Package 'causalimages'

August 8, 2023

Title causalimages: R Package for Causal Inference with Earth Observation, Biomedical, and Social Science Images

Version 0.1

Authors 'Connor Jerzak <connor.jerzak@gmail.com> [aut, cre], Fredrik Johansson <fredrik.johansson@chalmers.se> [aut], Adel Daoud <adel.daoud@liu.se> [aut]'

Description R Package for causal inference with earth observation, biomedical, and social science images and image sequences (i.e., videos)

Depends R (>= 3.3.3)

License
Creative Commons Attribution-Noncommercial-No Derivative Works 4.0, for academic use only.

Encoding UTF-8

LazyData true

Maintainer 'Connor Jerzak' <connor.jerzak@gmail.com>

Imports tensorflow

RoxygenNote 7.2.3

R topics documented:

AnalyzeImageConfounding 2
AnalyzeImageHeterogeneity 4
AnalyzeImageMediation 7
GetAndSaveGeolocatedImages 9
image2 10
LongLat2CRS 11
SimulateImageSystem 11
WriteTfRecord 12

Index **14**

`AnalyzeImageConfounding` *Perform causal estimation under image confounding*

Description

Under beta release. Full release in late 2023.

Usage

```
AnalyzeImageConfounding(obsW, obsY, acquireImageFxn, ...)
```

Arguments

`obsW`	A numeric vector where `0`'s correspond to control units and `1`'s to treated units.
`obsY`	A numeric vector containing observed outcomes.
`X`	(optional) A numeric matrix containing tabular information used if `orthogonalize = T`. `X` is normalized internally and salience maps with respect to `X` are transformed back to the original scale.
`doConvLowerDimProj`	(default = `T`) Should we project the `nFilters` convolutional feature dimensions down to `nDimLowerDimConv` to reduce the number of required parameters.
`nDimLowerDimConv`	(default = `3L`) If `doConvLowerDimProj = T`, then, in each convolutional layer, we project the `nFilters` feature dimensions down to `nDimLowerDimConv` to reduce the number of parameters needed.
`nFilters`	(default = `50L`) Integer specifying the number of convolutional filters used.
`nBoot`	(default = `100L`) Number of bootstrap iterations for uncertainty estimation.
`typeBoot`	(default = `SamplingOnly`) Bootstrap type. `typeBoot = 'SamplingOnly'` captures sampling uncertainty only. `typeBoot = 'EstimationAndSampling'` captures both estimation and sampling uncertainty.
`imageKeysOfUnits`	(default = `1:length(obsY)`) A vector of length `length(obsY)` specifying the unique image ID associated with each unit. Samples of `imageKeysOfUnits` are fed into `acquireImageFxn` to call images into memory.
`acquireImageFxn`	A function specifying how to load images representations associated with `imageKeysOfUnits` into memory. For example, if observation `3` has a value of `"a34f"` in `imageKeysOfUnits`, `acquireImageFxn` should extract the image associated with the unique key `"a34f"`. First argument should be image key values and second argument have be `training` (in case different behavior in training/inference mode).
`transportabilityMat`	(optional) A matrix with a column named `keys` specifying keys to be used by `acquireImageFxn` for generating treatment effect predictions for out-of-sample points.
`long, lat`	(optional) Vectors specifying longitude and latitude coordinates for units. Used only for describing highest and lowest probability neighorhood units if specified.

conda_env (default = NULL) A string specifying a conda environment wherein tensorflow, tensorflow_probability, and gc are installed.

conda_env_required (default = F) A Boolean stating whether use of the specified conda environment is required.

figuresTag (default = "") A string specifying an identifier that is appended to all figure names.

figuresPath (default = "./") A string specifying file path for saved figures made in the analysis.

tagInFigures (default = F) A Boolean specifying whether to visually include the tag in the figures.

plotBands (default = 1L) An integer or vector specifying which band position (from the acquired image representation) should be plotted in the visual results. If a vector, plotBands should have 3 (and only 3) dimensions (corresponding to the 3 dimensions to be used in RBG plotting).

simMode (default = F) Should the analysis be performed in comparison with ground truth from simulation?

plotResults (default = T) Should analysis results be plotted?

nDepthHidden_conv (default = 3L) Hidden depth of convolutional layer.

nDepthHidden_dense (default = 0L) Hidden depth of dense layers. Default of 0L means a single projection layer is performed after the convolutional layer (i.e., no hidden layers are used).

maxPoolSize (default = 2L) Integer specifying the max pooling size used in the convolutional layers.

strides (default = 2L) Integer specifying the strides used in the convolutional layers.=

batchSize (default = 50L) Batch size used in SGD optimization.

kernelSize (default = 5L) Dimensions used in convolution kernels.

nSGD (default = 400L) Number of stochastic gradient descent (SGD) iterations.

nDenseWidth (default = 32L) Width of dense projection layers post-convolutions.

channelNormalize (default = T) Should channelwise image feature normalization be attempted? Default is T, as this improves training.

tf_seed (default = NULL) Specification for the tensorflow seed.

quiet (default = F) Should we suppress information about progress?

