Package 'animl'

February 12, 2025

Title A Collection of ML Tools for Conservation Research

Version 2.0.0

Description Functions required to classify subjects within camera trap field data. The package can handle both images and videos. The authors recommend a two-step approach using Microsoft's 'MegaDector' model and then a second model trained on the classes of interest.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports grDevices, methods, pbapply, dplyr, jpeg, reticulate, parallel, exifr, av, magrittr, stats, rlang,

Depends R (>= 4.0.0)

NeedsCompilation no

Author Kyra Swanson [aut, cre] (<https://orcid.org/0000-0002-1496-3217>), Mathias Tobler [aut]

Maintainer Kyra Swanson <tswanson@sdzwa.org>

Repository CRAN

Date/Publication 2025-02-11 23:10:06 UTC

Contents

`build_file_manifest` *File Management Module*

Description

This module provides functions and classes for managing files and directories.

Usage

```
build_file_manifest(
  image_dir,
  exif = TRUE,
  out_file = NULL,
  offset = 0,
  recursive = TRUE
)
```

Arguments

`image_dir`	folder to search through and find media files
`exif`	returns date and time information from exif data, defaults to true
`out_file`	directory to save .csv of manifest to
`offset`	add offset in hours for videos when using the File Modified date, defaults to 0
`recursive`	Should directories be scanned recursively? Default TRUE

Details

Kyra Swanson 2023 Find Image/Video Files and Gather exif Data

Value

files dataframe with or without file dates

Examples

```
## Not run:
files <- build_file_manifest("C:\\Users\\usr\\Pictures\\")

## End(Not run)
```

`check_file` *Check for files existence and prompt user if they want to load*

Description

Check for files existence and prompt user if they want to load

Usage

```
check_file(file)
```

Arguments

`file`	the full path of the file to check

Value

a boolean indicating wether a file was found and the user wants to load or not

Examples

```
## Not run:
  checkFile("path/to/newfile.csv")

## End(Not run)
```

`crop_images` *Crops all images from an input file with specific required columns: Frame, bbox1, bbox2, bbox3, and bbox4*

Description

Crops all images from an input file with specific required columns: Frame, bbox1, bbox2, bbox3, and bbox4

Usage

```
crop_images(imagelist, outdir)
```

Arguments

imagelist	The path for the input csv file
outdir	The path where generated cropped images should be uploaded

Value

no return value, outputs the cropped image

Examples

```
## Not run:
cropImagesFromFile("/image/path/file.csv", "/output/path/")

## End(Not run)
```

---

detect_MD_batch *Apply MegaDetector to a Given Batch of Images*

Description

Apply MegaDetector to a Given Batch of Images

Usage

```
detect_MD_batch(
  detector,
  image_file_names,
  checkpoint_path = NULL,
  checkpoint_frequency = -1,
  confidence_threshold = 0.1,
  quiet = TRUE,
  image_size = NULL,
  file_col = "Frame"
)
```

Arguments

detector	preloaded md model
image_file_names	list of image filenames, a single image filename, or folder
checkpoint_path	path to checkpoint file
checkpoint_frequency	write results to checkpoint file every N images
confidence_threshold	only detections above this threshold are returned

quiet	print debugging statements when false, defaults to true
image_size	overrides default image size, 1280
file_col	select which column if image_file_names is a manifest

Value

list of dictionaries of MegaDetector detections

Examples

```
## Not run: mdres <- detectMD_batch(md_py, allframes$Frame)
```

`extract_frames` *Extract frames from video for classification*

Description

This function can take

Usage

```
extract_frames(
  files,
  out_dir = tempfile(),
  out_file = NULL,
  fps = NULL,
  frames = NULL,
  file_col = "FilePath",
  parallel = FALSE,
  workers = 1,
  checkpoint = 1000
)
```

Arguments

files	dataframe of videos
out_dir	directory to save frames to
out_file	file to which results will be saved
fps	frames per second, otherwise determine mathematically
frames	number of frames to sample
file_col	string value indexing which column contains file paths
parallel	Toggle for parallel processing, defaults to FALSE
workers	number of processors to use if parallel, defaults to 1
checkpoint	if not parallel, checkpoint ever n files, defaults to 1000

Value

dataframe of still frames for each video

Examples

```
## Not run:
frames <- extractFrames(videos, out_dir = "C:\\Users\\usr\\Videos\\", frames = 5)

## End(Not run)
```

`extract_frames_old` *Extract frames from video for classification*

Description

This function can take

Usage

```
extract_frames_old(
  files,
  out_dir = tempfile(),
  out_file = NULL,
  fps = NULL,
  frames = NULL,
  file_col = "FilePath",
  parallel = FALSE,
  workers = 1,
  checkpoint = 1000
)
```

