Adapting and Translating Annotation Datasets

robert.rettig@dfki.de

2024-11-01

This R Markdown file explains the steps necessary to adapt and transfer the JSON COCO Format to other categories and formats. This Script does not address necessary preprocessing steps to work with multiscale data. It is recommended to use Open Source GIS Software like QGIS (https://qgis.org/), to georeference pictures and orthomosaics taken by drones (https://docs.qgis.org/3.34/en/docs/user_manual/working_with_raster/georeferencer.html), for further processing. For satellite based images please consider AROSICS for coregistration (https://git.gfz-potsdam.de/danschef/arosics/).

This Markdown Script is using an example image from a field campaign which took place in summer 2024. The rest of the dataset will be published soon, so the scripts can be performed on multiple Images and Annotation files.

Please make sure to install the following packages before running the script: “terra”, “rjson”, “sf”, “dplyr”, “stringr”

```
json_file <-
  "Example_File/JSON_COCO/annotations/instances_default.json"
json_data <- rjson::fromJSON(file = json_file)
json_data_edit <- json_data


image <-
  terra::rast("Example_File/JSON_COCO/images/2024-06-27_Plot_70.png")
print(json_data_edit[["images"]][[1]][["file_name"]])
```

```
## [1] "2024-06-27_Plot_70.png"
```

```
terra::plot(image)
```

Cluster annotations to binary classification

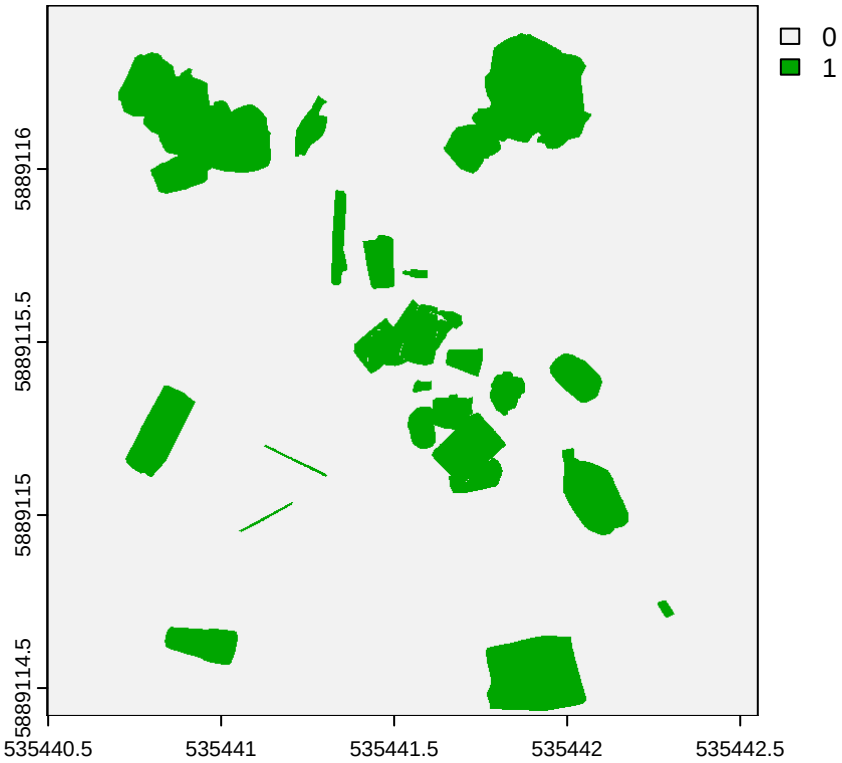
This process converts the annotations to a binary classification system, where all annotations are grouped under a single category named “objects”

```
for (n in seq_along(json_data_edit[["annotations"]])) {
  json_data_edit[["annotations"]][[n]][["category_id"]] <- 1
}

# delete other categories, which are not further used
json_data_edit[["categories"]][c(2:length(json_data_edit[["categories"]]))] <-
  NULL

# name the category according to the binary classification
json_data_edit[["categories"]][[1]][["name"]] <- c('objects')

# convert to JSON
exportJson <- toJSON(json_data_edit)

# Save the JSON to file
write(exportJson, file = "Example_File/JSON_COCO/annotations/instances_default_binary.json")
```

