Package 'sublime'

September 30, 2016

Type Package

Title Automatic Lesion Incidence Estimation and Detection using Multi-Modality Longitudinal Magnetic Resonance Images

Version 1.3

Date 2016-09-29

Maintainer Elizabeth M. Sweeney <elizabethmargaretsweeney@gmail.com>

Description Creates probability maps of incident and enlarging lesion voxels from a baseline and followup magnetic resonance imaging study in patients with multiple sclerosis.

License GPL

LazyData true

Depends oro.nifti

Imports downloader, graphics, stats, grDevices, AnalyzeFMRI

RoxygenNote 5.0.1.9000

NeedsCompilation no

Author Elizabeth M. Sweeney [aut, cre], John Muschelli [aut]

Repository CRAN

Date/Publication 2016-09-30 08:30:25

R topics documented:

SuBLIME-package *SuBLIME package: Subtraction-Based Logistic Inference for Modeling and Estimation*

Description

This packages runs the SuBLIME pipeline on preprocessed data for detection of lesions in patients iwth MS. The functions will normalize and runt he model

Details

Package:	SuBLIME
Type:	Package
Version:	1.0
Date:	2013-08-12
License:	What license is it under?

Author(s)

Elizabeth Sweeney, John Muschelli

Elizabeth M. Sweeney <emsweene@jhsph.edu>

References

Sweeney, E. M., Shinohara, R. T., Shea, C. D., Reich, D. S., & Crainiceanu, C. M. (2013). Automatic Lesion Incidence Estimation and Detection in Multiple Sclerosis Using Multisequence Longitudinal MRI. American Journal of Neuroradiology, 34(1), 68-73.

See Also

~~ AnalyzeFMRI ~~

download_data *Download SuBLIME data*

Description

Download test data for examples

Usage

```
download_data(folder = system.file(package = "SuBLIME"), force = FALSE)
```

Arguments

folder	Folder to download the data - usually SuBLIME folder, but may need a different directory due to permissions
force	Force download of file even if it exists

Value

Indicator if the file was downloaded and unzipped

---

normalize *Intensity-normalization by a image mask*

Description

This function normalized the image by the mean and standard deviation by intensities of voxels in the mask. In SuBLIME the mask is normal appearing white matter

Usage

```
normalize(image, mask = NULL)
```

Arguments

image	3D Array or object of class nifti
mask	Mask with same dimensions. If NULL, then original image is returned

Value

Object of class nifti or array, depending on image input

Examples

```
## Not run:
## put in ex here

## End(Not run)
```

SuBLIME_model *Sublime Predictive model*

Description

Predictive model for SuBLIME algorithm

Usage

```
SuBLIME_model
```

Format

An `lm` object, but with data an other things, notably qr removed

References

Sweeney, E. M., et al. "Automatic lesion incidence estimation and detection in multiple sclerosis using multisequence longitudinal MRI." American Journal of Neuroradiology 34.1 (2013): 68-73.

SuBLIME_prediction *Gets predicted probabilities from SuBLIME*

Description

Takes in MRI images from followup and gets predictions (probabilities) of the enhancing of lesions

Usage

```
SuBLIME_prediction(baseline_flair, follow_up_flair, baseline_pd, follow_up_pd,
  baseline_t2, follow_up_t2, baseline_t1, follow_up_t1, time_diff,
  baseline_nawm_mask = NULL, follow_up_nawm_mask = baseline_nawm_mask,
  brain_mask, model = SuBLIME::SuBLIME_model, voxsel = TRUE,
  smooth.using = c("GaussSmoothArray", "none"), voxsel.sigma = diag(3, 3),
  voxsel.ksize = 5, s.sigma = diag(3, 3), s.ksize = 3,
  plot.imgs = FALSE, slice = 90, pdfname = "diag.pdf", verbose = TRUE)
```

Arguments

`baseline_flair`	Baseline FLAIR image, either array or class nifti
`follow_up_flair`	Followup FLAIR image, either array or class nifti
`baseline_pd`	Baseline PD image, either array or class nifti
`follow_up_pd`	Followup PD image, either array or class nifti
`baseline_t2`	Baseline T2 image, either array or class nifti