Value

A list consiting of

- ATE_est ATE estimate.
- ATE_se Standard error estimate for the ATE.
- (images saved to disk if plotResults = T) If plotResults = T, causal salience plots are saved to disk characterizing the image confounding structure. See references for details.

References

- Connor T. Jerzak, Fredrik Johansson, Adel Daoud. Integrating Earth Observation Data into Causal Inference: Challenges and Opportunities. *ArXiv Preprint*, 2023.

Examples

```
# For a tutorial, see
# github.com/cjerzak/causalimages-software/
```

`AnalyzeImageHeterogeneity`

Decompose treatment effect heterogeneity by image

Description

Implements the image heterogeneity decomposition analysis of Jerzak, Johansson, and Daoud (2023). Users input in treatment and outcome data, along with a function specifying how to load in images using keys referenced to each unit (since loading in all image data will usually not be possible due to memory limitations). This function by default performs estimation, constructs salience maps, and can optionally perform estimation for new areas outside the original study sites in a transportability analysis.

Usage

```
AnalyzeImageHeterogeneity(obsW, obsY, acquireImageFxn, kClust_est, ...)
```

Arguments

`obsW`	A numeric vector where `0`'s correspond to control units and `1`'s to treated units.
`obsY`	A numeric vector containing observed outcomes.
`X`	(optional) A numeric matrix containing tabular information used if `orthogonalize = T`.
`orthogonalize`	(default = `F`) A Boolean specifying whether to perform the image decomposition after orthogonalizing with respect to tabular covariates specified in `X`.
`imageKeysOfUnits`	(default = `1:length(obsY)`) A vector of length `length(obsY)` specifying the unique image ID associated with each unit. Samples of `imageKeysOfUnits` are fed into `acquireImageFxn` to call images into memory.
`kClust_est`	(default = `2L`) Integer specifying the number of clusters used in estimation.
`acquireImageRepFxn`	A function specifying how to load images representations associated with `imageKeysOfUnits` into memory. For example, if observation `3` has a value of `"a34f"` in `imageKeysOfUnits`, `acquireImageFxn` should extract the image associated with the unique key `"a34f"`. First argument should be image key values and second argument have be `training` (in case behavior in training/)
`acquireImageFxn`	(default = `acquireImageRepFxn`) Similar to `acquireImageRepFxn`; this is a function specifying how to load images associated with `imageKeysOfUnits` into memory.

`transportabilityMat`
: (optional) A matrix with a column named `keys` specifying keys to be used by `acquireImageRepFxn` for generating treatment effect predictions for out-of-sample points.

`long, lat`
: (optional) Vectors specifying longitude and latitude coordinates for units. Used only for describing highest and lowest probability neighorhood units if specified.

`conda_env`
: (default = `NULL`) A string specifying a conda environment wherein `tensorflow`, `tensorflow_probability`, and `gc` are installed.

`conda_env_required`
: (default = `F`) A Boolean stating whether use of the specified conda environment is required.

`figuresTag`
: (default = `""`) A string specifying an identifier that is appended to all figure names.

`figuresPath`
: (default = `"./"`) A string specifying file path for saved figures made in the analysis.

`simMode`
: (default = `F`) Should the analysis be performed in comparison with ground truth from simulation?

`plotResults`
: (default = `T`) Should analysis results be plotted?

`nDepthHidden_conv`
: (default = `3L`) Hidden depth of convolutional layer.

`nDepthHidden_dense`
: (default = `0L`) Hidden depth of dense layers. Default of `0L` means a single projection layer is performed after the convolutional layer (i.e., no hidden layers are used).

`maxPoolSize`
: (default = `2L`) Integer specifying the max pooling size used in the convolutional layers.

`strides`
: (default = `2L`) Integer specifying the strides used in the convolutional layers.=

`yDensity`
: (default = `normal`) Specifies the density for the outcome. Current options include `normal` and `lognormal`.

`nMonte_variational`
: (default = `5L`) An integer specifying how many Monte Carlo iterations to use in the calculation of the expected likelihood in each training step.

