scanner-out cross validation on a training set of images from different scanners. This way, we aim to find features or parameters that are more robust for differences between scanners, compared to features or parameters optimized on data from a single scanner. We also proposed a feature-extraction method to find robust features by multiple kernel learning, based on maximizing centered kernel alignment (CKA) [20] on multi-scanner source data. In Chapter 6, this method was shown to improve between-scanner segmentation. Note that all these methods could in theory also be optimized on source and target data in case a sufficient amount of labeled target data is available. This could give features or parameters that are better optimized for the target data, rather than the source data. However, it may be hard to determine how much target data is sufficient.

7.3.2 Transfer Classification

7.3.2.1 Parameter Transfer

In parameter-transfer methods, source and target data use the same classification framework, with some parameters shared between source and target data and other parameters set differently [71]. These methods require a small amount of labeled target data and a larger amount of labeled source data. In this thesis, we studied only one parameter-transfer method, in Chapter 3: Adaptive SVM (A-SVM) [118]. A-SVM trains a classifier on the source samples that is used as regularization for a classifier trained on the target samples. Compared to instance-transfer methods, which simultaneously train on source and target samples, parameter trans-

fer methods train on fewer samples at the same time (only source or target). Also, source samples are used only once, independent on the number of target datasets to be segmented. Parameter-transfer methods are therefore relatively fast to train and require little memory compared to instance-transfer methods. An additional benefit of training a separate source classifier would be the possibility to share only the classifier with other parties, rather than the source data itself. This source classifier could then be fine tuned on a small amount of target data.

7.3.2.2 Instance Transfer

Instance transfer is based on weighting source data. Source samples can all be given the same weight, for regularization of a classifier on some labeled target data (similar to a parameter-transfer classifier), or different weights, according to target-data resemblance. Instance transfer is probably the most frequently used non-deep transfer-learning approach in medical image analysis. We presented sample-weighting methods in Chapter 3 and image-weighting methods in Chapters 4 and 6, which are both examples of instance transfer. Instance transfer can be used both with and without labeled target data. Both cases can handle differences in feature distributions $P(\boldsymbol{x})$ between source and target. When labeled target data is available, it is possible to also compensate for relatively small differences in labeling distributions $P(y|\boldsymbol{x})$ between source and target data.

We will first discuss the case with some labeled target data, which was presented in Chapter 3. Here, a classification framework can be trained on the labeled source and target samples. This setting is applicable to many problems, if enough labeled samples (source and target) are available. Contrary to the parameter-transfer classifier A-SVM, these instance-transfer classifiers use source and target samples simultaneously. An easy way to train a classifier on source and target data is by using the same weight for all source samples. Target samples are also all given the same weight, which is higher than that of the source samples. These weights are used in a weighted classifier, such as a weighted SVM. We referred to this method as weighted SVM (WSVM). Instance-transfer classifiers such as WSVM take the full distribution of the source data into account by using all source samples. Parameter-transfer classifiers on the other hand, use only the resulting classifier trained on the source samples, which may have been dominated by training samples that are more dissimilar to the target data and therefore less relevant. As a result, Instance-transfer classifiers might be more suitable for situations with large differences between source and target data. For example, we saw that in case of

large difference in class sizes between source and target, instance transfer was more suitable than parameter transfer. In our experiments, reweighting source or target samples as in the proposed Reweighted SVM and TrAdaBoost, makes training of the classifier much slower and yielded no additional improvement in performance compared to using fixed weights (as in WSVM). This result is in line with the work of Van Engelen et al. [97], who compared weighted and reweighted linear discriminant classification (LDC) for vessel plaque-component segmentation on MR images from different scanners. They concluded that weighted LDC improves performance over a non-transfer LDC, but reweighted LDC yields no additional improvement. I think whether reweighting could be beneficial for performance depends on the used classifier and dataset. For simple classifiers, which optimize few parameters on the data, I can imagine reweighting could be beneficial to better adapt the classifier to the target data. For more sophisticated classifiers, which are generally more time consuming to train, I would generally not recommend reweighting, as it greatly increases computation time, likely without increasing performance.

When there is no difference in labeling distributions $P(y|\boldsymbol{x})$ between source and target data[1], sample weighting can in theory completely eliminate differences between source and target distributions. In this case, no target labels are required to identify and compensate the distribution differences. For example, Goetz et al. [39] presented a sample-weighting method when training on samples from one slice and testing on a complete (3D) MR image. This resulted in a difference in the number of samples per class and a disproportionate sampling between training and test datasets, but no difference in labeling distributions. This training-test difference could therefore largely be eliminated by weighting training samples as to exactly match the weighted training distribution with the test distribution. I think such a weighting method could also be beneficial when training on subjects with different relative volumes of the tissues to segment (e.g. subjects with different anatomy, or healthy versus diseased people), since such a difference results in a relatively large difference in the number of samples per class and only small difference in sample appearance. Therefore, differences in labeling distributions might be small, depending on the used features. Multiscale features for example, might change when tissue volumes change. Depending on the used features, it might therefore be advisable to combine such sample weighting with another transfer method to overcome small differences in labeling distributions.

When training on images from different scanners or scanning parameters, there usually is a much larger difference in labeling distributions, since voxels of the same tissue have different

[1] Some people would use the term *domain adaptation* for this situation, instead of *transfer learning*.

intensity distributions between scanners. When no labeled target data is available, we proposed to use image weighting as discussed in Chapters 4 and 6. For this approach we assume that images with similar feature distribution $P(\boldsymbol{x})$ have similar labeling distributions $P(y|\boldsymbol{x})$. In Chapter 4, we showed that image weighting greatly outperformed sample weighting for segmentation across scanners, without requiring labeled target data. We also showed that image weighting can be beneficial even in a non-transfer setting, when labeled target images are available for training. This indicates that images that are generally assumed similar (images from the same dataset), also have some difference in distributions, so that weighting according to distribution similarity can still give improvement.

Cheplygina et al. [16] also showed the value of image weighting for brain-tissue segmentation on datasets from different scanners. Their method has two differences to our method. First of all, it determines the weights based on point-set distances, rather than distribution distances, which are easier to calculate since it requires no density estimation. Secondly, it determines individual distances between each training image and the test image, which requires a mapping function to convert these image distances to image weights. Determining a good mapping function can be quite tedious by requiring to optimize additional parameters. Our method circumvents such a step by determining all image weights jointly by minimizing the distance between the test PDF and the weighted training PDF. Weighting with the approach in [16] seems to perform similar to KL image weighting. I would therefore argue that preference between the different methods (KL, BD, MMD, [16]) can be based on computational efficiency, which depends on dataset characteristics such as the number of images, number of samples per image, number of features, and the ease to determine method parameters.

Image weighting has the advantage over the transfer classifiers presented in Chapter 3 that no labeled target data is needed to overcome a small difference in labeling distributions. However, image weighting is less generally applicable than classifiers such as A-SVM and WSVM for a number of reasons. First of all, image weighting requires a source dataset of multiple labeled images, preferably with different distributions, so that similar images can be found for a wide range of test distributions. Second, determining the image weights is computationally more expensive than training an A-SVM or WSVM transfer classifier. However, compared to transfer classifiers that use reweighting (such as Reweighted SVM or TrAdaBoost), image weighting might require less computational time. Lastly, in order to apply image weighting, an image must supply enough samples to reliably estimate the sample distribution or a point-set distance in feature space. I would advise to use KL and BD only when few (a couple dozen) features are used, since estimating the distribution can be both computationally expensive

and inaccurate for many features. I expect MMD image weighting and the method of [16] to be less sensitive to this problem, since they require no intermediate density estimation. However, I would advise to use few features for all proposed image-weighting methods if possible, since for all methods the number of samples required to determine the image weights scales quadratically with the number of features.

7.3.3 Feature-Representation Transfer

Feature-representation transfer aims to find a feature representation suitable for classification that additionally minimizes differences between domains. These methods can be used with or without labeled target data. We presented two approaches, both using labeled source and unlabeled target data: 1) a feature-space transformation (FST), based on unlabeled scan-rescan images from source and target scanner and 2) a transfer-kernel-learning method to learn a kernel space where source and target data is more similar.

The FST uses image registration of scan-rescan image pairs to learn a voxel mapping from the source to the target feature distribution. This method can be applied to voxelwise classification (or classification of other small parts such as small patches or supervoxels) in an application where image registration can provide a reliable voxel-to-voxel correspondence. However, for some body parts with relatively large position differences between scan sessions, scan-rescan images might not directly provide such a correspondence. This might be the case for example for the heart, lungs, or knee. For these cases, it would be advisable to investigate whether a robust mapping can be determined by non-rigid image registration. Determining a mapping from non-rigid image registration of images from different subjects would also be valuable for brain segmentation in cases where scan-rescan image pairs are not readily available (such as multi-center studies). For example, I would be interested to see whether a mapping can be learned from the non-rigid registration of images of different subjects. Alternatively, if images of different subjects give too many incorrect voxel mappings, one could think of registering the average images of a lot of source and target images, so that registration errors are averaged.

We also presented a transfer kernel-learning method. This method is more generally applicable than the FST method, since it requires no local correspondence between pairs of source and target images. This transfer kernel-learning method aims to find a kernel that is suitable for classification, by maximizing CKA of source data, and additionally minimizes distribution differences between source and target data, by minimizing MMD. In the presented set-up, it

requires labeled source data and unlabeled target data, but if target labels are available, they could be used to maximize CKA on the labeled target samples instead of, or in addition to, the source samples. In our experiments, this method performed similar to CKA kernel learning without an additional MMD term. This suggests that finding a kernel that improves distribution similarity does not necessarily improve between-scanner classification performance. However, I would expect the MMD term to be potentially beneficial in cases with other source-target differences than differences between datasets in the source data, so that CKA is not adapted to the differences observed between source and target data.

We also briefly experimented with transfer component analysis (TCA) [70]. TCA is an unsupervised method that finds a kernel space that minimizes MMD between training and test data, while simultaneously performing feature reduction to a chosen number of principal components. These results were not incorporated in any of the chapters of this thesis. TCA generally slightly improved classification performance across scanners. Performance increase was relatively small compared to the transfer-learning methods presented in this thesis. Although TCA finds a kernel space that makes source and target data more similar (by reducing distribution differences), no labels were used to optimize the learned kernel for classification. Also, only few samples (several thousand) can be used to train the kernel. These two limitations might result in only small performance improvement. Still, it overall outperformed intensity-normalization techniques such as range matching and distribution matching [69], which are also unsupervised. I think the main advantage of TCA over many intensity-normalization techniques is that it matches the distribution for all features at the same time, instead of normalizing the image intensity only.

