Package 'causalimages'

February 2, 2025

Title Causal Inference with Earth Observation, Bio-medical, and Social Science Images

Version 0.1

Description Provides a system for performing causal inference with earth observation, bio-medical, and social science images and image sequences (videos). The package uses a 'JAX' backend for GPU/TPU acceleration. Key functionalities include building conda-based backends (e.g., via 'BuildBackend'), implementing image-based confounder and heterogeneity analyses (e.g., 'AnalyzeImageConfounding', 'AnalyzeImageHeterogeneity'), and writing/reading large image corpora as '.tfrecord' files for use in training (via 'WriteTfRecord' and 'GetElementFromTfRecordAtIndices'). This allows researchers to scale causal inference to modern large-scale imagery data, bridging R with hardware-accelerated Python libraries. The package is partly based on Jerzak and Daoud (2023) <arXiv:2310.00233>.

URL https://github.com/cjerzak/causalimages-software

BugReports https://github.com/cjerzak/causalimages-software/issues

Depends R (>= 3.3.3)

License GPL-3

Encoding UTF-8

LazyData false

Imports tensorflow, latex2exp, keras, reticulate, viridis, geosphere, raster, animation, rrapply, glmnet, sf, data.table, grf, pROC, devtools

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.2

Contents

`AnalyzeImageConfounding` *Perform causal estimation under image confounding*

Description

Perform causal estimation under image confounding

Usage

```
AnalyzeImageConfounding(
  obsW,
  obsY,
  X = NULL,
  file = NULL,
  imageKeysOfUnits = NULL,
  fileTransport = NULL,
  imageKeysOfUnitsTransport = NULL,
  nBoot = 10L,
  inputAvePoolingSize = 1L,
  useTrainingPertubations = T,
  useScalePertubations = F,
  orthogonalize = F,
  transportabilityMat = NULL,
  latTransport = NULL,
  longTransport = NULL,
  lat = NULL,
  long = NULL,
  conda_env = "CausalImagesEnv",
```

```
  conda_env_required = T,
  Sys.setenv_text = NULL,
  figuresTag = NULL,
  figuresPath = "./",
  plotBands = 1L,
  plotResults = T,
  optimizeImageRep = T,
  nWidth_ImageRep = 64L,
  nDepth_ImageRep = 1L,
  kernelSize = 5L,
  nWidth_Dense = 64L,
  nDepth_Dense = 1L,
  imageModelClass = "VisionTransformer",
  pretrainedModel = NULL,
  strides = 2L,
  nDepth_TemporalRep = 3L,
  patchEmbedDim = 16L,
  dropoutRate = 0.1,
  batchSize = 16L,
  nSGD = 400L,
  testFrac = 0.05,
  TfRecords_BufferScaler = 4L,
  learningRateMax = 0.001,
  TFRecordControl = NULL,
  dataType = "image",
  image_dtype = "float16",
  atError = "stop",
  seed = NULL
)
```

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`	An optional 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.
`file`	Path to a tfrecord file generated by `WriteTfRecord`.
`imageKeysOfUnits`	A vector of length `length(obsY)` specifying the unique image ID associated with each unit. Samples of `imageKeysOfUnits` are fed into the package to call images into memory.
`nBoot`	Number of bootstrap iterations for uncertainty estimation.
`useTrainingPertubations`	Boolean specifying whether to randomly the image axes during training to reduce overfitting.
`transportabilityMat`	Optional matrix with a column named `imageKeysOfUnits` specifying keys to be used by the package 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 — A conda environment where computational environment lives, usually created via causalimages::BuildBackend(). Default = "CausalImagesEnv".

conda_env_required — A Boolean stating whether use of the specified conda environment is required.

figuresTag — A string specifying an identifier that is appended to all figure names.

figuresPath — A string specifying file path for saved figures made in the analysis.

plotBands — An integer or vector specifying which band position (from the 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).

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

optimizeImageRep — Boolean specifying whether to optimize over the image model representation (or only over downstream parameters).

nWidth_ImageRep — Integer specifying width of image model representation.

nDepth_ImageRep — Integer specifying depth of image model representation.

kernelSize — Dimensions used in spatial convolutions.

nWidth_Dense — Integer specifying width of image model representation.

nDepth_Dense — Integer specifying depth of dense model representation.