`nMonte_predictive`
: (default = `20L`) An integer specifying how many Monte Carlo iterations to use in the calculation of posterior means (e.g., mean cluster probabilities).

`nMonte_salience`
: (default = `100L`) An integer specifying how many Monte Carlo iterations to use in the calculation of the salience maps (e.g., image gradients of expected cluster probabilities).

`batchSize`
: (default = `25L`) Batch size used in SGD optimization.

`kernelSize`
: (default = `5L`) Dimensions used in convolution kernels.

`nSGD`
: (default = `400L`) Number of stochastic gradient descent (SGD) iterations.

`nDenseWidth`
: (default = `32L`) Width of dense projection layers post-convolutions.

`reparameterizationType`
: (default = `"Flipout"`) Either `"Flipout"`, or `"Reparameterization"`. Specifies the estimator used in the Bayesian neural components. With `"Flipout"`, convolutions are performed via CPU; with ‘"Reparameterization", they are performed by GPU if available.

doConvLowerDimProj
: (default = T) Should we project the nFilters convolutional feature dimensions down to nDimLowerDimConv to reduce the number of required parameters.

nDimLowerDimConv
: (default = 3L) If doConvLowerDimProj = T, then, in each convolutional layer, we project the nFilters feature dimensions down to nDimLowerDimConv to reduce the number of parameters needed.

nFilters
: (default = 32L) Integer specifying the number of convolutional filters used.

channelNormalize
: (default = T) Should channelwise image feature normalization be attempted? Default is T, as this improves training.

quiet
: (default = F) Should we suppress information about progress?

Value

A list consiting of

- clusterTaus_mean default
- clusterTaus_sd Estimated image effect cluster standard deviations.
- clusterProbs_mean. Estimated mean image effect cluster probabilities.
- clusterTaus_sd. Estimated image effect cluster probability standard deviations.
- clusterProbs_lowerConf. Estimated lower confidence for effect cluster probabilities.
- impliedATE. Implied ATE.
- individualTau_est. Estimated individual-level image-based treatment effects.
- transportabilityMat. Transportability matrix withestimated cluster information.
- plottedCoordinates. List containing coordinates plotted in salience maps.
- whichNA_dropped. A vector containing observations dropped due to missingness.

References

- Connor T. Jerzak, Fredrik Johansson, Adel Daoud. Image-based Treatment Effect Heterogeneity. Forthcoming in *Proceedings of the Second Conference on Causal Learning and Reasoning (CLeaR), Proceedings of Machine Learning Research (PMLR)*, 2023.

Examples

```
# For a tutorial, see
# github.com/cjerzak/causalimages-software/
```

AnalyzeImageMediation *AnalyzeImageMediation*

Description

Under construction.

Usage

```
AnalyzeImageMediation()
```

Arguments

obsW	A numeric vector where 0's correspond to control units and 1's to treated units.
obsY	A numeric vector containing observed outcomes.
X	(optional) A numeric matrix containing tabular information used if orthogonalize = T.
doConvLowerDimProj	(default = T) Should we project the nFilters convolutional feature dimensions down to nDimLowerDimConv to reduce the number of required parameters.
nDimLowerDimConv	(default = 3L) If doConvLowerDimProj = T, then, in each convolutional layer, we project the nFilters feature dimensions down to nDimLowerDimConv to reduce the number of parameters needed.
nFilters	(default = 32L) Integer specifying the number of convolutional filters used.
orthogonalize	(default = F) A Boolean specifying whether to perform the image decomposition after orthogonalizing with respect to tabular covariates specified in X.
imageKeysOfUnits	(default = 1:length(obsY)) A vector of length length(obsY) specifying the unique image ID associated with each unit. Samples of imageKeysOfUnits are fed into acquireImageFxn to call images into memory.
acquireImageRepFxn	A function specifying how to load images representations associated with imageKeysOfUnits into memory. For example, if observation 3 has a value of "a34f" in imageKeysOfUnits, acquireImageFxn should extract the image associated with the unique key "a34f". First argument should be image key values and second argument have be training (in case behavior in training/)
acquireImageFxn	(default = acquireImageRepFxn) Similar to acquireImageRepFxn; this is a function specifying how to load images associated with imageKeysOfUnits into memory.
transportabilityMat	(optional) A matrix with a column named keys specifying keys to be used by acquireImageRepFxn for generating treatment effect predictions for out-of-sample points.
long, lat	(optional) Vectors specifying longitude and latitude coordinates for units. Used only for describing highest and lowest probability neighorhood units if specified.