follow_up_t2	Followup T2 image, either array or class nifti
baseline_t1	Baseline T1 image, either array or class nifti
follow_up_t1	Followup T1 image, either array or class nifti
time_diff	Difference in time (in days) between baseline and followup, numeric
baseline_nawm_mask	Baseline Normal Appearing white matter mask, either array or class nifti. Will be coerced to logical usign baseline_nawm_mask > 0. If NULL, no NAWM normalization is done (assumes data is already normalized)
follow_up_nawm_mask	Followup Normal Appearing white matter mask, either array or class nifti. Will be coerced to logical usign follow_up_nawm_mask > 0. Defaults to baseline_nawm_mask if not specified. If NULL, no NAWM normalization is done (assumes data is already normalized)
brain_mask	Brain mask, either array or class nifti. Will be #' coerced to logical usign brain_mask > 0.
model	Model of class lm or set of coefficients.
voxsel	Do Voxel Selection based on normalized T2 (logical)
smooth.using	Character vector to decide if using GaussSmoothArray from AnalyzeFMRI or fslsmooth from fslr package
voxsel.sigma	Sigma passed to voxel_select
voxsel.ksize	Kernel size passed to voxel_select
s.sigma	Sigma passed to GaussSmoothArray
s.ksize	Kernel size passed to GaussSmoothArray
plot.imgs	Plot images along the way
slice	Slice to be plotted
pdfname	Name of pdf created for plot.imgs
verbose	Print Diagnostic Messages

Value

array

See Also

predict

Examples

```
## Not run:
download_data()
modes = c("FLAIR", "PD", "T2", "VolumetricT1")
modals = paste0(modes, "norm.nii.gz")
base_files = system.file(file.path("01/Baseline", modals), package="SuBLIME")
base_imgs = lapply(base_files, readNIfTI, reorient=FALSE)
f_files = system.file(file.path("01/FollowUp", modals), package="SuBLIME")
```

```
f_imgs = lapply(f_files, readNIfTI, reorient=FALSE)
names(base_imgs) = names(f_imgs) = modes
baseline_nawm_file =  system.file("01/Baseline/nawm.nii.gz", package="SuBLIME")
baseline_nawm_mask =  readNIfTI(baseline_nawm_file, reorient=FALSE)
baseline_nawm_mask = drop(baseline_nawm_mask)
follow_up_nawm_file =  system.file("01/FollowUp/nawm.nii.gz", package="SuBLIME")
follow_up_nawm_mask =  readNIfTI(follow_up_nawm_file, reorient=FALSE)
brain_file =  system.file("01/duramask.nii.gz", package="SuBLIME")
brain_mask =  readNIfTI(brain_file, reorient=FALSE)
brain_mask = drop(brain_mask)

follow_up_nawm_mask = NULL
baseline_nawm_mask = NULL
smooth.using = "GaussSmoothArray"
verbose = TRUE
time_diff = 10
voxsel = TRUE
model = SuBLIME_model
#voxsel.sigma = s.sigma =diag(3,3)
#s.ksize = 3
#voxsel.ksize = 5

outimg = SuBLIME_prediction(
baseline_flair = base_imgs[["FLAIR"]],
follow_up_flair= f_imgs[["FLAIR"]],
baseline_pd = base_imgs[["PD"]],
follow_up_pd = f_imgs[["PD"]],
baseline_t2 = base_imgs[["T2"]],
follow_up_t2 = f_imgs[["T2"]],
baseline_t1 = base_imgs[["VolumetricT1"]],
follow_up_t1 = f_imgs[["VolumetricT1"]],
time_diff = time_diff,
baseline_nawm_mask = baseline_nawm_mask,
brain_mask = brain_mask,
voxsel = voxsel,
model = model, plot.imgs= TRUE,
pdfname = "~/Dropbox/SuBLIME_Web_Test/01/pckg_diagnostc.pdf"
)

names(base_imgs) = paste0("baseline_", c("flair", "pd", "t2", "t1"))
names(f_imgs) = paste0("follow_up_", c("flair", "pd", "t2", "t1"))
attach(base_imgs)
attach(f_imgs)

## End(Not run)
```

---

voxel_select *Select Potential Voxels*

Description

Takes the difference in T2 images, smoothes this difference, and then finds voxels greater than one SD of the smooothed mask as potential voxels and returns it

Usage

```
voxel_select(normalized_baseline_t2, normalized_follow_up_t2, brain_mask,
  sigma = diag(3, 3), ksize = 5)
```

Arguments

`normalized_baseline_t2`	Baseline T2 image, array or object class nifti that
`normalized_follow_up_t2`	Followup T2 image, array or object class nifti that
`brain_mask`	A 3D 0-1 mask that delimits where the smoothing occurs, passed to `GaussSmoothArray`
`sigma`	Sigma passed to `GaussSmoothArray`
`ksize`	Kernel size passed to `GaussSmoothArray`

Value

Array or object class nifti depending on imput iamges

See Also

GaussSmoothArray

Index