7.3.4 Combining Methods

In theory, any of the mentioned techniques can be combined with the aim to further reduce differences between datasets. I think the most logical combination would be to combine a feature-representation transfer method and a transfer classifier (parameter- or instance-transfer method), since they operate at different stages of the classification framework and could potentially strengthen each other. For example, one can combine an FST or kernel-learning method with sample weighting. Here, the FST or kernel aims to map the features, followed by a weighted classifier that can be used to weight samples that are transformed incorrectly and increase weights of samples that are transformed correctly. For example by weighting according to distribution similarity to (labeled or unlabeled) target samples. One

could also combine an FST with image weighting in order to downweight incorrectly transformed images. Combining image weighting and sample weighting could also be an option, to first determine suitable source images and then use some labeled target samples to downweight source samples that contradict the target data.

We studied the combination of transfer kernel learning and image weighting. Image weighting performs well when provided with some training images with distributions similar to that of the test image. But image weighting is unable to make feature distributions of images more similar to the test image (it can only give a positive weight or a weight of zero to an image). A feature-representation-transfer method might solve this problem by determining a feature space where training and test data is more similar. Combining image weighting and feature representation transfer might therefore improve performance or require a smaller source dataset. We showed a modest improvement when combining image weighting with transfer kernel learning and showed how kernel and image weights can be optimized jointly. However, the used multiple-kernel-learning method (with 60 base kernels) might not be flexible enough to overcome the distribution differences between images. TCA on the other hand, which is much more flexible in choosing its kernel, seems more successful in reducing distribution differences between training and test data. I therefore think larger improvement might be obtained by using a feature-representation-transfer method that is more flexible than the proposed multiple-kernel-learning method.

7.4 Practical Benefit

The focus of this thesis is mainly methodological: to investigate transfer learning for medical image segmentation. I think that this study (together with others) shows the potential of transfer learning for supervised medical image segmentation in clinical and epidemiological research and clinical practice. Transfer learning has the potential to robustify supervised-learning methods that are currently used for segmentation of homogeneous datasets, so that larger differences between images can be handled. This way, these methods can more easily be applied to datasets with larger differences between images, such as multi-center or multi-scanner studies, or clinical data. In clinical data, differences between patients and image quality are often larger than in research, which also asks for robust (transfer-learning) methods. With such robust methods, it also becomes easier to build good segmentation methods that can be shared between institutions and applied to different datasets, either out of

the box, or after some fine tuning on a small target dataset. This way, facilitating the study of large amounts of data and decreasing the need for (expensive) labeled training data. I therefore think transfer learning can mean a big step forward in the applicability of supervised machine-learning methods for medical image analysis.

Since the aim was to study the benefits of transfer learning, I started with a relatively simple (non-transfer) segmentation method, which was presented in Chapter 2. This method consisted of a multi-feature voxelwise classification with a support-vector-machine classifier. This relatively simple baseline technique was combined with transfer-learning techniques, in order to study the added value of the proposed transfer-learning techniques. Using a relatively simple baseline gives insight in the classification framework and simplifies studying the influence of the transfer-learning technique. In practice, one might want to use a different or more extensive framework that has proven its value for non-transfer learning. For example, one could use different pre-processing, a more extensive normalization technique, the use of more image features or a deep-learning framework to determine optimal features, a different classifier, or a post-processing step. I think it would be interesting to further research the added value of transfer-learning methods for different frameworks that are considered current state of the art for single-scanner data. Especially, I think it would be interesting to study the use of transfer learning in deep learning, as deep learning is currently providing state-of-the-art results in many segmentation tasks. A number of researchers have started to investigate transfer deep learning.

7.4.1 Transfer Deep Learning

In deep learning, transfer learning is often used as term to indicate pretraining on data from a different dataset, followed by training on the target dataset. For example, various methods have been presented that pre-train an architecture on non-medical images ([42] provides a comprehensive overview). It is also possible to train on medical images from the same task, but different scanners/scanning parameters (e.g. [37] for white-matter-hyperintensity segmentation). A much more elegant approach, in my eyes, is to train a network that is aware of differences between datasets and tries to learn an invariant representation. Van Tulder et al. [106] for example, present two deep-learning architectures based on autoencoders that learn a shared representation between image modalities on a high level. This shared representation is learned on unlabeled images of the same subject scanned with different modalities and applied to single modality train and test images. Kamnitsas et al. [56] aims to learn a

high-level shared representation between training and test data. They added a “domain discriminator” to classify between training and test data and use its output to adapt the invariant representation. In the end, the invariant representation should become so good that the discriminator becomes unable to distinguish between samples from the different domains.

I think it would be of interest to investigate how the presented transfer-learning techniques and the drawn conclusions can be transferred to a deep-learning architecture. Like an SVM classifier, a deep-learning architecture could also use sample weighting, or image weighting based on minimizing difference between source and target distributions, for example by minimizing MMD. It would also be possible to use images of subjects scanned with source and target scanner to minimize representation differences between scanners. In this regard, it might be worth investigating whether the learned shared representation of Kamnitsas et al. [56] can be improved based on such rescan images. Van Tulder et al. [106] use images of different modalities (for the same subject) to learn an invariant representation. It would be interesting to compare this method to our FST on the tested hippocampus datasets. Here, the method of Van Tulder et al. [106] could train a shared representation on scan-rescan images and apply the framework to train and test data from the the different scanners. Another interesting deep-learning approach to learn a shared representation might be to optimize a representation based on the criteria we used to optimize a representation in kernel space, such as MMD minimization between source and target.

7.4.2 Computational Load

Calculation time and memory usage is another important aspect for use in practice. We mainly focused on classification accuracy, rather than calculation time and memory. Table 7.1 gives an idea of the computational load of the various methods presented in this thesis relative to each other. Many of the presented methods can be speeded up. For example, the FST method presented in Chapter 5 could be speeded up by using only one subject scanned on both scanners instead of all available scan-rescan images (4 to 9 in our datasets). The MMD kernel-learning and image-weighting method presented in Chapter 6 could be speeded up by using a fast MMD-calculation method such as the one used in [122].

For use in practice, computational load for testing might be of more importance than that for training. When a method is applied to single-scanner data, it could be sufficient to train a classification framework once and apply it to all test data. This way, a transfer classifier such as

WSVM or A-SVM or the use of an FST need not be more time consuming to apply than a non-transfer method. For image weighting, we calculated weights for each test image individually, to account for image- and patient-specific characteristics, but this could be speeded up by using the same weights for all same-scanner test images. Transfer kernel learning can also be used to determine a single kernel for all same-scanner test images. However, applying the kernel to the test samples requires to project als test samples in feature space, which takes some calculation time.

7.4.3 When and How to Use Transfer Learning

Table 7.1 provides an overview of data requirements and performance for each of the methods, which can be of help when considering which of the presented transfer-learning techniques to use in a practical situation. I would like to highlight two methods I think are most interesting to apply in practice: the FST and image weighting. The FST can be applied to voxelwise classification in applications where registration of images of the same subject can provide a voxel-to-voxel mapping. It is likely the best choice when appearance differences between training and test images are very large, for example as in Figures 5.2(m) and (s). Since the FST is trained on (unlabeled) images from both training and test scanner, it can handle larger differences in image appearance than the other methods presented.

Image weighting, either with or without kernel learning, seems to provide the most convincing improvement in performance when distribution differences between training and test data are relatively small. It requires a large and diverse enough training set that incorporates images similar enough to the test image. These images should have not only similar distributions $P(\boldsymbol{x})$, but also similar labeling distributions $P(y|\boldsymbol{x})$. For example, image weighting is not suitable for training and testing on very different scanning modalities, contrary to the FST method, which can handle much bigger data differences. I think image weighting is best applied to training data with similar expert segmentations. With the presented image weighting, many images are given a zero weight, so only a few images are used to train the classifier. This can result in very different biases between two segmented images if positive weights are given to training images from different observers with very different segmentations. I think image weighting is a very suitable method if differences between images are relatively small (both in distributions and labels), classes are not too small, and an image provides enough samples to estimate a distribution. Image weighting can also be applied to other medical image analysis tasks where per image a decision is made (rather than per voxel), like computer-aided diagnosis. Here,

image weights can be determined based on voxel distributions between training images and a test image and used in a weighted classifier.

Overall, I think transfer-learning is a promising technique for medical image segmentation across scanners or scan protocols and to handle differences between patient groups. Over the past decades, many machine-learning methods have been developed to solve medical image segmentation problems by training on representative data. Transfer learning can be (and is) used to take these methods a step further, by training once on available data and generating a classification framework that can be applied to similar (but not necessarily completely representative) data. This way, transfer learning can reduce or eliminate the need to manually segment representative training data and facilitating the use of these methods in practice.

Chapter 8

References

[1] V. Ablavsky, C. Becker, and P. Fua. Transfer learning by sharing support vectors. Technical report, EPFL, 2012.

[2] P. Anbeek, K. Vincken, G. van Bochove, M. van Osch, J. van der Grond, et al. Probabilistic segmentation of brain tissue in MR imaging. *NeuroImage*, 27(4):795–804, 2005.

[3] P. Anbeek, K. Vincken, M. van Osch, R. Bisschops, and J. van der Grond. Automatic segmentation of different-sized white matter lesions by voxel probability estimation. *Medical Image Analysis*, 8(3):205–215, 2004.

[4] J. Ashburner and K. Friston. Unified segmentation. *NeuroImage*, 26(3):839–851, 2005.

[5] K. Babalola, B. Patenaude, P. Aljabar, J. Schnabel, D. Kennedy, W. Crum, S. Smith, T. Cootes, M. Jenkinson, and D. Rueckert. An evaluation of four automatic methods of segmenting the subcortical structures in the brain. *NeuroImage*, 47(4):1435–1447, 2009.

[6] P. Bazin and D. Pham. Topology-preserving tissue classification of magnetic resonance brain images. *Medical Imaging, IEEE Transactions on*, 26(4):487–496, 2007.

[7] C. Becker, C. Christoudias, and P. Fua. Domain adaptation for microscopy imaging. *Medical Imaging, IEEE Transactions on*, 34(5):1125–1139, 2015.

[8] M. Boccardi, M. Bocchetta, F. Morency, D. Collins, M. Nishikawa, R. Ganzola, M. Grothe, D. Wolf, A. Redolfi, M. Pievani, et al. Training labels for hippocampal segmentation based on the EADC-ADNI harmonized hippocampal protocol. *Alzheimer's & Dementia*, 11(2):175–183, 2015.

[9] E. Bron, R. Steketee, G. Houston, R. Oliver, H. Achterberg, M. Loog, J. Swieten, A. Hammers, W. Niessen, M. Smits, et al. Diagnostic classification of arterial spin labeling and structural MRI in presenile early stage dementia. *Human Brain Mapping*, 35(9):4916–4931, 2014.

