REDD+ Feasability Assessment

Review of emissions estimates & activity data used in the RSPB Gola feasibility assessment

2024-12-23

Contents

Summary

Project files reviewed

Filename	Filepath
community forestry zone REDD objectives.docx	~/20087 - RSPB Gola Feasibility/Working Files/Gola VER estimate/
ER_Workbook_Gola_Liberia.xlsx	~/20087 - RSPB Gola Feasibility/Working Files/Gola VER estimate/
ERR_assumptions_meeting_notes_final.docx[1]	~/20087 - RSPB Gola Feasibility/Working Files/Gola VER estimate/
Proxy Baseline Allocation Report.docx	~/20087 - RSPB Gola Feasibility/Working Files/Gola VER estimate/
VER Notes.docx	~/20087 - RSPB Gola Feasibility/Working Files/Gola VER estimate/
10b Gola REDD Baseline Workplan VCS.pdf	~/20087 - RSPB Gola Feasibility/Working Files/Data from RSPB/OneDrive_1_02-05-2024.zip
ProjectArea.shp	~/20087 - RSPB Gola Feasibility/Working Files/Winrock_GIS Analysis Gola/

Import aoi

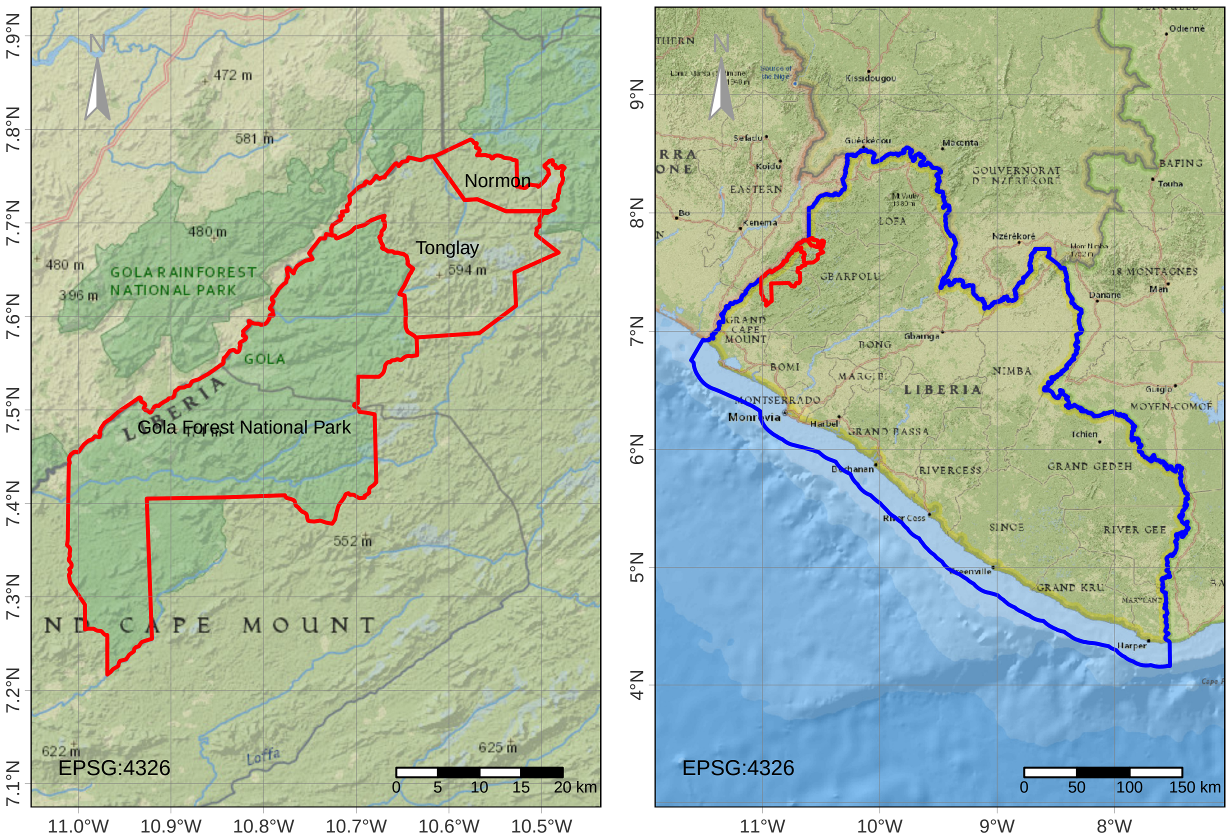
[1] "RSPB is concerned that the Verra baselines will be inaccurate because they will use: existing freely available global data like ESA (European Space Agency) landcover and Hansen Global Forest Change to measure forest cover. These overestimate the amount of native forest that exists because these data sources can't distinguish between native forest and some other habitats such as agroforest in the Greater Gola Landscape (GGL), despite doing so effectively elsewhere (Brittany has verified this in the GGL). This is because the GGL is fine-grained (small patches of rotational swidden agriculture) and agroforestry often has dense canopy cover, which means only satellites collecting very high-resolution images can detect small habitat patches and small gaps in canopy cover. This is important because although the habitats can look similar, the above-ground carbon is far less than that of forest."

```r
aoi = sf::read_sf("~/OneDrive - Winrock International Institute for Agricultural Development/20087 - RSPB Gola Feasibility/Working Files/Winrock_GIS Analysis Gola/ProjectArea.shp") |> sf::st_transform(3857) # supports mosaicking across multiple UTMs
aoi = aoi |>
  sf::st_cast("POLYGON") |>
  sf::st_cast("MULTIPOLYGON") |>
  dplyr::filter(
   NAME    == "Gola Forest National Park" |
     NAME == "Tonglay" |
     NAME == "Normon")

query = osmdata::opq("Liberia") |> # opq = overpass query
  osmdata::add_osm_feature(key = "boundary", value = "administrative") |>
  osmdata::osmdata_sf()

boundaries = query$osm_multipolygons
country  = boundaries |>
  dplyr::filter(admin_level == "2", name == "Liberia") |>
  dplyr::select(name, admin_level, geometry) |>
  sf::st_cast() |>
  sf::st_transform(3857)
sf::st_write(country, "./data/aoi/liberia_boundary_national.shp", delete_layer
   =T)

bbox_country_1 = terra::vect(terra::ext(terra::vect(country)) * 1.1)
bbox_country_2 = terra::vect(terra::ext(terra::vect(country)) * 1.6)
bbox_aoi_1    = terra::vect(terra::ext(terra::vect(aoi)) * 1.1)
bbox_aoi_2    = terra::vect(terra::ext(terra::vect(aoi)) * 1.6)
bbox_aoi      = sf::st_as_sf(bbox_aoi_1)
bbox_country = sf::st_as_sf(bbox_country_1)
terra::crs(bbox_aoi_1)    = "epsg:3857"
terra::crs(bbox_aoi_2)    = "epsg:3857"
terra::crs(bbox_country_1)  = "epsg:3857"
terra::crs(bbox_country_2)  = "epsg:3857"
sf::st_crs(bbox_aoi)      = 3857
sf::st_crs(bbox_country) = 3857
```