Arguments

`files`	dataframe of videos
`out_dir`	directory to save frames to
`out_file`	file to which results will be saved
`fps`	frames per second, otherwise determine mathematically
`frames`	number of frames to sample
`file_col`	string value indexing which column contains file paths
`parallel`	Toggle for parallel processing, defaults to FALSE
`workers`	number of processors to use if parallel, defaults to 1
`checkpoint`	if not parallel, checkpoint ever n files, defaults to 1000

Value

dataframe of still frames for each video

Examples

```
## Not run:
frames <- extractFrames(videos, out_dir = "C:\\Users\\usr\\Videos\\", frames = 5)

## End(Not run)
```

`extract_frame_single` *Extract Frames for Single Video*

Description

Extract Frames for Single Video

Usage

```
extract_frame_single(file_path, out_dir, fps = NULL, frames = NULL)
```

Arguments

`file_path`	filepath to image
`out_dir`	directory to save frames to
`fps`	number of frames per second to save
`frames`	number of frames evenly distributed to save

Value

dataframe of filepaths, frame paths

Examples

```
## Not run:
result <- extractFramesSingle(video$FilePath, out_dir, frames=3)

## End(Not run)
```

get_animals *Return a dataframe of only MD animals*

Description

Return a dataframe of only MD animals

Usage

```
get_animals(manifest)
```

Arguments

manifest	all megadetector frames

Value

animal frames classified by MD

Examples

```
## Not run:
animals <- getAnimals(imagesall)

## End(Not run)
```

get_empty *Return MD empty, vehicle and human images in a dataframe*

Description

Return MD empty, vehicle and human images in a dataframe

Usage

```
get_empty(manifest)
```

Arguments

manifest	all megadetector frames

Value

list of empty/human/vehicle allframes with md classification

Examples

```
## Not run:
empty <- getEmpty(imagesall)

## End(Not run)
```

---

load_data *Load .csv or .Rdata file*

Description

Load .csv or .Rdata file

Usage

```
load_data(file)
```

Arguments

file	the full path of the file to load

Value

data extracted from the file

Examples

```
## Not run:
  loadData("path/to/newfile.csv")

## End(Not run)
```

---

load_model *Load a Classifier Model with animl-py*

Description

Load a Classifier Model with animl-py

Usage

```
load_model(model_path, class_file, device = NULL, architecture = "CTL")
```

Arguments

model_path	path to model
class_file	path to class list
device	send model to the specified device
architecture	model architecture

Value

list of c(classifier, class_list)

Examples

```
## Not run: andes <- loadModel('andes_v1.pt','andes_classes.csv')
```

---

megadetector *Load MegaDetector*

Description

Load MegaDetector

Usage

```
megadetector(model_path, device = NULL)
```

Arguments

model_path	path to MegaDetector model (v5)
device	load model onto given device description

Value

megadetector object

Examples

```
## Not run: md_py <- megadetector("/mnt/machinelearning/megaDetector/md_v5a.0.0.pt")
```

parse_MD *parse MD results into a simple dataframe*

Description

parse MD results into a simple dataframe

Usage

```
parse_MD(
  results,
  manifest = NULL,
  out_file = NULL,
  buffer = 0.02,
  threshold = 0,
  file_col = "Frame"
)
```

Arguments

results	json output from megadetector
manifest	dataframe containing all frames
out_file	path to save dataframe
buffer	percentage buffer to move bbox away from image edge
threshold	confidence threshold to include bbox
file_col	column in manifest that refers to file paths

Value

original dataframe including md results

Examples

```
## Not run:
mdresults <- parseMD(mdres)

## End(Not run)
```

plot_boxes *Plot bounding boxes on image from md results*

Description

Plot bounding boxes on image from md results

Usage

```
plot_boxes(image, label = FALSE, minconf = 0)
```

Arguments

image	The mdres for the image
label	T/F toggle to plot MD category
minconf	minimum confidence to plot box

Value

no return value, produces bounding box in plot panel

Examples

```
## Not run:
mdres <- classifyImageMD(mdsession, images$FilePath[30000])
plotBoxes(mdres, minconf = 0.5)

## End(Not run)
```

predict_species *Infer Species for Given Detections*

Description

Infer Species for Given Detections

Usage

```
predict_species(
  detections,
  model,
  classes,
  device = NULL,
  out_file = NULL,
  raw = FALSE,
  file_col = "Frame",
```

```
  crop = TRUE,
  resize_width = 299,
  resize_height = 299,
  normalize = TRUE,
  batch_size = 1,
  workers = 1
)
```

Arguments

detections	manifest of animal detections
model	loaded classifier model
classes	data.frame of classes
device	send model to the specified device
out_file	path to csv to save results to
raw	output raw logits in addition to manifest
file_col	column in manifest containing file paths
crop	use bbox to crop images before feeding into model
resize_width	image width input size
resize_height	image height input size
normalize	normalize the tensor before inference
batch_size	batch size for generator
workers	number of processes