[10] A. Carass, S. Roy, A. Jog, J. Cuzzocreo, E. Magrath, A. Gherman, J. Button, J. Nguyen, F. Prados, C. Sudre, et al. Longitudinal multiple sclerosis lesion segmentation: resource & challenge. *NeuroImage*, 148(1):77–102, 2017.

[11] M. Cardoso, K. Leung, M. Modat, S. Keihaninejad, D. Cash, J. Barnes, N. Fox, and S. Ourselin. STEPS: Similarity and Truth Estimation for Propagated Segmentations and its application to hippocampal segmentation and brain parcelation. *Medical Image Analysis*, 17(6):671–684, 2013.

[12] S. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(4):300–307, 2007.

[13] C. Chang and C. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

[14] B. Cheng, D. Zhang, B. Jie, and D. Shen. Sparse multimodal manifold-regularized transfer learning for MCI conversion prediction. In *Machine Learning in Medical Imaging*, pages 251–259. Springer, 2013.

[15] B. Cheng, D. Zhang, and D. Shen. Domain transfer learning for MCI conversion

prediction. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2012*, pages 82–90. Springer, 2012.

[16] V. Cheplygina, A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Asymmetric similarity-weighted ensembles for image segmentation. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*, pages 273–277. IEEE, 2016.

[17] J. Christensen. Normalization of brain magnetic resonance images using histogram even-order derivative analysis. *Magnetic Resonance Imaging*, 21(7):817–820, 2003.

[18] C. Cocosco, A. Zijdenbos, and A. Evans. A fully automatic and robust brain MRI tissue classification method. *Medical Image Analysis*, 7(4):513–527, 2003.

[19] T. Coleman and Y. Li. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6(2):418–445, 1996.

[20] C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13(3):795–828, 2012.

[21] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[22] P. Coupé, J. Manjón, V. Fonov, J. Pruessner, M. Robles, and D. Collins. Patch-based segmentation using expert priors: Application to hippocampus and ventricle segmentation. *NeuroImage*, 54(2):940–954, 2011.

[23] W. Dai, Q. Yang, G. Xue, and Y. Yu. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 193–200. ACM, 2007.

[24] R. de Boer, H. Vrooman, M. Ikram, M. Vernooij, M. Breteler, A. van der Lugt, and W. Niessen. Accuracy and reproducibility study of automatic MRI brain tissue segmentation methods. *NeuroImage*, 51(3):1047–1056, 2010.

[25] R. de Boer, H. Vrooman, F. van der Lijn, M. Vernooij, M. Ikram, A. van der Lugt, M. Breteler, and W. Niessen. White matter lesion extension to automatic brain tissue segmentation on MRI. *NeuroImage*, 45(4):1151–1161, 2009.

[26] J. de Groot, F. de Leeuw, M. Oudkerk, J. van Gijn, A. Hofman, J. Jolles, and M. Breteler. Periventricular cerebral white matter lesions predict rate of cognitive decline. *Annals of Neurology*, 52(3):335–341, 2002.

[27] L. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945.

[28] V. Dill, A. Franco, and M. S. Pinho. Automated methods for hippocampus segmentation: the evolution and a review of the state of the art. *Neuroinformatics*, 13(2):1–18, 2015.

[29] L. Duan, I. Tsang, and D. Xu. Domain transfer multiple kernel learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(3):465–479, 2012.

[30] W. Fan, I. Davidson, B. Zadrozny, and P. Yu. An improved categorization of classifier's sensitivity on sample selection bias. In *Data Mining, Fifth IEEE International Conference on*, pages 4–11. IEEE, 2005.

[31] B. Fischl. FreeSurfer. *NeuroImage*,

62(2):774–781, 2012.

[32] B. Fischl, D. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. van der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, et al. Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. *Neuron*, 33(3):341–355, 2002.

[33] J. Folkesson, E. Dam, O. Olsen, P. Pettersen, and C. Christiansen. Segmenting articular cartilage automatically using a voxel classification approach. *Medical Imaging, IEEE Transactions on*, 26(1):106–115, 2007.

[34] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, pages 23–37. Springer, 1995.

[35] K. Geras and C. Sutton. Multiple-source cross-validation. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1292–1300. JMLR Workshop and Conference Proceedings, 2013.

[36] E. Geremia, O. Clatz, B. Menze, E. Konukoglu, A. Criminisi, and N. Ayache. Spatial decision forests for MS lesion segmentation in multi-channel magnetic resonance images. *NeuroImage*, 57(2):378–390, 2011.

[37] M. Ghafoorian, A. Mehrtash, T. Kapur, N. Karssemeijer, E. Marchiori, M. Pesteie, C. Guttmann, F. de Leeuw, C. Tempany, B. van Ginneken, et al. Transfer learning for domain adaptation in MRI: Application in brain lesion segmentation. *arXiv preprint arXiv:1702.07841*, 2017.

[38] M. Goetz, C. Weber, F. Binczyk, J. Polanska, R. Tarnawski, B. Bobek-Billewicz, U. Koethe, J. Kleesiek, B. Stieltjes, and K. Maier-Hein. DALSA: Domain adaptation for supervised learning from sparsely annotated MR images. *Medical Imaging, IEEE Transactions on*, 35(1):284–196, 2015.

[39] M. Goetz, C. Weber, B. Stieltjes, and K. Maier-Hein. Learning from small amounts of labeled data in a brain tumor classification task. *Advances in Neural Information Processing Systems*, 2014.

[40] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12(7):2211–2268, 2011.

[41] H. Greenspan, A. Ruf, and J. Goldberger. Constrained Gaussian mixture model framework for automatic segmentation of MR brain images. *Medical Imaging, IEEE Transactions on*, 25(9):1233–1245, 2006.

[42] H. Greenspan, B. van Ginneken, and R. Summers. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35(5):1153–1159, 2016.

[43] R. Guerrero, C. Ledig, and D. Rueckert. Manifold alignment and transfer learning for classification of AlzheimerâĂŹs disease. In *Machine Learning in Medical Imaging*, pages 77–84. Springer, 2014.

[44] A. Guimond, A. Roche, N. Ayache, and J. Meunier. Three-dimensional multimodal brain warping using the demons algorithm and adaptive intensity corrections. *Medical Imaging, IEEE Transactions on*, 20(1):58–69, 2001.

[45] T. Heimann, P. Mountney, M. John, and R. Ionasec. Real-time ultrasound transducer localization in fluoroscopy images by transfer learning from synthetic training data. *Medical Image Analysis*, 18(8):1320–1328, 2014.

[46] J. Huang, A. Gretton, K. M. B., B. Schölkopf, and A. Smola. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, pages 601–608. NIPS, 2006.

[47] J. Huang, A. Gretton, K. Borgwardt, B. Schölkopf, and A. Smola. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, pages 601–608. NIPS, 2007.

[48] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(9):850–863, 1993.

[49] J. Iglesias, E. Konukoglu, D. Zikic, B. Glocker, K. van Leemput, and B. Fischl. Is synthesizing MRI contrast useful for inter-modality analysis? In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*, pages 631–638. Springer, 2013.

[50] J. Iglesias, C. Liu, P. Thompson, and Z. Tu. Robust brain extraction across datasets and comparison with publicly available methods. *Medical Imaging, IEEE Transactions on*, 30(9):1617–1634, 2011.

[51] M. Ikram, A. van der Lugt, W. Niessen, P. Koudstaal, G. Krestin, A. Hofman, D. Bos, and M. Vernooij. The Rotterdam Scan Study: design update 2016 and main findings. *European journal of epidemiology*, 30(12):1299–1315, 2015.

[52] M. Ikram, A. van der Lugt, W. Niessen, G. Krestin, P. Koudstaal, A. Hofman, M. Breteler, and M. Vernooij. The Rotterdam Scan Study: design and update up to 2012. *European Journal of Epidemiology*, 26(10):811–824, 2011.

[53] M. Ikram, H. Vrooman, M. Vernooij, F. van der Lijn, A. Hofman, A. van der Lugt, W. Niessen, and M. Breteler. Brain tissue volumes in the general elderly population: The Rotterdam Scan Study. *Neurobiology of Aging*, 29(6):882–890, 2008.

[54] F. Jager and J. Hornegger. Nonrigid registration of joint histograms for intensity standardization in magnetic resonance imaging. *Medical Imaging, IEEE Transactions on*, 28(1):137–150, 2009.

[55] J. Jiang and C. Zhai. Instance weighting for domain adaptation in NLP. In *ACL*, volume 7, pages 264–271, 2007.

[56] K. Kamnitsas, C. Baumgartner, C. Ledig, V. Newcombe, J. Simpson, A. Kane, D. Menon, A. Nori, A. Criminisi, D. Rueckert, et al. Unsupervised domain adaptation in brain lesion segmentation with adversarial networks. In *International Conference on Information Processing in Medical Imaging*, pages 597–609. Springer, 2017.

[57] J. Kleesiek, G. Urban, A. Hubert, D. Schwarz, K. Maier-Hein, M. Bendszus, and A. Biller. Deep MRI brain extraction: a 3D convolutional neural network for skull stripping. *NeuroImage*, 129(1):460–469, 2016.

[58] S. Klein, M. Staring, K. Murphy,

M. Viergever, and J. Pluim. Elastix: a toolbox for intensity-based medical image registration. *Medical Imaging, IEEE Transactions on*, 29(1):196–205, 2010.

[59] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1785–1792. IEEE, 2011.

[60] K. Leung, M. Clarkson, J. Bartlett, S. Clegg, C. Jack, M. Weiner, N. Fox, S. Ourselin, Alzheimer's Disease Neuroimaging Initiative, et al. Robust atrophy rate measurement in Alzheimer's disease using multi-site serial MRI: tissue-specific intensity normalization and parameter selection. *NeuroImage*, 50(2):516–523, 2010.

[61] F. Liu, D. Xu, M. Ferguson, B. Chu, T. Saam, N. Takaya, T. Hatsukami, C. Yuan, and W. Kerwin. Automated in vivo segmentation of carotid plaque MRI with morphology-enhanced probability maps. *Magnetic Resonance in Medicine*, 55(3):659–668, 2006.

[62] J. Marroquin, B. Vemuri, S. Botello, E. Calderon, and A. Fernandez-Bouzas. An accurate and efficient Bayesian method for automatic segmentation of brain MRI. *Medical Imaging, IEEE Transactions on*, 21(8):934–945, 2002.

[63] A. Mayer and H. Greenspan. An adaptive mean-shift framework for MRI brain segmentation. *Medical Imaging, IEEE Transactions on*, 28(8):1238–1250, 2009.

[64] A. Mendrik, K. Vincken, H. Kuijf, M. Breeuwer, W. Bouvy, J. de Bresser, A. Alansary, M. de Bruijne, A. Carass, A. El-Baz, et al. MRBrainS challenge: online evaluation framework for brain image segmentation in 3T MRI scans. *Computational Intelligence and Neuroscience*, 2015:1–16, 2015.