Area check

In Liberia, the official definition of forest land is provided by the Forestry Development Authority (Liberia 2019), including areas of land that meet the following criteria:

- Canopy cover of minimum 30%;
- Canopy height of minimum 5m or the capacity to reach it;
- Covering a minimum of 1 hectare of land.

```
aoi$area_m2 = round(as.numeric(st_area(aoi) * 0.0001, 4))
aoi_select = aoi |>
  dplyr::select(NAME, DESIG, area_m2) |>
  dplyr::filter(
    NAME == "Gola Forest National Park" |
      NAME == "Tonglay" |
      NAME == "Normon") |>
  sf::st_drop_geometry() |>
  janitor::adorn_totals() |>
  knitr::kable(font_size = 7) |>
  kable_styling("striped", full_width = F)
aoi_select
```

NAME	DESIG	area_m2
Gola Forest National Park	National Park	90922
Tonglay	Community Forest	30247

Normon	Community Forest	7170
Total	-	128339

```
# check for artefacts or "forest slivers"
slivers = aoi |>
  dplyr::filter(as.numeric(area_m2) < 1)
slivers # no slivers found
```

```
Simple feature collection with 0 features and 30 fields
Bounding box:  xmin: NA ymin: NA xmax: NA ymax: NA
Projected CRS: WGS 84 / Pseudo-Mercator
# A tibble: 0 x 31
# i 31 variables: WDPAID <dbl>, WDPA_PID <chr>, NAME <chr>, ORIG_NAME <chr>,
#   DESIG <chr>, DESIG_TYPE <chr>, IUCN_CAT <chr>, INT_CRIT <chr>,
#   MARINE <chr>, REP_M_AREA <dbl>, GIS_M_AREA <dbl>, REP_AREA <dbl>,
#   GIS_AREA <dbl>, STATUS <chr>, STATUS_YR <dbl>, GOV_TYPE <chr>,
#   MANG_AUTH <chr>, MANG_PLAN <chr>, VERIF <chr>, METADATAID <dbl>,
#   SUB_LOC <chr>, PARENT_ISO <chr>, ISO3 <chr>, Comments <chr>,
#   Landscape <chr>, Shape_Leng <dbl>, Shape_Area <dbl>, Areaha <dbl>, ...
```

Results confirm the dataset is free from forest patches that are smaller than the approved area definition.