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

dropoutRate — Droppout rate used in training used to prevent overfitting (dropoutRate = 0 corresponds to no dropout).

batchSize — Batch size used in SGD optimization. Default = 50L.

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

testFrac — Default = 0.1. Fraction of observations held out as a test set to evaluate out-of-sample loss values.

TfRecords_BufferScaler — The buffer size used in tfrecords mode is batchSize*TfRecords_BufferScaler. Lower TfRecords_BufferScaler towards 1 if out-of-memory problems.

dataType — (default = "image") String specifying whether to assume "image" or "video" data types.

Value

Returns a list consisting of

- ATE_est ATE estimate.
- ATE_se Standard error estimate for the ATE.
- plotResults If set to TRUE, 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 or image sequence*

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,
  X = NULL,
  orthogonalize = F,
  imageKeysOfUnits = 1:length(obsY),
  kClust_est = 2,
  file = NULL,
  transportabilityMat = NULL,
  lat = NULL,
  long = NULL,
  conda_env = "CausalImagesEnv",
  conda_env_required = T,
  figuresTag = "",
  figuresPath = "./",
  plotBands = 1L,
  heterogeneityModelType = "variational_minimal",
  plotResults = F,
  optimizeImageRep = T,
  nWidth_ImageRep = 64L,
  nDepth_ImageRep = 1L,
  nWidth_Dense = 64L,
  nDepth_Dense = 1L,
  nDepth_TemporalRep = 1L,
  useTrainingPertubations = T,
  strides = 2L,
  pretrainedModel = NULL,
  testFrac = 0.1,
  kernelSize = 5L,
  learningRateMax = 0.001,
  TFRecordControl = NULL,
```

```
  patchEmbedDim = 16L,
  nSGD = 500L,
  batchSize = 16L,
  seed = NULL,
  Sys.setenv_text = NULL,
  imageModelClass = "VisionTransformer",
  nMonte_predictive = 10L,
  nMonte_salience = 10L,
  nMonte_variational = 2L,
  TfRecords_BufferScaler = 4L,
  temperature = 1,
  inputAvePoolingSize = 1L,
  dataType = "image"
)
```

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	Optimal numeric matrix containing tabular information used if orthogonalize = T.
orthogonalize	A Boolean specifying whether to perform the image decomposition after orthogonalizing with respect to tabular covariates specified in X.
imageKeysOfUnits	A vector of length length(obsY) specifying the unique image ID associated with each unit. Samples of imageKeysOfUnits are fed into the package to call images into memory.
kClust_est	Integer specifying the number of clusters used in estimation. Default is 2L.
file	Path to a tfrecord file generated by WriteTfRecord.
transportabilityMat	An optional matrix with a column named key specifying keys to be used for generating treatment effect predictions for out-of-sample points in earth observation data settings.
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	A conda environment where computational environment lives, usually created via causalimages::BuildBackend(). Default = "CausalImagesEnv".
conda_env_required	A Boolean stating whether use of the specified conda environment is required.
figuresTag	A string specifying an identifier that is appended to all figure names.
figuresPath	A string specifying file path for saved figures made in the analysis.
plotBands	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 RGB plotting).
plotResults	Should analysis results be plotted?
optimizeImageRep	Boolean specifying whether to optimize over the image model representation (or only over downstream parameters).

`nWidth_ImageRep`
: Integer specifying width of image model representation.

`nDepth_ImageRep`
: Integer specifying depth of image model representation.

`nWidth_Dense`
: Integer specifying width of image model representation.

`nDepth_Dense`
: Integer specifying depth of dense model representation.

`strides`
: Integer specifying the strides used in the convolutional layers.=

`kernelSize`
: Dimensions used in spatial convolutions.

`nSGD`
: Number of stochastic gradient descent (SGD) iterations.

`batchSize`
: Batch size used in SGD optimization.

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

`nMonte_salience`
: 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).

`nMonte_variational`
: An integer specifying how many Monte Carlo iterations to use in the calculation of the expected likelihood in each training step.

`TfRecords_BufferScaler`
: The buffer size used in `tfrecords` mode is `batchSize*TfRecords_BufferScaler`. Lower `TfRecords_BufferScaler` towards 1 if out-of-memory problems.

`dataType`
: String specifying whether to assume `"image"` or `"video"` data types.

Value

Returns a list consiting of

- `clusterTaus_mean` default
- `clusterProbs_mean`. Estimated mean image effect cluster probabilities.
- `clusterTaus_sigma`. Estimated cluster 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/
```