[65] B. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, et al. The multimodal brain tumor image segmentation benchmark (BRATS). *Medical Imaging, IEEE Transactions on*, 34(10):1993–2024, 2015.

[66] M. Modat, G. Ridgway, Z. Taylor, M. Lehmann, J. Barnes, D. Hawkes, N. Fox, and S. Ourselin. Fast free-form deformation using graphics processing units. *Computer Methods and Programs in Biomedicine*, 98(3):278–284, 2010.

[67] P. Moeskops, M. Viergever, A. Mendrik, L. de Vries, M. Benders, and I. Išgum. Automatic segmentation of MR brain images with a convolutional neural network. *Medical Imaging, IEEE Transactions on*, 35(5):1252–1261, 2016.

[68] S. Mueller, M. Weiner, L. Thal, R. Petersen, C. Jack, W. Jagust, J. Trojanowski, A. Toga, and L. Beckett. Ways toward an early diagnosis in Alzheimer's disease: the Alzheimer's Disease Neuroimaging Initiative (ADNI). *Alzheimer's & Dementia*, 1(1):55–66, 2005.

[69] L. Nyul, J. Udupa, and X. Zhang. New variants of a method of MRI scale standardization. *Medical Imaging, IEEE Transactions on*, 19(2):143–150, 2000.

[70] S. Pan, I. Tsang, J. Kwok, and Q. Yang. Domain adaptation via transfer component

analysis. *Neural Networks, IEEE Transactions on*, 22(2):199–210, 2011.

[71] S. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[72] V. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *Signal Processing Magazine, IEEE*, 32(3):53–69, 2015.

[73] Z. Peng, W. Wee, and J. Lee. Automatic segmentation of MR brain images using spatial-varying Gaussian mixture and Markov random field approach. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*, page 80. IEEE, 2006.

[74] S. Pereira, A. Pinto, V. Alves, and C. Silva. Brain tumor segmentation using convolutional neural networks in MRI images. *Medical Imaging, IEEE Transactions on*, 35(5):1240–1251, 2016.

[75] S. Powell, V. Magnotta, H. Johnson, V. Jammalamadaka, R. Pierson, and N. Andreasen. Registration and machine learning-based automated segmentation of subcortical and cerebellar brain structures. *NeuroImage*, 39(1):238–247, 2008.

[76] N. Prins, E. van Dijk, T. den Heijer, S. Vermeer, P. Koudstaal, M. Oudkerk, A. Hofman, and M. Breteler. Cerebral white matter lesions and the risk of dementia. *Archives of Neurology*, 61(10):1531–1534, 2004.

[77] J. Rajapakse and F. Kruggel. Segmentation of MR images with intensity inhomogeneities. *Image and Vision Computing*, 16(3):165–180, 1998.

[78] N. Robitaille, A. Mouiha, B. Crépeault, F. Valdivia, and S. Duchesne. Tissue-based MRI intensity standardization: application to multicentric datasets. *Journal of Biomedical Imaging*, 2012(4):1–11, 2012.

[79] S. Roy, A. Carass, and J. Prince. A compressed sensing approach for MR tissue contrast synthesis. In *Information Processing in Medical Imaging*, pages 371–383. Springer, 2011.

[80] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. *Computer Vision–ECCV 2010*, pages 213–226, 2010.

[81] M. Schmidt. A method for standardizing MR intensities between slices and volumes. *University of Alberta*, 2005.

[82] B. Scholkopf and A. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[83] M. Shah, Y. Xiao, N. Subbanna, S. Francis, D. Arnold, D. Collins, and T. Arbel. Evaluating intensity normalization on MRIs of human brain with multiple sclerosis. *Medical Image Analysis*, 15(2):267–282, 2011.

[84] D. Shattuck, S. Sandor-Leahy, K. Schaper, D. Rottenberg, R. Leahy, et al. Magnetic resonance image tissue classification using a partial volume model. *NeuroImage*, 13(5):856–876, 2001.

[85] B. Silverman. *Density estimation for statistics and data analysis*, volume 26. Chapman & Hall/CRC, 1986.

[86] M. Siyal and L. Yu. An intelligent modified fuzzy *c*-means based algorithm for bias esti-

mation and segmentation of brain MRI. *Pattern Recognition Letters*, 26(13):2052–2062, 2005.

[87] J. Sled, A. Zijdenbos, and A. Evans. A nonparametric method for automatic correction of intensity nonuniformity in MRI data. *Medical Imaging, IEEE Transactions on*, 17(1):87–97, 1998.

[88] S. Smith. Fast robust automated brain extraction. *Human Brain Mapping*, 17(3):143–155, 2002.

[89] M. Styner, J. Lee, B. Chin, M. Chin, O. Commowick, H. Tran, S. Markovic-Plese, V. Jewells, and S. Warfield. 3D segmentation in the clinic: A grand challenge II: MS lesion segmentation. *Midas Journal*, pages 1–6, 2008.

[90] M. Sugiyama, S. Nakajima, H. Kashima, P. Von Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. *Advances in Neural Information Processing Systems*, 20:1433–1440, 2008.

[91] M. Sugiyama, T. Suzuki, and T. Kanamori. Density ratio estimation: A comprehensive review. *RIMS Kokyuroku*, pages 10–31, 2010.

[92] P. Thévenaz and M. Unser. Optimization of mutual information for multiresolution image registration. *Image Processing, IEEE Transactions on*, 9(12):2083–2099, 2000.

[93] J. Tohka, A. Zijdenbos, A. Evans, et al. Fast and robust parameter estimation for statistical partial volume models in brain MRI. *NeuroImage*, 23(1):84–97, 2004.

[94] N. Tustison, B. Avants, P. Cook, Y. Zheng, A. Egan, P. Yushkevich, and J. Gee. N4ITK: improved N3 bias correction. *Medical Imaging, IEEE Transactions on*, 29(6):1310–1320, 2010.

[95] F. van der Lijn, M. de Bruijne, S. Klein, T. D. Heijer, Y. Hoogendam, A. van der Lugt, M. Breteler, and W. Niessen. Automated brain structure segmentation based on atlas registration and appearance models. *Medical Imaging, IEEE Transactions on*, 31(2):276–286, 2012.

[96] F. van der Lijn, T. den Heijer, M. Breteler, and W. Niessen. Hippocampus segmentation in MR images using atlas registration, voxel classification, and graph cuts. *NeuroImage*, 43(4):708–720, 2008.

[97] A. van Engelen, A. van Dijk, M. Truijman, R. van T Klooster, A. van Opbroek, A. van der Lugt, W. Niessen, M. Kooi, and M. de Bruijne. Multi-center MRI carotid plaque component segmentation using feature normalization and transfer learning. *Medical Imaging, IEEE Transactions on*, 34(6):1294–1305, 2015.

[98] K. van Leemput, F. Maes, D. Vandermeulen, and P. Suetens. Automated model-based tissue classification of MR images of the brain. *Medical Imaging, IEEE Transactions on*, 18(10):897–908, 1999.

[99] A. van Opbroek, H. Achterberg, and M. de Bruijne. Feature-space transformation improves supervised segmentation across scanners. In *Machine Learning meets Medical Imaging*, pages 85–93. Springer, 2015.

[100] A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Supervised image segmen-

tation across scanner protocols: A transfer learning approach. In *Machine Learning in Medical Imaging*, pages 160–167. Springer, 2012.

[101] A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. A transfer-learning approach to image segmentation across scanners by maximizing distribution similarity. In *Machine Learning in Medical Imaging*, pages 49–56. Springer, 2013.

[102] A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Transfer learning improves supervised image segmentation across imaging protocols. *Medical Imaging, IEEE Transactions on*, 34(5):1018–1030, 2015.

[103] A. van Opbroek, F. van der Lijn, and M. de Bruijne. Automated brain-tissue segmentation by multi-feature SVM classification. *Grand Challenge on MR Brain Image Segmentation workshop–MRBRainS13 http://mrbrains13.isi.uu.nl/*, 2013.

[104] A. van Opbroek, M. Vernooij, M. Ikram, and M. de Bruijne. Weighting training images by maximizing distribution similarity for supervised segmentation across scanners. *Medical Image Analysis*, 24(1):245–254, 2015.

[105] G. van Tulder and M. de Bruijne. Why does synthesized data improve multi-sequence classification? In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 531–538. Springer, 2015.

[106] G. van Tulder and M. de Bruijne. Representation learning for cross-modality classification. In *MICCAI Workshop on Medical Computer Vision*, pages 126–136. Springer, 2016.

[107] S. Vermeer, M. Hollander, E. van Dijk, A. Hofman, P. Koudstaal, and M. Breteler. Silent brain infarcts and white matter lesions increase stroke risk in the general population. *Stroke*, 34(5):1126–1129, 2003.

[108] H. Vrooman, C. Cocosco, F. van der Lijn, R. Stokking, M. A. Ikram, M. Vernooij, M. Breteler, and W. Niessen. Multi-spectral brain tissue segmentation using automatically trained k-Nearest-Neighbor classification. *NeuroImage*, 37(1):71–81, 2007.

[109] H. Wang, S. Das, J. Suh, M. Altinay, J. Pluta, C. Craige, B. Avants, P. Yushkevich, Alzheimer's Disease Neuroimaging Initiative, et al. A learning-based wrapper method to correct systematic errors in automatic image segmentation: consistently improved performance in hippocampus, cortex and brain segmentation. *NeuroImage*, 55(3):968–985, 2011.

[110] H. Wang, J. Suh, S. Das, J. Pluta, C. Craige, P. Yushkevich, et al. Multi-atlas segmentation with joint label fusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(3):611–623, 2013.

[111] H. Wang and P. Yushkevich. Multi-atlas segmentation with joint label fusion and corrective learning-an open source implementation. *Frontiers in Neuroinformatics*, 7(27):1–12, 2013.

[112] S. Warfield, K. Zou, and W. Wells. Simultaneous truth and performance level estimation (STAPLE): an algorithm for the validation of image segmentation. *Medical Imaging, IEEE Transactions on*, 23(7):903–921,

2004.

[113] N. Weisenfeld and S. Warfteld. Normalization of joint image-intensity statistics in MRI using the Kullback-Leibler divergence. In *Biomedical Imaging: Nano to Macro, 2004. IEEE International Symposium on*, pages 101–104. IEEE, 2004.

[114] W. Wells III, W. Grimson, R. Kikinis, and F. Jolesz. Adaptive segmentation of MRI data. *Medical Imaging, IEEE Transactions on*, 15(4):429–442, 1996.

[115] M. Wels, Y. Zheng, M. Huber, J. Hornegger, and D. Comaniciu. A discriminative model-constrained EM approach to 3D MRI brain tissue classification and intensity nonuniformity correction. *Physics in Medicine and Biology*, 56(11):3269, 2011.