conda_env (default = NULL) A string specifying a conda environment wherein tensorflow, tensorflow_probability, and gc are installed.

figuresKey (default = "") A string specifying an identifier that is appended to all figure names.

figuresPath (default = "./") A string specifying file path for saved figures made in the analysis.

simMode (default = F) Should the analysis be performed in comparison with ground truth from simulation?

plotResults (default = T) Should analysis results be plotted?

nDepthHidden_conv (default = 3L) Hidden depth of convolutional layer.

nDepthHidden_dense (default = 0L) Hidden depth of dense layers. Default of 0L means a single projection layer is performed after the convolutional layer (i.e., no hidden layers are used).

maxPoolSize (default = 2L) Integer specifying the max pooling size used in the convolutional layers.

strides (default = 2L) Integer specifying the strides used in the convolutional layers.=

yDensity (default = normal) Specifies the density for the outcome. Current options include normal and lognormal.

nMonte_variational (default = 5L) An integer specifying how many Monte Carlo iterations to use in the calculation of the expected likelihood in each training step.

nMonte_predictive (default = 20L) An integer specifying how many Monte Carlo iterations to use in the calculation of posterior means (e.g., mean cluster probabilities).

nMonte_salience (default = 100L) An integer specifying how many Monte Carlo iterations to use in the calculation of the salience maps (e.g., image gradients of expected cluster probabilities).

batchSize (default = 25L) Batch size used in SGD optimization.

kernelSize (default = 5L) Dimensions used in convolution kernels.

nSGD (default = 400L) Number of stochastic gradient descent (SGD) iterations.

nDenseWidth (default = 32L) Width of dense projection layers post-convolutions.

reparameterizationType (default = "Flipout") Either "Flipout", or "Reparameterization". Specifies the estimator used in the Bayesian neural components. With "Flipout", convolutions are performed via CPU; with '"Reparameterization", they are performed by GPU if available.

channelNormalize (default = T) Should channelwise image feature normalization be attempted? Default is T, as this improves training.

quiet (default = F) Should we suppress information about progress?

Value

A list consiting of

- ATE_est ATE estimate.
- ATE_se Standard error estimate for the ATE.

References

- TBA.

Examples

```
# For a tutorial, see
# github.com/cjerzak/causalimages-software/
```

`GetAndSaveGeolocatedImages` *Getting and saving geo-located images from a pool of .tif's*

Description

A function that finds the image slice associated with the `long` and `lat` values, saves images by band (if `save_as = "csv"`) in save_folder.

Usage

```
GetAndSaveGeolocatedImages(long, lat, keys, tif_pool, save_folder)
```

Arguments

`long`	Vector of numeric longitudes.
`lat`	Vector of numeric latitudes.
`keys`	The image keys associated with the long/lat coordinates.
`tif_pool`	A character vector specifying the fully qualified path to a corpus of .tif files.
`image_pixel_width`	An even integer specifying the pixel width (and height) of the saved images.
`save_folder`	(default = `"."`) What folder should be used to save the output? Example: `"~/Downloads"`
`save_as`	(default = `".csv"`) What format should the output be saved as? Only one option currently (`.csv`)
`lyrs`	(default = NULL) Integer (vector) specifying the layers to be extracted. Default is for all layers to be extracted.

Value

Finds the image slice associated with the `long` and `lat` values, saves images by band (if `save_as = "csv"`) in save_folder. The save format is: `sprintf("%s/Key%s_BAND%s.csv", save_folder, keys[i], band_)`

Examples

```
# Example use (not run)
MASTER_IMAGE_POOL_FULL_DIR <- c("./LargeTifs/tif1.tif","./LargeTifs/tif2.tif")
GetAndSaveGeolocatedImages(
                        long = GeoKeyMat$geo_long,
                        lat = GeoKeyMat$geo_lat,
                        image_pixel_width = 500L,
                        keys = row.names(GeoKeyMat),
                        tif_pool = MASTER_IMAGE_POOL_FULL_DIR,
                        save_folder = "./Data/Uganda2000_processed",
                        save_as = "csv",
                        lyrs = NULL)
```

---

image2 *Visualizing matrices as heatmaps with correct north-south-east-west orientation*

Description

A function for generating a heatmap representation of a matrix with correct spatial orientation.

Usage

```
image2( x )
```

Arguments

x	(required) The numeric matrix to be visualized.
xaxt	(default = "") The x-axis tick labels.
yaxt	(default = "") The y-axis tick labels.
xlab	(default = "") The x-axis labels.
ylab	(default = "") The y-axis labels.
main	(default = "") The main figure label.
cex.main	(default = 1.) The main figure label sizing factor.
box	(default = F) Should a box be plotted around the image?