`BuildBackend` *Build the environment for CausalImages models. Builds a conda environment in which jax, tensorflow, tensorflow-probability optax, equinox, and jmp are installed.*

Description

Build the environment for CausalImages models. Builds a conda environment in which jax, tensorflow, tensorflow-probability optax, equinox, and jmp are installed.

Usage

```
BuildBackend(conda_env = "CausalImagesEnv", conda = "auto")
```

Arguments

conda_env	(default = "CausalImagesEnv") Name of the conda environment in which to place the backends.
conda	(default = auto) The path to a conda executable. Using "auto" allows reticulate to attempt to automatically find an appropriate conda binary.

Value

Builds the computational environment for causalimages. This function requires an Internet connection. You may find out a list of conda Python paths via: system("which python")

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,
  image_pixel_width = 256L,
  save_folder = ".",
```

```
  save_as = "csv",
  lyrs = NULL
)
```

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)
```

GetElementFromTfRecordAtIndices

Reads unique key indices from a .tfrecord file.

Description

Reads unique key indices from a .tfrecord file saved via a call to causalimages::WriteTfRecord.

Usage

```
GetElementFromTfRecordAtIndices(uniqueKeyIndices, file,
    conda_env, conda_env_required)
```

Arguments

uniqueKeyIndices	(integer vector) Unique image indices to be retrieved from a .tfrecord
conda_env	(Default = NULL) A conda environment where tensorflow v2 lives. Used only if a version of tensorflow is not already active.
conda_env_required	(default = F) A Boolean stating whether use of the specified conda environment is required.
file	(character string) A character string stating the path to a .tfrecord

Value

Returns content from a .tfrecord associated with uniqueKeyIndices

Examples

```
# Example usage (not run):
#GetElementFromTfRecordAtIndices(
  #uniqueKeyIndices = 1:10,
  #file = "./NigeriaConfoundApp.tfrecord")
```

GetImageRepresentations

Generates image and video representations useful in earth observation tasks for casual inference.

Description

Generates image and video representations useful in earth observation tasks for casual inference, following the approach in Rolf, Esther, et al. (2021).

Usage

```
GetImageRepresentations(
  imageKeysOfUnits = NULL,
  file = NULL,
  conda_env = "CausalImagesEnv",
  conda_env_required = T,
  returnContents = T,
  getRepresentations = T,
  imageModelClass = "VisionTransformer",
  NORM_MEAN = NULL,
  NORM_SD = NULL,
  Sys.setenv_text = NULL,
  InitImageProcess = NULL,
  pretrainedModel = NULL,
  lat = NULL,
  long = NULL,
  image_dtype = NULL,
```

```
  image_dtype_tf = NULL,
  nWidth_ImageRep = 64L,
  nDepth_ImageRep = 1L,
  nDepth_TemporalRep = 1L,
  batchSize = 16L,
  optimizeImageRep = T,
  strides = 1L,
  kernelSize = 3L,
  patchEmbedDim = 16L,
  TfRecords_BufferScaler = 10L,
  dropoutRate,
  dataType = "image",
  bn_momentum = 0.99,
  inputAvePoolingSize = 1L,
  CleanupEnv = FALSE,
  initializingFxns = FALSE,
  seed = NULL
)
```