[116] A. Worth. The Internet Brain Segmentation Repository (IBSR), 2009.

[117] P. Wu and T. Dietterich. Improving SVM accuracy by training on auxiliary data sources. In *Proceedings of the 21st International Conference on Machine Learning*, pages 110–117. ACM, 2004.

[118] J. Yang, R. Yan, and A. Hauptmann. Cross-domain video concept detection using adaptive SVMs. In *Proceedings of the 15th International Conference on Multimedia*, pages 188–197. ACM, 2007.

[119] Z. Yi, A. Criminisi, J. Shotton, and A. Blake. Discriminative, semantic segmentation of brain tissue in MR images. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2009*, pages 558–565, 2009.

[120] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *International Conference on Machine Learning*, pages 114–121. ACM, 2004.

[121] D. Zhang, Q. Guo, G. Wu, and D. Shen. Sparse patch-based label fusion for multi-atlas segmentation. In *Multimodal Brain Image Analysis*, pages 94–102. Springer, 2012.

[122] J. Zhao and D. Meng. FastMMD: Ensemble of circular discrepancy for efficient two-sample test. *Neural Computation*, 27(6):1345–1372, 2015.

[123] Y. Zhou and J. Bai. Atlas-based fuzzy connectedness segmentation and intensity nonuniformity correction applied to brain MRI. *Biomedical Engineering, IEEE Transactions on*, 54(1):122–129, 2007.

[124] Y. Zhuge and J. Udupa. Intensity standardization simplifies brain MR image segmentation. *Computer Vision and Image Understanding*, 113(10):1095–1103, 2009.

Chapter 9

Samenvatting

9.1 Introductie

Beeldvorming speelt een belangrijke rol in de medische wereld, zowel in de kliniek als in het onderzoek. In de kliniek helpt beeldvorming bij het stellen van een diagnose en het geven van een prognose, maar ook bij het plannen en begeleiden van een ingreep. Hier kan beeldvorming bijvoorbeeld informatie verschaffen over anatomie en beeldgeleide interventies visualiseren. In medisch onderzoek wordt beeldvorming gebruikt voor het bestuderen van de uiterlijke kenmerken van een ziekte, het ziekteverloop en tevens voor het bepalen van de effecten van een behandeling. Hierbij is het vergelijken van beelden, zowel van dezelfde patiënt op verschillende tijdstippen als tussen patiënten, van groot belang. Bij voorkeur wordt dit vergelijken gedaan op een kwantitatieve manier; met het gebruiken van zogenaamde kwantitatieve beeldbiomarkers. Dit zijn waardes die iets kunnen vertellen over een ziekte, zoals afmetingen of vormen van weefsels of structuren. Zo is bijvoorbeeld het volume en de vorm van bepaalde delen van de hersenen (hersenstructuren) voorspellend voor de ontwikkeling van verschillende vormen van dementie. Deze kwantitatieve beeldbiomarkers kunnen helpen bij het bepalen van het ziekteverloop door beelden van een patiënt te vergelijken met eerdere beelden, maar ook bij het bepalen van verschillen tussen patiënten of tussen een patiënt en een gezond persoon. Hierdoor kunnen we hopelijk beter begrijpen wat de verschillen zijn tussen patiënten en gezonde personen en hoe ziektes zich ontwikkelen. Zo kan bijvoorbeeld door het vergelijken van waardes van een patiënt met een database met waardes van gezonde personen, worden bepaald welke waardes kenmerkend kunnen zijn voor de betreffende ziekte.

MRI- en CT-scans worden het meest gebruikt voor het extraheren van kwantitatieve beeldbiomarkers, omdat ze driedimensionale beelden geven. Hiervan is MRI voor veel toepassingen in het bijzonder populair. Ten eerste omdat MRI, in tegenstelling tot CT, niet schadelijk is voor het menselijk lichaam. Daarnaast geeft MRI een beter contrast tussen zachte weefsels zoals vet- en spierweefsel en de verschillende organen dan CT. Dit maakt MRI in het bijzonder nuttig voor onderzoek naar zachte weefsels zoals de hersenen. Tenslotte kan een MRI-scanner verschillende sequenties genereren: beelden die verschillende contrasten geven tussen bepaalde soorten weefsels, waardoor er nog meer verschillende weefsels onderscheiden kunnen worden. Om kwantitatieve beeldbiomarkers te kunnen meten in MRI- en CT-beelden, moeten deze gesegmenteerd worden in de weefsels of structuren waarin men geïnteresseerd is. Figuur 9.1 geeft een voorbeeld van een plak van een MRI-hersenbeeld en een bijbehorende segmentatie in vier verschillende onderdelen: achtergrond en de drie soorten hersenweefsels (hersenvocht, grijze stof en witte stof). Het handmatig maken van een dergelijke segmentatie

is echter enorm arbeidsintensief. Daarnaast is een handmatige segmentatie ook subjectief; een ander persoon, maar ook dezelfde persoon op een ander moment, zal een andere segmentatie genereren. Hierdoor kan het lastig zijn segmentaties met elkaar te vergelijken, zeker als er slechts kleine verschillen zijn tussen beelden. Om deze twee redenen wordt er de afgelopen decennia veel onderzoek gedaan naar het automatisch genereren van dergelijke segmentaties.

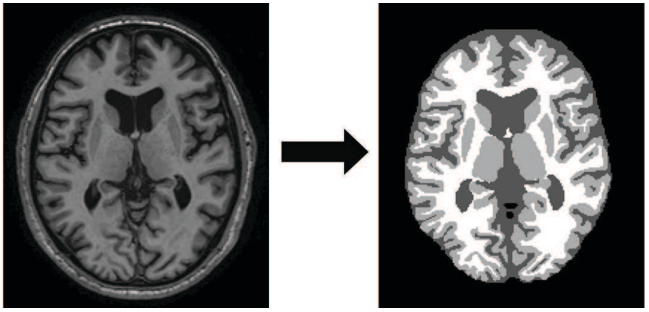


Figure 9.1: *Voorbeeld van medische beeldsegmentatie. Links: een plak van een MRI-beeld van de hersenen. Rechts: een segmentatie van het beeld in achtergrond (zwart) en de drie hersenweefsels: hersenvocht, grijze stof en witte stof (respectievelijk in donkergrijs, lichtgrijs en wit).*

9.1.1 Machine Learning voor Medische Beeldsegmentatie

Supervised machine learning heeft zich de afgelopen jaren bewezen als waardevolle techniek voor automatische medische beeldsegmentatie. Bij dit soort technieken worden voorbeelden gebruikt om een beslissingsmodel te trainen. Het voordeel van het automatisch bepalen van beslissingen aan de hand van voorbeelden is dat een ontwikkelaar niet zelf hoeft te programmeren hoe beslissingen worden gemaakt, wat heel ingewikkeld en arbeidsintensief zou zijn. In ons geval bestaan de genoemde voorbeelden uit MRI-scans en bijbehorende manuele segmentaties, deze worden ook wel de *trainingsbeelden* genoemd. Uit deze trainingsbeelden wordt vervolgens een groot aantal *trainingsamples* geëxtraheerd voor elk van de verschillende *klasses*, d.w.z. de verschillende weefsels of structuren die gesegmenteerd zijn. Deze samples bestaan in het algemeen uit voxels (d.w.z. drie-dimensionale pixels) of stukjes bestaande uit een klein aantal voxels, bijvoorbeeld $3 \times 3 \times 3$ voxels. Voor al deze trainingssamples worden vervolgens *features* gemeten: beeldeigenschappen van het sample. Voorbeelden van zulk soort features zijn bijvoorbeeld de grijswaarde (intensiteit) in het MRI-beeld, locatie in het beeld

(weergegeven in bijvoorbeeld x,y,z coördinaten), maar ook buurtinformatie zoals de gemiddelde grijswaarde rondom het sample, of verschil in grijswaarde met andere samples. Door het meten van deze features voor alle trainingssamples, kunnen deze worden voorgesteld als punten in een *featureruimte*. Deze featureruimte is vaak hoogdimensionaal, omdat veel features (tientallen, honderden, of zelfs duizenden features) worden gemeten. In deze featureruimte wordt vervolgens een beslissingsmodel, een *classifier*, geoptimaliseerd (of *getraind*) die zo goed mogelijk de samples van de verschillende klasses kan onderscheiden. Wanneer het beslissingsmodel getraind is, kan het gebruikt worden om een nieuw beeld, het *testbeeld* te segmenteren. Hiertoe wordt dit testbeeld eerst opgesplitst in samples, de *testsamples*. Voor deze testsamples worden dezelfde features gemeten als voor de trainingssamples om ze in de featureruimte te kunnen plaatsen. Tenslotte wordt de getrainde classifier toegepast om te bepalen tot welke klasse elk testsample behoort. Dit geeft vervolgens een classificatie voor elk sample in het beeld: de segmentatie. De kwaliteit van de gegenereerde automatische segmentatie kan vervolgens worden bepaald als voor het testbeeld ook een manuele segmentatie beschikbaar is (een zogenaamde *ground truth*), bijvoorbeeld door het meten van het percentage correct gesegmenteerde voxels.

Dit soort supervised machine-learningmethodes werken in het algemeen heel goed onder de voorwaarde dat de trainingsdataset groot genoeg is om een goed belissingsmodel te trainen en daarnaast representatief is voor de testbeelden. Echter, als training- en testbeelden teveel verschillen, zal een beslissingsmodel vaak niet goed presteren. Dit soort verschillen komen vaak voor als training- en testbeelden gemaakt zijn met verschillende MRI-scanners (bijvoorbeeld met een andere veldsterkte of een ander merk of type scanner), verschillende acquisitieparameters, of als de trainingsdata bestaat uit een andere populatie dan de testdata (personen met een andere leeftijd of het andere geslacht, zieke versus gezonde personen, of een ander stadium van ziekte). Dit soort verschillen zorgen voor verschillen in de gemeten features en daardoor voor verschillen tussen de verdeling van trainings- en testsamples in de featureruimte. Een classifier getraind op de trainingssamples zal daardoor vaak geen goede beslissing geven voor testsamples, wat resulteert in slechtere segmentaties. Dit heeft als effect dat een supervised machine-learningmethode die goed werkt op een bepaalde dataset (beelden van een bepaalde populatie geacquireerd met een bepaalde scanner) vaak niet kan worden toegepast op een andere dataset (bijvoorbeeld beelden van een ander ziekenhuis of populatie). Dit vormt een groot probleem voor het gebruik van dit soort methodes in de praktijk, omdat het telkens opnieuw samenstellen van een grote representatieve trainingsdataset enorm tijdsintensief is.