LULC check

Data processing

- raster normalization applied cloudless pixel ranking & median back-fill;
- baseline beyond temporal extent of sentinel (**landsat used instead?**);
- training sample extracted from GLanCE dataset[2], which was processed using **class migration** algorithm;
  - Although Verra lacking requirements for class migration (i.e. VT0007, VMD0055, VM0048), we may advise client on best practices and showcase improved accuracy in following comparisons (Verra 2023, 2024, 2021).
  - Level-1 classes in the GLanCE dataset were recoded to match feature labels reported in the "Lookups" sheet of "ER_Workbook_Gola_Liberia.xlsx". For review, the following table compares GLanCE's data dictionary and Liberia's methodological report to present feature classes in their original format and converted format ("»") (Woodcock et al., n.d.; Liberia 2019).

RSPB classes	GLanCE classes	
Bareground (0)	Barren (4) » Bareground (0)	Areas of soils, sand, or rocks where <10% is vegetated
Regrowth (1)	Herbaceous (7) » Regrowth (1)	Areas of <30% tree, >10% vegetation, but <10% shrub
Farmbush (2)	Shrublands (6) » Farmbush (2)	Areas of <30% tree, >10% vegetation, & >10% shrub
Forest (3)	Tree Cover (5) » Forest (3)	Areas of tree cover > 30%.
Water (4)	Water (1) » Water (4)	Areas covered with water year-round (lakes & streams)
	Developed (3) » Urban (99)	Areas covered with structures, built-up
	Ice/Snow (2) » Ice/Snow (88)	Areas of snow cover > 50% year-round
Swamp (5)		

[2] For replication, the full unprocessed dataset was stored in personal drive folder here: https://drive.google.com/file/d/1FhWTpSGFRTodDCY2gSGhssLuP2Plq4ZE/view?usp=drive_link - To extract from source, java script and google earth engine account needed.

RSPB classes	GLanCE classes
Cocoa (6)	
Oil Palm (7)	

Training samples were fitted to a Random Forest model, which was post-processed with a Bayesian smoothing, and evaluated with confusion matrix and probabilistic uncertainty estimates of pixel classification between vectorized class boundaries. To strengthen land classifiers, new samples were then added to areas of high pixel uncertainty, before land cover was re-classified and re-evaluated with a second round of confusion matrix.

Training samples

```r
# filter training dataset spatiall & temporally and
# relabel for use in `sits` & `ForestToolsRS` functions
samples_raw = read.csv("./data/training_samples/glance_training.csv")

samples_clean = samples_raw |>
  dplyr::select(Lon, Lat, Glance_Class_ID_level1, Start_Year, End_Year) |>
  dplyr::rename(longitude = Lon) |>
  dplyr::rename(latitude = Lat) |>
  dplyr::rename(label_old = Glance_Class_ID_level1) |>
  dplyr::mutate(start_date = as.Date(paste(Start_Year, "01", "01", sep = "-"))
    ) |>
  dplyr::mutate(end_date = as.Date(paste(End_Year, "01", "01", sep = "-"))) |>
  dplyr::select(longitude, latitude, start_date, end_date, label_old) |>
  dplyr::mutate(code = case_when(
    label_old == '4' ~ 0,
    label_old == '7' ~ 1,
    label_old == '6' ~ 2,
    label_old == '5' ~ 3,
    label_old == '1' ~ 4,
    label_old == '3' ~ 99,
    label_old == '2' ~ 88)
    ) |>
  dplyr::mutate(label = case_when(
    code == '0'  ~ "Bareground",
    code == '1'  ~ "Regrowth",
    code == '2'  ~ "Farmbush",
    code == '3'  ~ "TreeCover",
    code == '4'  ~ "Water",
    code == '99' ~ "Urban",
    code == '88' ~ "Snow")
    ) |>
  dplyr::select(-label_old)
samples_clean$label = base::as.factor(samples_clean$label)

samples_points = sf::st_as_sf(
  samples_clean,
  crs    = 4326,
  coords = c("longitude", "latitude")) |>
  sf::st_transform(3857)

samples_clipped  = sf::st_intersection(samples_points, country) # n = 364
samples_country  = samples_points[samples_clipped, ]
samples_baseline = samples_country |>
```

```
  dplyr::filter(start_date < "2014-01-01" | end_date > "2014-01-01") # n = 121

dplyr::glimpse(samples_country)
dplyr::glimpse(samples_baseline)
sf::st_write(samples_country, "./data/training_samples/glance_spatial_clip.shp
    ", delete_layer = T)
sf::st_write(samples_baseline, "./data/training_samples/glance_temporal_clip.
    shp", delete_layer = T)
#dplyr::glimpse(samples_country)
#dplyr::glimpse(samples_baseline)
dplyr::count(samples_country, label)

tmap::tmap_mode("view")
tmap::tm_shape(country) + tmap::tm_borders(col = "blue", lwd = 2) +
  tmap::tm_shape(aoi) + tmap::tm_borders(col = "red", lwd = 1) +
  tmap::tm_shape(samples_country) +
  tmap::tm_dots(col = "orange") +
  tmap::tm_text("label", just = "right", col="white", size=0.5) +
  tmap::tm_compass(position = c("left", "bottom")) +
  tmap::tm_basemap("Esri.WorldImagery") +
  tmap::tm_scale_bar()
```