Arguments

imageKeysOfUnits	A vector of length length(imageKeysOfUnits) specifying the unique image ID associated with each unit. Samples of imageKeysOfUnits are fed into the package to call images into memory.
file	Path to a tfrecord file generated by causalimages::WriteTfRecord.
conda_env	A conda environment where computational environment lives, usually created via causalimages::BuildBackend(). Default = "CausalImagesEnv"
conda_env_required	A Boolean stating whether use of the specified conda environment is required.
InitImageProcess	(default = NULL) Initial image processing function. Usually left NULL.
nWidth_ImageRep	Number of embedding features output.
batchSize	Integer specifying batch size in obtaining representations.
strides	Integer specifying the strides used in the convolutional layers.
kernelSize	Dimensions used in the convolution kernels.
TfRecords_BufferScaler	The buffer size used in tfrecords mode is batchSize*TfRecords_BufferScaler. Lower TfRecords_BufferScaler towards 1 if out-of-memory problems.
dataType	String specifying whether to assume "image" or "video" data types. Default is "image".

Value

A list containing two items:

- Representations (matrix) A matrix containing image/video representations, with rows corresponding to observations.
- ImageRepArm_OneObs, ImageRepArm_batch_R, ImageRepArm_batch (functions) Image modeling functions.

- `ImageModel_And_State_And_MPPolicy_List` List containing image model parameters fed into functions.

References

- Rolf, Esther, et al. "A generalizable and accessible approach to machine learning with global satellite imagery." *Nature Communications* 12.1 (2021): 4392.

Examples

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

`GetMoments` *Get moments for normalization*

Description

A function obtaining moments for normalization

Usage

```
GetMoments(iterator, dataType, image_dtype, momentCalIters = 34L)
```

Arguments

`iterator`	An iterator
`dataType`	A string denoting data type
`momentCalIters`	Number of minibatches with which to estimate moments

Value

Returns mean/sd normalization arrays.

Examples

```
# (Not run)
# GetMoments(iterator, dataType, image_dtype, momentCalIters = 34L)
```

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,
  xaxt = NULL,
  yaxt = NULL,
  xlab = "",
  ylab = "",
  main = NULL,
  cex.main = NULL,
  col.lab = "black",
  col.main = "black",
  cex.lab = 1.5,
  box = F
)
```

Arguments

x	The numeric matrix to be visualized.
xaxt	The x-axis tick labels.
yaxt	The y-axis tick labels.
xlab	The x-axis labels.
ylab	The y-axis labels.
main	The main figure label.
cex.main	The main figure label sizing factor.
col.lab	Axis label color.
col.main	Main label color.
cex.lab	Cex for the labels.
box	Draw a box 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

```
# (Not run)
#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"))
```

---

print2 *print2*

Description

A function prints a string with date and time.

Usage

```
print2(text, quiet = F)
```

Arguments

x	Character string to be printed, with date and time.

Value

Prints with date and time.

Examples

```
message("Hello world")
```

`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 )
```

`TrainDefine` *Defines a trainer.*

Description

Defines trainers defined in TrainDefine().

Usage

```
TrainDefine()
```

Arguments

.	No parameters.

Value

Defines a training sequence

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/
```

TrainDo *Runs a trainer.*

Description

Runs trainers defined in TrainDefine().

Usage

```
TrainDo()
```

Arguments

.	No parameters.

Value

Performs training.

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/
```

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,
  uniqueImageKeys,
  acquireImageFxn,
  writeVideo = F,
  image_dtype = "float16",
  conda_env = "CausalImagesEnv",
  conda_env_required = T,
  Sys.setenv_text = NULL
)
```

Arguments

file	A character string naming a file for writing.
uniqueImageKeys	A vector specifying the unique image keys of the corpus. A key grabs an image/video array via acquireImageFxn(key)
acquireImageFxn	A function whose input is an observation index and whose output is an image.
writeVideo	(default = FALSE) Should we assume we're writing image sequence data of form batch by time by height by width by channels?
conda_env	(default = "CausalImagesEnv") A conda environment where computational environment lives, usually created via causalimages::BuildBackend()
conda_env_required	(default = T) A Boolean stating whether use of the specified conda environment is required.

Value

Writes a unique key-referenced .tfrecord from an image/video corpus for use in image-based causal inference training.

Examples

```
# Example usage (not run):
#WriteTfRecord(
#  file = "./NigeriaConfoundApp.tfrecord",
#  uniqueImageKeys = 1:n,
#  acquireImageFxn = acquireImageFxn)
```

Index