Value

Returns a heatmap representation of the matrix, x, with correct north/south/east/west orientation.

Examples

```
#set seed
set.seed(1)

#Geneate data
x <- matrix(rnorm(50*50), ncol = 50)
diag(x) <- 3

# create plot
image2(x, main = "Example Text", cex.main = 2)
```

LongLat2CRS *Get the spatial point of long/lat coordinates*

Description

A function converts long/lat coordinates into a spatial points object defined by a coordinate reference system (CRS).

Usage

```
LongLat2CRS(long, lat, CRS_ref)
```

Arguments

long	Vector of numeric longitudes.
lat	Vector of numeric latitudes.
CRS_ref	A CRS into which the long-lat point should be projected.

Value

Returns the long/lat location as a spatial point in the new CRS defined by CRS_ref

Examples

```
spatialPt <- LongLat2CRS(
                  long = 49.932,
                  lat = 35.432,
             CRS_ref = sp::CRS("+proj=lcc +lat_1=48 +lat_2=33 +lon_0=-100 +ellps=WGS84"))
```

SimulateImageSystem *Simulate causal systems involving images*

Description

This function generates simulated causal structures using images. It is currently under construction.

Usage

```
SimulateImageSystem(...)
```

Arguments

dag	*(character string)* An input DAG specifying causal structure. This input should be of the form 'i->t,i->y,t->y,....' Currently, only one node in a DAG can be an image (this should be labeled "i"). The non-image nodes can have arbitrary string labels. The image can be a confounder, effect moderator, effect mediator. If the image is to be used as a moderator, use the notation, t-i>y.

`...`	*(optional)* In estimation mode, users input the data matrices associated with the non-image nodes of `DAG` and image node `i`. For example, if `x` is a DAG node, users must, in estimation mode, supply data to `x` in a form that can be coerced to a tensor.
`treatment`	*(character string, optional)* In estimation mode, users specify the treatment variable here. If `treatment` is specified, users must provide other data inputs to the DAG (see `...`).
`image_pool`	*(character string, optional)* The path to where analysis specific images are located. This can be specified both in simulation and estimation mode. If not specified, the simulation uses a pool of Landsat images from Nigeria.
`analysis_level`	*(character string, default is `'scene'`)* Defines the unit of analysis used in the simulation framework. This is ignored in estimation mode, where the unit of analysis is inferred from the data dimensions.
`control`	*(list)* A list containing control parameters in the data generating process.

Value

A list:

- In *simulation mode*, the function returns a list with as many elements as unique nodes in `DAG`. Each element represents the simulated data.
- In *estimation mode*,the function returns an estimated treatment effect with 95\

References

- Connor T. Jerzak, Fredrik Johansson, Adel Daoud. Image-based Treatment Effect Heterogeneity. Forthcoming in *Proceedings of the Second Conference on Causal Learning and Reasoning (CLeaR), Proceedings of Machine Learning Research (PMLR)*, 2023.

Examples

```
#set seed
set.seed(1)

# Simulation mode
#simulatedData <- causalimage('r->i, i->t, t->y, r->y')
#print(names(simulatedData))

# Estimation mode
#estimatedResults <- causalimage('r->i, i->t, t->y, r->y', y=y, r=r, y=y', treatment='t')
#print( estimatedResults )
```

`WriteTfRecord` *Write an image corpus as a .tfrecord file*

Description

Writes an image corpus to a `.tfrecord` file for rapid reading of images into memory for fast ML training.

Usage

```
WriteTfRecord(file,acquireImageRepFxn,conda_env)
```

Arguments

file	A character string naming a file for writing.
imageKeys	A vector specifying the image keys of the corpus. A key grabs an image via acquireImageRepFxn(key)
acquireImageRepFxn	A function whose input is an observation index and whose output is an image.
conda_env	A conda environment where tensorflow v2 lives.

Value

Writes an index-referenced .tfrecord from an image corpus for use in image-based causal inference training.

Examples

```
# Example usage:
#WriteTfRecord(
#  file = "./NigeriaConfoundApp.tfrecord",
#  acquireImageRepFxn = acquireImageRepFxn,
#  conda_env = "tensorflow_m1")
```

Index

AnalyzeImageConfounding, 2
AnalyzeImageHeterogeneity, 4
AnalyzeImageMediation, 7

GetAndSaveGeolocatedImages, 9

image2, 10

LongLat2CRS, 11

SimulateImageSystem, 11

WriteTfRecord, 12