White Matter Lesion Segmentation (WMLS) Manual

1. Introduction

White matter lesions (WMLs) are brain abnormalities that appear in different brain diseases, such as multiple sclerosis (MS), head injury, hypertension-related vascular diseases, and various forms of dementia. Their incidence also increases with normal aging. MRI is routinely used as surrogate in the study of WMLs, as MRI signal changes reflect certain aspects of the underlying brain pathology. Out of the many available MRI acquisition protocols, T1-w and T2-w, PD, and FLAIR are most commonly used for evaluation of WML load in the brain. Computer analysis methods have started to complement expert readings of these images, as they may improve throughput and consistency, in addition to providing more accurate quantitative measures of WMLs. Computer analysis is even more critical in longitudinal studies that involve relatively small changes in WML, thereby rendering it advantageous, if not necessary, to use unbiased computer-assisted segmentation methods to detect WML changes longitudinally.

We describe in this document a novel multi-spectral WML segmentation protocol via incorporating information from T1-w, T2-w, PD-w and FLAIR MR brain images. Numerous techniques in medical image analysis are incorporated to achieve the AAA (Automatic, Accurate and Applicable) criteria.

2. Overview of the Algorithm

The WML segmentation algorithm [1-3] uses a combination of image analysis and machine learning techniques such as Support Vector Machine (SVM) [5]. Image intensities from multiple MR acquisition protocols, after co-registration, are used to form voxel-wise attribute vectors that help reduce ambiguity in discriminating lesion and normal tissue during segmentation. Three steps are involved (see Fig. 1):

1. Preprocessing includes co-registration of the MR images, skull stripping [4], intensity normalization, and inhomogeneity correction.

2. For training the classification model, a set of training samples is prepared by expert readers via manual segmentation.

3. The WML segmentation result is generated by voxel-wise classification. False positive voxels are eliminated by considering the distance distribution in a Hilbert space.

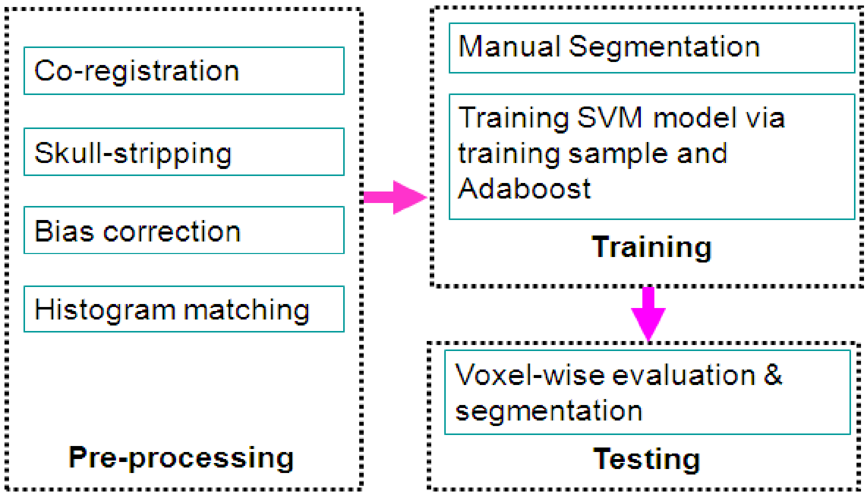

Fig. 1. Overview of the WML segmentation algorithm.

3. Installation

3.1. Required Software Packages

The following is needed:

- Insight Toolkit (Ver. 2.4 or higher)
- CMake (Ver. 2.6 or higher)
- 3D Slicer (Ver. 3.4 or higher)

3.2. Download & Installation

The software package can either be downloaded from the 3D Slicer website or from NITRC.

a) 3D Slicer (for users):

Details on module installation can be found in the tutorial: White_Matter_Lesion_Segmentation_Tutorial.pdf, which can be found at http://www.nitrc.org/projects/hammerwml.

After launching 3D Slicer.

1. Press F2 or go to View >> Application Settings >> Module Settings on the menu of Slicer3.
2. Click the "add a preset" button.
3. Select the location of the White Matter Lesion Segmentation modules (*wmlstrain* and *wmlstest*).
4. Close Slicer3 and restart.

b) NITRC (for developers)

1. Download and extract the source codes and sample data from http://www.nitrc.org/projects/hammerwml.
2. Run CMake based on CMakeList.txt in the *src* folder and compile the code by using the *make* command.
3. The generated executable files (*wmlstrain* and *wmlstest*) can be used with 3D Slicer following the instructions given above.