Create a binary raster mask

Based on the created binary annotations, herewith a binary raster mask and polygons is created, for further extraction of pixel values e.g. for pixel wise classifications. Please provide the world file (e.g. .aux.xml) or the georeferenced original *.tif to use the georeference information for further processing.

```
## [1] "Mask image: Nr.1, with name: 2024-06-27_Plot_70.png and ID 27. With a total of 25 annotations"
```



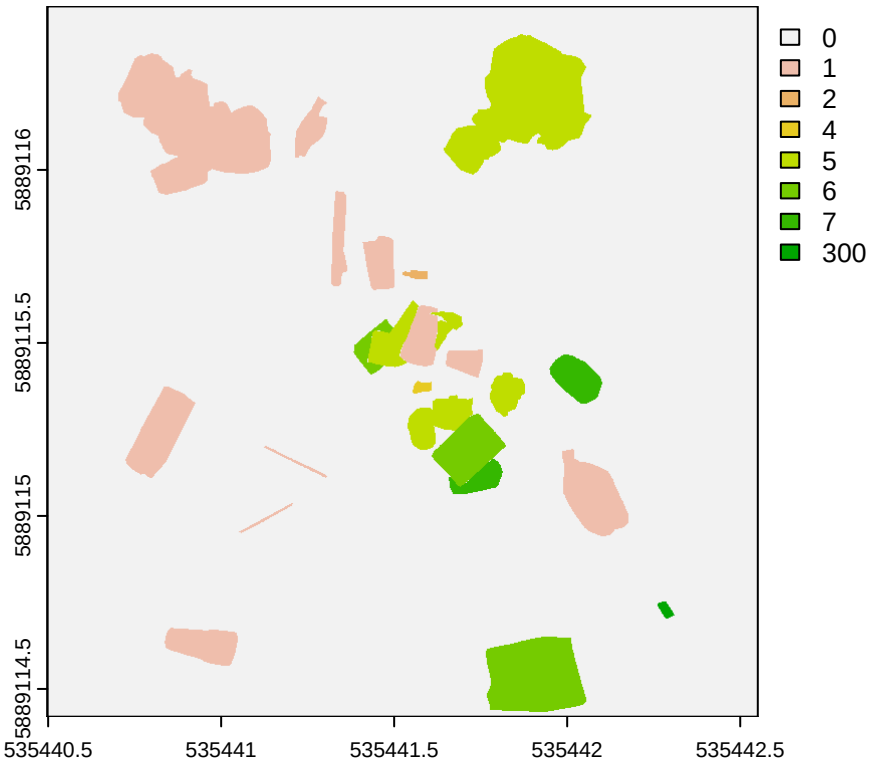
Cluster annotations to material type categories

We created a prelimenary version of a LUT to address most common objects and material types, materials and usage. While clustering our annotations to material types, we will use this list and according ID's as guidance.

```
## [1] "Resulting materialtype categories for the annotation file: "
```

```
## [1] "Plastic"
## [1] "Metal"
## [1] "Cardboard"
## [1] "Organic"
## [1] "Others"
## [1] "Glas"
## [1] "Paper"
```