9.2 Dit Proefschrift

9.2.1 Transfer Learning

In dit proefschrift bestudeer ik of *transfer learning* kan helpen in het geval van dit soort verschillen tussen trainings- en testbeelden. Transfer learning is een relatief nieuw veld binnen de machine learning, dat bestaat uit methodes die kunnen omgaan met bepaalde verschillen tussen trainings- en testdata. Ik heb bestudeerd hoe transfer-learningmethodes kunnen worden toegepast op medische beeldsegmentatie en heb laten zien dat deze kunnen zorgen voor betere automatische segmentaties vergeleken met traditionele supervised machine-learningmethodes.

Bij transfer learning wordt onderscheid gemaakt tussen representatieve en onrepresentatieve trainingdata: *targetdata* bestaat uit data die representatief is voor de testdata (d.w.z. zelfde scanner, acquisitieparameters en populatie) en *sourcedata* bestaat uit data die vergelijkbaar, maar niet representatief is voor de testdata (d.w.z. andere scanner, acquisitieparameters of populatie). In de praktijk is er vaak relatief veel sourcedata beschikbaar, waarvoor ook klasselabels (manuele segmentaties) beschikbaar zijn. Targetdata is echter schaars, omdat het dataset-specifiek is. Aangezien manuele segmentatie arbeidsintensief is, zijn er voor targetdata vaak geen of slechts weinig manuele segmentaties beschikbaar. De door ons ontwikkelde methodes gebruiken daarom veelal een grote hoeveelheid sourcedata die *gelabeld* is; er zijn manuele segmentaties beschikbaar die gebruikt kunnen worden voor het trainen. Onze methodes gebruiken daarnaast weinig targetdata, die voor sommige methodes gelabeld moet zijn, maar voor de meeste van onze methodes ongelabeld is (er is geen manuele segmentatie beschikbaar). In sommige methodes zal de targetdata slechts bestaan uit het testbeeld (waarvan een manuele segmentatie niet gebruikt wordt voor het trainen, maar wel om de kwaliteit van de segmentatie te evalueren). Transfer-learningmethodes die gelabelde sourcedata en ongelabelde targetdata gebruiken worden *transductieve transfer-learningmethodes* genoemd. Methodes die daarentegen ook targetlabels gebruiken worden *inductieve transfer-learningmethodes* genoemd. Zoals gezegd heb ik onderzoek gedaan naar methodes uit beide categoriën.

Daarnaast maak ik in dit proefschrift onderscheid tussen twee verschillende aanpakken van transfer-learningmethodes. Ten eerste hebben we *transferclassificatie* bestudeerd, waarbij het verschil tussen training- en testdata wordt aangepakt bij de classificatie. Ten tweede hebben we *feature-representatietransfer* bestudeerd, waarbij features worden bepaald die verschillen

tussen training- en testdata verkleinen. Tenslotte hebben we ook bestudeerd of een combinatie van de twee aanpakken verbetering brengt ten opzichte van de individuele aanpakken.

9.2.2 Referentiemethode zonder Transfer Learning

In **Hoodstuk 2** hebben we een supervised machine-learningmethode gepresenteerd die geen transfer learning gebruikt. Deze methode is vervolgens gebruikt als startpunt en referentiemethode voor de transfer-learningtechnieken gepresenteerd in de andere hoofdstukken. De betreffende methode doet een classificatie per voxel met een support vector machine (SVM) classifier. Drie verschillende featuretypes zijn gebruikt: 1) de intensiteit van de voxel in drie verschillende MRI-sequenties; 2) Gaussian-scale-spacefeatures, dit zijn afgeleide intensiteitsfeatures die informatie over naburige voxels meenemen; 3) de coördinaten van de voxel in het beeld. De methode werd toegepast op de dataset van de MRBrainS13 hersenweefselsegmentatiewedstrijd, waarbij MRI-hersenbeelden moeten worden gesegmenteerd in achtergrond, hersenvocht, grijze stof en witte stof (gelijk Figuur 9.1). Deze dataset bestaat uit vijf trainingsbeelden en twaalf testbeelden van personen gescand met dezelfde scanner en acquisitieparameters. Hierbij kunnen we derhalve aannemen dat de trainingsbeelden representatief zijn voor de testbeelden. Onze methode gaf goede segmentatieresultaten op deze dataset en won uiteindelijk de tweede plaats in de bijbehorende wedstrijd.

9.2.3 Transferclassificatie

We hebben verschillende transferclassificatiemethodes onderzocht voor het segmenteren van beelden van verschillende datasets, afkomstig van verschillende scanners en populaties. Ten eerste, in **Hoodstuk 3**, hebben we vier transferclassifiers vergeleken voor inductieve transfer learning (veel gelabelde sourcedata en weinig gelabelde targetdata). Drie van de vier onderzochte methodes geven een gewicht aan source- en targetdata, waarbij sourcedata een lager gewicht krijgt dan targetdata, aangezien het minder vergelijkbaar is met de testdata. Twee van deze methodes passen vervolgens deze gewichten nog aan (op verschillende manieren), aan de hand van gelijkenis met de targetdata. De vierde classifier traint apart op source- en targetdata met als restrictie dat de classifier op de targetdata zo min mogelijk mag afwijken van de classifier op de sourcedata, maar wel de targetdata zo goed mogelijk moet classificeren. De prestatie van deze vier transferclassifiers hebben we vergeleken met dat van de referentieme-

thode (die geen transfer learning gebruikt) op twee medische segmentatieproblemen: hersenweefselsegmentatie (segmentatie van de hersenen in hersenvocht, grijze stof, witte stof) en segmentatie van wittestofafwijkingen (segmentatie van gezond hersenweefsel en afwijkend hersenweefsel), welke een biomaker zijn voor onder andere dementie en MS. In beide toepassingen gaf het gebruik van een transferclassifier verbetering t.o.v. de referentiemethode als te weinig targetdata beschikbaar is om een goede niet-transferclassifier te trainen.

In **Hoofdstuk 4** heb ik een andere transferclassificatietechniek gepresenteerd die, in tegenstelling tot de methodes in Hoofdstuk 3, geen labels nodig heeft van de target data (enkel het testbeeld zonder labels). Deze methode gebruikt trainingsbeelden van verschillende datasets, die elk een gewicht krijgen. Deze beeldgewichten worden vervolgens gebruikt voor het trainen van een gewogen SVM classifier. De beeldgewichten worden bepaald aan de hand van de verdeling van hun voxels in de featureruimte, op zo'n manier dat de (gewogen) trainingsverdeling zo veel mogelijk lijkt op de verdeling van de testvoxels. We hebben drie verschillende maten bestudeerd voor het verschil tussen de training- en testverdeling: de Kullback-Leibler divergentie, de Bhattacharyya afstandsfunctie en de Euclidische afstandsfunctie. Deze drie maten resulteerden in ietwat verschillende beeldgewichten. We hebben deze methode geëvalueerd op drie verschillende toepassingen: hersenweefselsegmentatie, segmentatie van wittestofafwijkingen en hersensegmentatie (segmentatie van de hersenen en de achtergrond). Ook hebben we zowel de transductieve setting bestudeerd (alleen trainingsbeelden van de sourcescanner) als de inductieve setting (trainingsbeelden van zowel source- als targetdataset). In de eerste setting gaf beeldwegen een significant beter resultaat dan een (niet-transfer) ongewogen classifier voor alledrie de toepassingen. Voor de tweede setting gaf beeldwegen ook een significant beter resultaat dan ongewogen trainen op alle (source- en target-) beelden voor alledrie de toepassingen. Daarnaast gaf beeldwegen zelfs een significant beter resultaat dan ongewogen trainen op alleen de targetbeelden voor hersenweefsel- en hersensegmentatie. Dit resultaat suggereert dat beeldwegen zelfs verbetering kan brengen in de traditionele supervised machine-learningsetting waarin representatieve (target) beelden beschikbaar zijn voor trainen.

9.2.4 Feature-representatietransfer

In Hoofdstuk 3 en 4 heb ik methodes gepresenteerd die verschillen tussen trainings- en testdata proberen op te lossen in de classifier, zonder aanpassing van de features. **Hoofdstuk 5** gebruikt een andere aanpak, waarbij een featureruimte wordt bepaald waarin verschillen in

verdeling tussen trainings- en testdata kleiner zijn. De methode heeft als doel een *featureruimtetransformatie* (FST) te leren, een relatie tussen de source- en targetfeatureruimte. Deze FST wordt bepaald aan de hand van ongelabelde beelden van een of meerdere personen die gescand zijn met zowel de source- als de targetscanner. Door deze twee beelden op elkaar te leggen, krijgen we de relatie tussen voxels (en bijbehorende features) van de sourceverdeling en voxels van de targetverdeling. Deze relatie kan vervolgens worden toegepast op voxels van gelabelde trainingsbeelden om de features te transformeren naar waardes die meer lijken op de testverdeling. Na transformatie kunnen de trainingssamples en hun labels tenslotte worden gebruikt voor het trainen van een SVM classifier. De toegevoegde waarde van de FST voor transfer learning hebben we getest op segmentatie van de hippocampus, een hersenstructuur waarvan het volume en de vorm biomarkers zijn voor de ziekte van Alzheimer. We hebben hierbij twee verschillende experimenten gedaan op twee multi-scanner datasets: een met relatief kleine verschillen tussen beelden en een met veel grotere verschillen tussen beelden. Op beide datasets gaf de FST een significante verbetering. We hebben daarnaast ook laten zien hoe de FST gecombineerd kan worden met een veelgebruikte en goedwerkende hippocampussegmentatiemethode om deze beter te laten presteren op data van verschillende scanners.

In **Hoofdstuk 6** hebben we tenslotte een combinatie van transferclassificatie en feature-representatietransfer bestudeerd. Hiertoe hebben we beeldweging gecombineerd met kernel learning, een methode waarbij een zeer hoogdimensionale featureruimte (de *kernelruimte*) wordt geleerd. We hebben twee verschillende kernel-learningmethodes bestudeerd. Een methode zoekt een kernelruimte die de classificatie van sourcedata verbetert, de andere methode zoekt een kernelruimte waarin source- en targetverdelingen meer op elkaar lijken. Voor beeldweging hebben we de methodes uit Hoofdstuk 4 met de Kullback-Leibler en Bhattacharyya afstandfunctie gebruikt. Daarnaast hebben we een nieuwe beeldweging geïntroduceerd, gebaseerd op maximum mean discrepancy (MMD) minimalisatie, welke het tegelijkertijd optimaliseren van kernel en beeldgewichten mogelijk maakt. We hebben de toegevoegde waarde van beeldwegen en kernel learning afzonderlijk en gezamenlijk bestudeerd op drie toepassingen: hersenweefselsegmentatie, segmentatie van wittestofafwijkingen en hippocampussegmentatie. Zowel beeldwegen als kernel learning afzonderlijk verbeterden de classificatie ten opzichte van een referentiemethode zonder beeldweging en kernel learning. Hierbij was de segmentatie met de nieuwe MMD beeldweging vergelijkbaar met de twee methodes van Hoofdstuk 4. De twee verschillende kernel-learningmethodes presteerden ook vergelijkbaar. Als beeldweging en kernel learning gecombineerd werden, verbeterden de resultaten nog een beetje ten opzichte van het gebruik van alleen beeldwegen of alleen kernel learning. Het gezamenlijk optimaliseren van de kernel en beeldgewichten, wat efficiënter is omdat het slechts een enkele

optimalisatie vergt, gaf een zeer vergelijkbaar resultaat als losse optimalisatie.