Value

detection manifest with added prediction and confidence columns

Examples

```
## Not run: animals <- predictSpecies(animals, classifier[[1]], classifier[[2]], raw=FALSE)
```

---

remove_link *Remove Sorted Links*

Description

Remove Sorted Links

Usage

```
remove_link(manifest, link_col = "Link")
```

Arguments

manifest	DataFrame of classified images
link_col	column in manifest that contains link paths

Value

manifest without link column

Examples

```
## Not run:
remove_link(manifest)

## End(Not run)
```

---

save_data *Save Data to Given File*

Description

Save Data to Given File

Usage

```
save_data(data, out_file, prompt = TRUE)
```

Arguments

data	the dataframe to be saved
out_file	the full path of the saved file
prompt	if true, prompts the user to confirm overwrite

Value

none

Examples

```
## Not run:
 saveData(files,"path/to/newfile.csv")

## End(Not run)
```

`sequence_classification` *Leverage sequences to classify images*

Description

This function applies image classifications at a sequence level by leveraging information from multiple images. A sequence is defined as all images at the same camera/station where the time between consecutive images is <=maxdiff. This can improve classification accuracy, but assumes that only one species is present in each sequence. If you regularly expect multiple species to occur in an image or sequence don't use this function.

Usage

```
sequence_classification(
  animals,
  empty = NULL,
  predictions,
  classes,
  stationcolumn,
  emptyclass = "",
  sortcolumns = NULL,
  recordfield = "FilePath",
  maxdiff = 60
)
```

Arguments

`animals`	sub-selection of all images that contain MD animals
`empty`	optional, data frame non-animal images (empty, human and vehicle) that will be merged back with animal imagages
`predictions`	data frame of prediction probabilities from the classifySpecies function
`classes`	a vector or species corresponding to the columns of 'predictions'
`stationcolumn`	a column in the animals and empty data frame that indicates the camera or camera station
`emptyclass`	a string indicating the class that should be considered 'Empty'
`sortcolumns`	optional sort order. The default is 'stationcolumnumn' and DateTime.
`recordfield`	a field indicating a single record. The default is FilePath for single images/videos.
`maxdiff`	maximum difference between images in seconds to be included in a sequence, defaults to 60

Details

This function retains "Empty" classification even if other images within the sequence are predicted to contain animals. Classification confidence is weighted by MD confidence.

Value

data frame with predictions and confidence values for animals and empty images

Examples

```
## Not run:
predictions <-classifyCropsSpecies(images,modelfile,resize=456)
animals <- allframes[allframes$max_detection_category==1,]
empty <- setEmpty(allframes)
animals <- sequenceClassification(animals, empty, predictions, classes,
                                  emptyclass = "Empty",
                                  stationcolumnumn="StationID", maxdiff=60)

## End(Not run)
```

---

sort_MD *Create SymLink Directories and Sort Classified Images Based on MD Results*

Description

Create SymLink Directories and Sort Classified Images Based on MD Results

Usage

```
sort_MD(
  manifest,
  link_dir,
  file_col = "FilePath",
  unique_name = "UniqueName",
  copy = FALSE
)
```

Arguments

manifest	DataFrame of classified images
link_dir	Destination directory for symlinks
file_col	Colun containing file paths
unique_name	Unique image name identifier
copy	Toggle to determine copy or hard link, defaults to link

Value

manifest with added link columns

Examples

```
## Not run:
sort_MD(manifest, link_dir)

## End(Not run)
```

`sort_species` *Create SymLink Directories and Sort Classified Images*

Description

Create SymLink Directories and Sort Classified Images

Usage

```
sort_species(
  manifest,
  link_dir,
  file_col = "FilePath",
  unique_name = "UniqueName",
  copy = FALSE
)
```

Arguments

`manifest`	DataFrame of classified images
`link_dir`	Destination directory for symlinks
`file_col`	Colun containing file paths
`unique_name`	Unique image name identifier
`copy`	Toggle to determine copy or hard link, defaults to link

Value

manifest with added link columns

Examples

```
## Not run:
manifest <- sort_species(manifest, link_dir)

## End(Not run)
```

update_labels *Udate Results from File Browser*

Description

Udate Results from File Browser

Usage

```
update_labels(manifest, link_dir, unique_name = "UniqueName")
```

Arguments

manifest	dataframe containing file data and predictions
link_dir	directory to sort files into
unique_name	column name indicating a unique file name for each row

Value

dataframe with new "Species" column that contains the verified species

Examples

```
## Not run:
results <- updateResults(resultsfile, linkdir)

## End(Not run)
```

WorkingDirectory *Set Working Directory and Save File Global Variables*

Description

Set Working Directory and Save File Global Variables

Usage

```
WorkingDirectory(workingdir, pkg.env)
```

Arguments

workingdir	local directory that contains data to process
pkg.env	environment to create global variables in

Value

None

Examples

```
## Not run:
WorkingDirectory(/home/kyra/animl/examples)

## End(Not run)
```

Index