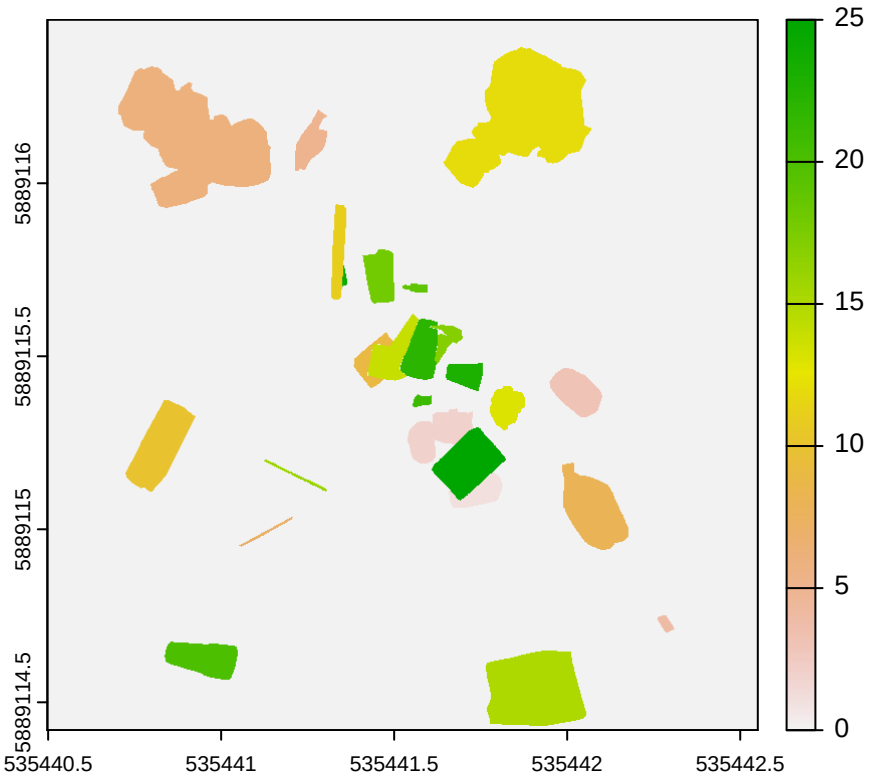
```
## [1] "Mask image: Nr.1, with name: 2024-06-27_Plot_70.png and ID 27. With a total of 25 annotations"
```



For Panoptic Segmentation: Create a unique ID for each instance of the materialtype categorie

For the panoptic segmentation, a layer of the initial annotated categories is needed, as well as a layer, for unique identification of each instance.

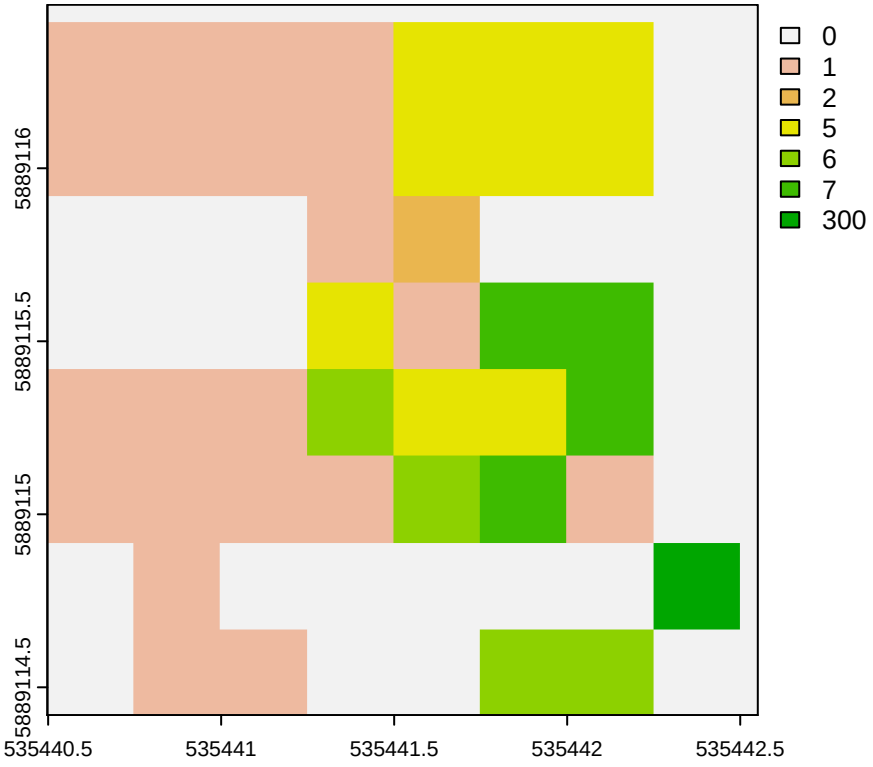

```
## [1] "Mask image: Nr.1, with name: 2024-06-27_Plot_70.png and ID 27. With a total of 25 annotations"
```



For Tilewise Classification: Create a tile for the dominant materialtype

For the panoptic segmentation, a layer of the initial annotated categories is needed, as well as a layer, for unique identification of each instance.

```
## [1] "Mask image: Nr.1, with name: 2024-06-27_Plot_70.png and ID 27. With a total of 25 annotations"
```


0
1
2
5
6
7
300
5889116
5889115.5
5889115
5889114.5
535440.5
535441
535441.5
535442
535442.5