9.3 Conclusie

De resultaten in dit proefschrift laten zien dat transfer-learningmethodes medische beeldsegmentatietechnieken kunnen verbeteren in het geval van training- en testdata van verschillende scanners, acquisitieparameters of populaties. Transfer learning kan zowel gebruikt worden in de classificatiestap, door het gebruik van een transferclassifier of beeldweging, als bij de featurerepresentatie, door het gebruik van een featureruimtetransformatie of kernel learning. We hebben ook laten zien dat een transferclassifier en feature-representatietransfer gecombineerd kunnen worden voor een mogelijk extra verbetering. Door beter om te gaan met verschillen tussen datasets kan transfer learning de toepasbaarheid van supervised medische beeldbewerkingstechnieken verbeteren. Deze betere toepasbaarheid op data met verschillende karakteristieken is zowel voor de kliniek als het medische onderzoek zeer nuttig.

Chapter 10

Dankwoord

De afgelopen zeven jaar heb ik altijd beweerd dat een promotie niet heel veel anders is dan elke andere baan, maar nu dit proefschrift voor me ligt ben ik toch geneigd deze stelling enigszins te herzien. Aan de ene kant geniet je als promovendus de bijna oneindige vrijheid te onderzoeken wat jíj het interessantst vindt met het proefschrift als ultiem doel aan het einde van de reis. Aan de andere kant kan het enorm eenzaam voelen om als enige dit specifieke doel te hebben. En voor de planners onder ons (of misschien ook juist de niet-planners) kan het proefschrift ook soms als een donderwolk boven je hoofd hangen, want onderzoek laat zich moeilijk plannen. Ik denk dat iedereen in elke baan zichzelf beter leert kennen, maar de dingen die ik over mezelf heb geleerd tijdens deze reis, had ik nooit voor mogelijk gehouden. Er zijn veel mensen geweest die me, in verschillende vormen, geholpen en gesteund hebben. Deze bijzondere mensen wil ik hierbij dan ook enorm bedanken.

Ten eerste mijn co-promotor, Marleen de Bruijne. Marleen, ik ben eigenlijk vrij toevallig bij jou begonnen als afstudeerder. Jouw kamergenoot, Stefan Klein, begeleidde me bij het zoeken naar een afstudeeropdracht. Toen ik aan het einde van de kennismakingsmiddag tegen hem vertelde dat ik eigenlijk iets met “wat meer machine learning” zocht draaide jij je om en zei “dan heb ik denk ik wel iets voor je”. Je VIDI-aanvraag zat toen nog in de pipeline, maar ik mocht alvast op het onderwerp afstuderen. Blijkbaar zag je in dat jaar potentie in me, want toen de aanvraag was goedgekeurd bood je me een PhD-plek aan die ik (na wat twijfelen of een PhD wel bij me paste) aannam. Dat is ondertussen bijna zeven jaar geleden. Bij deze wil ik je enorm bedanken voor alle hulp in deze jaren in de vorm van nieuwe ideeën, feedback, *out-of-the-box* denken en me er af en toe op wijzen dat het tijd was een paper af te maken of eindelijk die planning te herzien. Maar bovenal wil ik je bedanken voor het vertrouwen en de steun tijdens de twee moeilijkste momenten: mijn *mid-PhD-crisis* en bij het afmaken van mijn laatste paper, toen al mijn tijd opging aan mijn PostDoc-functie. Uiteindelijk is het allemaal gelukt en daarin heb jij een grote rol gespeeld, waarvoor mijn dank.

Ten tweede mijn promotor, Wiro Niessen. Wiro, ik kreeg al snel het gevoel dat jij niet zozeer boven als wel naast ons, de BIGR promovendi, staat. Je hebt een groot vertrouwen in iedereen in de groep en geeft ons zoveel mogelijk vrijheid opdat iedereen zijn passie kan volgen. Ik denk dat het precies dit vertrouwen is dat de BIGR groep zo’n fijne groep maakt om te werken, waarin ieder op zijn eigen tempo zijn eigen weg kan bewandelen. Daarnaast heb je een aanstekelijk enthousiasme voor het vakgebied en een fijne *hands-on* mentaliteit, wat me vaak geholpen heeft zaken in een nieuw perspectief te zien, een nieuw doel te stellen en deze te bereiken.

I would also like to thank all my committee members for their time and effort. I especially thank the members of my inner committee, dr. Ben Glocker, dr. Ivana Išgum, and prof.dr. Aad van der Lugt for reading and approving my thesis. Prof.dr. Tom Heskes, dr. Marion Smits, and dr.ir. Veronika Cheplygina thank you for your willingness to be part of my committee.

Esther en Hakim, bedankt voor alle steun de afgelopen zeven jaar en dat jullie mijn paranimfen willen zijn. Met jullie allebei heb ik voor mijn gevoel *min of meer* een halve PhD gedeeld. Esther, we begonnen tegelijk als *interns*, waarop we allebei bij BIGR bleven voor een PhD. Inhoudelijk deden we vaak nét iets anders, maar daarnaast waren we op alle fronten PhD-maatjes. Na de PhD's (toen jij je proefschrift al snel af had en ik maar met dat laatste paper bleef worstelen) werden we infrastructuurmaatjes en werkten we ineens ook inhoudelijk veel samen, wat zowel leuk als heel leerzaam was. Samen hebben we veel bereikt, veel lol gehad en heel veel besproken, waarvoor ik je wil bedanken. Hakim, ik weet nog dat jij me in mijn eerste week een spontane rondleiding door het EE-gebouw gaf en enthousiast ronduit praatte alsof je me al jaren kende. Jouw *drive* om mensen zich welkom te laten voelen en jouw enthousiasme om samen te werken heb ik altijd enorm gewaardeerd. Na de MBM-meeting kwam je vaak naar me toe om te brainstormen over ofwel jouw ofwel mijn machine-learningmethode. In de infrastructuurgroep heb ik je nog beter leren kennen, ook omdat we beiden ernaast nog een PhD af moesten maken. Ons enthousiasme voor het ontdekken van nieuwe methodes heeft voor ons allebei veel vertraging opgeleverd, maar een half jaar geleden heb jij eindelijk je PhD afgerond en vandaag ben ik aan de beurt.

Of course I would like to thank all my other colleagues and former colleagues at the BIGR group. I consider myself very lucky to work among such a diverse group of smart, talented, friendly people. There are a few people I would like to thank in particular. Although BIGR is a very technical group, we have a rather large group of women. Arna, Carolyn, Emilie, Esther, Hortense, Nora, Veronika, Wyke, Zahra, thanks for all the fun we had at BIGR-girls events. Arna, Carolyn, Esther, Wyke, Zahra, thanks for all the stories we shared over lunch or a cup of tea. Being among you made me feel very at home in this male-dominated field. Gijs, bedankt dat je vele jaren mijn kamergenoot en mede-transfer-learning-PhD bent geweest. We hebben weinig echt samengewerkt, maar des te meer gebrainstormd en gewoon samen gelachen. Florian, thanks for being my roommate for the last couple of months and introducing me to the wonderful world of deep learning. You and Gijs make working at EMC much more fun than working from home. Veronika, toen ik hoorde dat jij als PostDoc op mijn project zou gaan werken was ik meteen enthousiast. Ik heb heel veel van je geleerd in die twee jaar en wil je enorm bedanken voor de fijne samenwerking. I would also like to thank the infrastructure

group: Adriaan, Esther, Hakim, Marcel, Mattias for teaching me about infrastructure, FASTR, XNAT, Python, Git, but also about coördinating people, working in a team, planning, Scrum, software development and much more. I learned so much from each one of you, while also having so much fun. Marius, jij ook enorm bedankt voor alle hulp op het infra-vlak en de fijne gesprekken. Joram, bedankt dat ik jouw afstudeerbegeleider mocht zijn en zo het onderzoeksprocess een keer van de andere kant mocht bekijken. Ik wens je heel veel plezier en succes met je eigen PhD! Tenslotte wil ik Désirée en Petra graag enorm bedanken voor het regelen van alle benodigdheden tijdens het PhD-traject.

Arfan en Meike, bedankt dat jullie je steeds weer door mijn wiskundige papers hebben geworsteld om de meerwaarde op epidemiologisch vlak te verduidelijken. Aad en Daniel, grote dank aan jullie voor de ontzettend fijne en leerzame samenwerking binnen het BBMRI-project. Bedankt ook voor het consistent voorzichtig controleren of er nog steeds vooruitgang zat in het afmaken van de PhD.

Voor velen van mijn vrienden zal een PhD waarschijnlijk ook niet veel anders zijn dan elke andere baan, met als zekere uitzonderingen de vrienden die zichzelf *doctor* mogen noemen. Eén voordeel van een PhD is in ieder geval het kunnen uitspreken van een woord van dank naar onze vriendschap, iets dat Nederlanders naar mijn mening veel te weinig doen. Aart, Anne, Ardaan, Bart, Bram, Cecile, Charlotte, Dorien, Elfri, Irene, Ivonne, Jacqueline, Jan, Jan-Paul, Julia, Juliette, Jurian, Lieke, Maarten, Madeleine, Marleen, Maud, Menno, Michaela, Michiel, Nienke, Nina, Olga, Steven, Tara, Wouter, bedankt voor onze vriendschap. Bedankt voor de steun en het af en toe vragen hoe het gaat met m'n promotie, maar vooral voor al het plezier en de ervaringen die we samen hebben gedeeld. Ook mijn trainingsmaatjes bij CrossFit Delft en in het bijzonder Barry en Ron wil ik bedanken voor de afgelopen jaren. Misschien was m'n PhD sneller af geweest als Lieke me niet had geïntroduceerd tot de CrossFit-wereld, misschien had ik juist een manier gemist om even afstand te nemen tot het onderzoek, maar het is zeker dat de afgelopen jaren een stuk minder leuk waren geweest zonder jullie. Elke januari stelden we doelen voor het komende jaar. Velen hiervan heb ik nog steeds niet gehaald, maar die PhD kan bij deze in ieder geval worden afgestreept!