4. User Guide

4.1.Training

4.1.1 GUI

Details on GUI usage for training the classifiers are included in the tutorial file: White_Matter_Lesion_Segmentation_Tutorial.pdf, which can be found at http://www.nitrc.org/projects/hammerwml.

4.1.2 Command line

wmlstrain *DataDirectory TrainSubjectList SuffixList ModelDirectory*

- *DataDirectory*: directory that contains the training images
- *TrainSubjectList*: a text file listing the file names (w/o extensions) of the training images
- *SuffixList*: a text file listing the file name extensions of the training images
- *ModelDirectory*: path to the directory where the training results will be stored

4.2. Testing

4.1.1 GUI

Details on GUI usage for training the classifiers are included in the tutorial file: White_Matter_Lesion_Segmentation_Tutorial.pdf, which can be found at http://www.nitrc.org/projects/hammerwml.

4.1.2 Command line

wmlstest *Modelname TestSubjectList SuffixList*

- *Modelname*: path to the directory where the training results were stored
- *TestSubjectList*: a text file listing the file names (w/o extensions) of the testing images

- *SuffixList*: a text file listing the file name extensions of the testing images.

The segmentation results will be saved as *afterfinaltest_threshold.** in the specified destination folder.

5. Developer's guide

Here we provide the detailed descriptions on the ITK classes related to training, testing and preprocessing, respectively. Once the source code is modified, new executable files can be generated by following the instructions in 3.2.2.

5.1. Training

- main function in wmlstrain.cxx: This performs feature selection by calling a feature selector (itk::WMLGetSelectedFeature) and carries out training based on the features.
- itk::WMLGetSelectedFeature<ImageType, MeasurementVectorType>: This class extracts features for both lesion and non-lesion ROIs from training samples.
- itk::WMLAdaptiveBooster< ImageType, MeasurementVectorType >: This class help select best feature set by testing the initial feature set iteratively.
- Also, SVM-related classes for building SVM models are included in the training part: itk::SVMSolverBase, itk::SVMSolver1, itk::SVMKernelBase

5.2. Testing

- main function in wmlstest.cxx: This performs feature selection by calling a feature selector itk::WMLGetSelectedFeature) and carries out testing based on the features.
- itk::WMLGetSelectedFeature<ImageType, MeasurementVectorType>: This class extracts features from testing sample.
- itk:: WMLTestingProcessor<InputImageType, OutputImageType>: Performs testing based on the selected features from testing image and learned SVM model in the training stage.

5.3. Preprocessing

Before training and testing, preprocessing such as skull stripping, affine registration, and bias correction need to be performed.

-itk::WMLPreprocessor<InputImageType, OutputImageType>: The preprocessing core.

-itk::AffineRegistration<InputImageType, OutputImageType>: Aligns all sample images.

-itk::BiasCorrection<InputImageType, OutputImageType>: Bias correction.

-itk::HistogramMatching<InputImageType, OutputImageType>: Matches the intensity histograms of training and testing images.

-itk::SkullStripping<InputImageType, OutputImageType>: Skull stripping to extract the brain parenchyma.

-itk::RemoveEyeregion<InputImageType, OutputImageType>: Removes the eyes from the images.

6. References

[1] ZQ Lao, DG Shen, A Jawad, B Karacali, DF Liu, ER Melhem, NR Bryan, C Davatzikos, "Automated Segmentation of White Matter Lesions in 3D Brain MR Images, Using Multivariate Pattern Classification", Third IEEE International Symposium on Biomedical Imaging (ISBI 2006), April 6-9, 2006, Arlington, VA, USA.

[2] ZQ Lao, DG Shen, DF Liu, AF Jawad, ER Melhem, LJ Launer, RN Bryan, C. Davatzikos, "Computer-assisted segmentation of white matter lesions in 3D MR images, using support vector machine", Academic Radiology, 15(3):300-313, 2008.

[3] Yuchen Xie and Xiaodong Tao, “White Matter Lesion Segmentation Using Weakly Labeled MR Images”, SPIE Medical Imaging, 2011.

[4] Xiaodong Tao, Ming-ching Chang, "A Skull Stripping Method Using Deformable Surface and Tissue Classification", SPIE Medical Imaging, San Diego, CA, 2010.

[5] T. Collobert and S. Bengio, "SVMTorch: Support vector machine for large-scale regression problems," Journal of Machine Learning Research, vol. 1, pp. 143-160, 2001.