Grote dank natuurlijk ook aan mijn eigen familie en mijn schoonfamilie, voor alle interesse en steun, maar vooral ook voor alle gezelligheid en warmte. Papa, jij was de eerste wetenschapper in mijn leven en de dag dat ik begon aan mijn PhD had ik in mijn hoofd alvast één van mijn stellingen vastgelegd. Als jij me niet van kleins af aan had uitgedaagd met natuurkundige vraagstukken was ik waarschijnlijk nooit wetenschapper geworden. Mama, bedankt voor alle

steun, liefde en trots. Je zou jezelf nooit een techneut noemen, maar die liefde voor cijfertjes heb ik toch echt van jou. Jij en papa hebben samen de fundering gelegd voor de volwassen vrouw die ik nu ben, waarvoor ik ontzettend dankbaar ben. Pieter, ook dank aan jou dat je mijn broer bent. Je hebt zelf ontzettend veel bereikt, wat me enorm trots maakt. Bedankt dat je altijd interesse toont in hoe het gaat met mij en mijn onderzoek.

Lieve Steffan, 12 jaar geleden had ik het genoegen een jaar naast je te mogen zitten in het bestuur van *Christiaan Huygens*. In dat jaar hebben we elkaar gevonden en één jaar naast elkaar werd twaalf jaar. Maar ook na al die tijd maakt mijn hart nog vaak een sprongetje als ik naast je zit. Bedankt voor je oneindige liefde, steun, hulp en vertrouwen. Je herinnert me eraan dat “alles goed komt” en je inspireert me dagelijks een nog betere versie van mezelf te worden. Ik had niemand beters naast me kunnen wensen in de reis die *promotie* heet.

Publications

Journal Papers

A. van Opbroek, H. Achterberg, M. Vernooij, and M. de Bruijne. Transfer learning for image segmentation by combining image weighting and kernel learning. *Submitted*.

A. van Opbroek, H. Achterberg, M. Vernooij, M. Ikram, and M. de Bruijne. Transfer learning by feature-space transformation: a method for hippocampus segmentation across scanners. *Submitted*.

V. Cheplygina, **A. van Opbroek**, M.A. Ikram, M.W. Vernooij, and M. de Bruijne. Transfer Learning by Asymmetric Image Weighting for Segmentation across Scanners. *arXiv preprint arXiv:1703.04981*, 2017.

A. van Opbroek, M. Vernooij, M. Ikram, and M. de Bruijne. Weighting training images by maximizing distribution similarity for supervised segmentation across scanners. *Medical Image Analysis*, 24(1):245–254, 2015.

A. Mendrik, K. Vincken, H. Kuijf, M. Breeuwer, W. Bouvy, J. de Bresser, A. Alansary, M. de Bruijne, A. Carass, A. El-Baz, A. Jog, R. Katyal, A. Khan, F. van der Lijn, Q. Mahmood, R. Mukherjee, **A. van Opbroek**, S. Paneri, S. Pereira, M. Persson, M. Rajchl, D. Sarikaya, Ö. Smedby, C. Silva, H. Vrooman, S. Vyas, C. Wang, L. Zhao, G. Biessels, and M. Viergever. MRBrainS challenge: online evaluation framework for brain image segmentation in 3T MRI scans. *Computational Intelligence and Neuroscience*, 2015:1–16, 2015.

A. van Engelen, A. van Dijk, M. Truijman, R. van ‘t Klooster, **A. van Opbroek**, A. van der Lugt,

W. Niessen, M. Kooi, and M. de Bruijne. Multi-center MRI carotid plaque component segmentation using feature normalization and transfer learning. *Medical Imaging, IEEE Transactions on*, 34(6):1294–1305, 2015.

A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Transfer learning improves supervised image segmentation across imaging protocols. *Medical Imaging, IEEE Transactions on*, 34(5):1018–1030, 2015.

M. Dekking, D. Kong, and **A. van Opbroek**. Plumes in kinetic transport: how the simple random walk can be too simple. *Stochastic Models*, 28(4): 635–648, 2012.

Conference and Workshop Papers

V. Cheplygina, **A. van Opbroek**, M. Ikram, M. Vernooij, and M. de Bruijne. Asymmetric similarity-weighted ensembles for image segmentation. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*, pages 273–277. IEEE, 2016.

A. van Opbroek, H. Achterberg, and M. de Bruijne. Feature-space transformation improves supervised segmentation across scanners. In *Machine Learning meets Medical Imaging*, pages 85–93. Springer, 2015.

A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. A transfer-learning approach to image segmentation across scanners by maximizing distribution similarity. In *Machine Learning in Medical Imaging*, pages 49–56. Springer, 2013.

A. van Opbroek, F. van der Lijn, and M. de Bruijne. Automated brain-tissue segmentation by multi-feature SVM classification. *Grand Challenge on MR Brain Image Segmentation workshop – MRBRainS13 http://mrbrains13.isi.uu.nl/*, 2013.

A. van Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Supervised image segmentation across scanner protocols: A transfer learning approach. In *Machine Learning in Medical Imaging*, pages 160–167. Springer, 2012.

Conference Abstracts

H. Achterberg, **A. van Opbroek**, M. Koek, D. Bos, M. Vernooij, M. Ikram, H. Hulshoff Pol, W. Niessen, and A. van der Lugt. Quantitative imaging biomarkers for biobanking. *Global Biobank Week*, 2017.

T. Kroes, H. Achterberg, B. van Lew, M. Koek, **A. van Opbroek**, A. Versteeg, and B. Lelieveldt. Pipeline inspection and monitoring. *Global Biobank Week*, 2017.

M. Koek, H. Achterberg, **A. van Opbroek**, A. Versteeg, D. Bos, B. van Lew, T. Kroes, M. Zwiers, Y. Caspi, H. Hulshoff Pol, A. van der Lugt, and W. Niessen. Imaging biomarker infrastructure. *Global Biobank Week*, 2017.

L. Tap, **A. van Opbroek**, W. Niessen, M. Smits, and F. Mattace-Raso. Vascular aging is associated with the severity of cerebral white matter lesion load. *Artery Research* 20: 100–101, 2017.

PhD Portfolio

PhD period 2011-2018
Departments Radiology & Medical Informatics
Research School ASCI

In-depth courses

Knowledge driven Image Segmentation (ASCI)	2012
Machine Learning Summer School, Santa Cruz de la Palma, Spain (PASCAL2)	2012
Presentation Course (Medical Informatics, Erasmus MC)	2012
Advanced Pattern Recognition (ASCI)	2012
Principles of Research in Medicine and Epidemiology (NIHES)	2012
Domain Adaptation Summer School, Copenhagen, Denmark (DIKU)	2012
English Biomedical Writing and Communication (Erasmus MC)	2012-2013
Computer Vision by Learning (ASCI)	2014
C++ course (BIGR)	2013-2014
Software and Data Carpentry Intructor Training, Manchester, UK (Elixir)	2015

International conferences

Medical Image Computing and Computer-Assisted Intervention - MICCAI, Nice, France (attendance)	2012
Medical Image Computing and Computer-Assisted Intervention - MICCAI, Nagoya, Japan (attendance)	2013
International Conference on Machine Learning - ICML, Lille, France (attendance)	2015

Seminars and workshops

Fall Meeting Nederlandse Vereniging voor Patroonherkenning en Beeldverwerking - NVPHBV, Delft (attendance)	2011
MICCAI Workshop on Machine Learning in Medical Imaging - MLMI, Nice, France (oral presentation)	2012
Medical Imaging Symposium for PhD students - MISP, Nijmegen (attendance)	2012
MICCAI Workshop on Machine Learning in Medical Imaging Workshop - MLMI, Nagoya, Japan (poster presentation)	2013
MICCAI Grand Challenge on MR Brain Image Segmentation Workshop - MRBrainsS, Nagoya, Japan (participation and oral presentation)	2013
Medical Imaging Symposium for PhD students - MISP, Utrecht (attendance)	2013
Medical Imaging Symposium for PhD students - MISP, Leiden (organization and attendance)	2014
ICML Workshop on Deep Learning - Lille, France (attendance)	2015
ICML Workshop on Machine Learning meets Medical Imaging - MLMI, Lille, France (oral presentation)	2015
Medical Imaging Symposium for PhD students - MISP, Amsterdam (attendance)	2015
Medical Imaging Symposium for PhD students - MISP, Eindhoven (oral presentation)	2016

Research seminar series

Biomedical Imaging Group Rotterdam Seminars, bi-weekly (2 presentations)	2011-2018
Medical Informatics Research Lunch, bi-weekly (2 presentations)	2011-2018
Medical Image Segmentation and Registration Seminars, weekly (14 presentations)	2011-2016
Model-Based Medical Image Analysis Seminars, weekly (27 presentations)	2011-2018

Teaching experience

Teaching Introduction to Image Processing to medical students	2012-2014
Global BioImaging course, Heidelberg, Germany (presentation and practical)	2016
Supervision master thesis project - Joram Keijser, Project: Learning Transferable	

and Discriminative Features by Differentiable Sparse Coding 2016-2017

Awards, nominations and grants

Second prize, MICCAI Grand Challenge on MR Brain Image Segmentation Workshop - MRBrainsS, Nagoya, Japan (participation and oral presentation) 2013

Other

De Jonge Akademie on Wheels 2012
BIGR Outing (organization) 2012
NRC Career Cafe, PhD Edition 2014
BIGR Open-lab Day (organization) 2016
Health-RI: Empowering personalized medicine and health research (poster presentation) 2016

About the author

Anna Gretha (Annegreet) van Opbroek was born in 1986 in Woubrugge, the Netherlands. In 2004 she finished secondary school in Tilburg with the predicate 'Cum Laude' and started a Bachelor in Applied Mathematics at the Delft University of Technology. During her Bachelor she specialized in probability, statistics, and stochastic research and broadened her knowledge with a minor in Applied Physics. Afterwards, Annegreet did a Master in Biomedical Engineering with specialization in Medical Imaging. Following her enthousiasm for probability&statistics, she soon became intrigued with machine learning. She conducted her masters thesis at the Biomedical Imaging Group Rotterdam (BIGR) at the Erasmus Medical Centre in Rotterdam, the Netherlands, on the application of transductive transfer-learning techniques in biomedical image segmentation.

Subsequently, Annegreet started a PhD at the BIGR group in October 2011 on the application of transfer learning for biomedical image segmentation. She was supervised by Wiro Niessen (promotor) and Marleen de Bruijne (co-promotor). She investigated the applicability of transfer-learning techniques to account for differences between training and test images, such as differences in used scanners, scanning protocols, and patient characteristics. She also herself developed several new transfer-learning techniques for medical image segmentation. The application of her project mainly lies in MRI brain segmentation.

As of September 2015, Annegreet has been working as a post-doctoral researcher at the BIGR group for a multi-center project on the development of IT infrastructure for large imaging studies. Here, she coordinated software developers at the different centers and develops and robustifies neuro-image-segmentation techniques for large multi-center datasets.

ISBN 978-94-6299-952-7