Raster collection

Given the project's baseline start date occurs before the launch of the Sentinel 2 satellite, this workflow relied on Landsat data collections. That being said, it remains uncertain what is the required temporal extent of baselines and whether Verra will accept a 6- or 7-year timeline, thereby allowing use of Sentinel data for many projects currently in development. In addition, recent achievements by the Dynamic World V2 data team add to the call for Sentinel based classification, even in the early months of 2015 its initial lifespan.

For sharing and replication, the processed dataset of Landsat Collection-2-Level-2 corrected raster imagery will be temporarily stored in the following google drive folder with download permissions enabled: *Dataset of Processed Rasters Stored Here*

```
# assemble cube from stac
cube_2024_aws = sits_cube(
  source       = "AWS",
  collection   = "LANDSAT-C2-L2",
  roi          = aoi,
  bands        = c("BLUE", "GREEN", "RED", "NIR08", "SWIR16", "SWIR22", "CLOUD"
      ),
  start_date   = "2024-01-01",
  end_date     = "2024-03-01"
  )

# normalize cube
cube_2024_reg = sits_regularize(
  cube         = cube_2024aws,
  res          = 10,
  period       = "P60D",
  multicores   = 16,
  output_dir   = "./data/cube_2024"
  )

# Derive NDVI
cube_2024_spectral <- sits::sits_apply(
```

```
  data = cube_2024_reg,
  NDVI = (NIR08 - RED) / (NIR08 + RED),
  output_dir = './data/cube_2024',
  memsize = 6,
  multicores = 4,
  progress = T
  )

NDVI_2024 = list.files("./data/cube_2024",
  pattern = 'NDVI', full.names = T, all.files = FALSE)|>
  lapply(terra::rast)|>
  sprc() |>
  mosaic()
terra::mask(NDVI_2024, vect(aoi))
aoi = sf::st_transform(aoi, crs(NDVI_2024))
NDVI_2024 = terra::crop(NDVI_2024, vect(aoi), mask=T)
NDVI_2024 = NDVI_2024 * 0.0001
writeRaster(NDVI_2024, "./data/cube_mosaics/NDVI_2024_01.tif", overwrite=T)
```

To save on reading, the above process was repeated behind the scenes for the years 2014 and 2019. All bands were then stacked by year. In the following maps, RGB & NDVI composites were overlaid side-by-side, which can be switched on and off using the small widget in top right of map frame.

```
# reassemble processed rasters
RGB_2014 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....RGB_
    2014-01-01.tif")
RGB_2019 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....RGB_
    2019-01-01.tif")
RGB_2024 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....RGB_
    2024-01-01.tif")
RGB_2014 = raster::stretch(RGB_2014, minv = 0, maxv = 255, minq = 0.1, maxq =
    0.99)
RGB_2019 = raster::stretch(RGB_2019, minv = 0, maxv = 255, minq = 0.1, maxq =
    0.99)
RGB_2024 = raster::stretch(RGB_2024, minv = 0, maxv = 255, minq = 0.1, maxq =
    0.99)

NDVI_2014 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....NDVI_
    2014-01-01.tif")
NDVI_2019 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....NDVI_
    2019-01-01.tif")
NDVI_2024 = terra::rast("./data/cube_mosaics/LANDSAT_TM-ETM-OLI_....NDVI_
    2024-01-01.tif")
hist(NDVI_2014, main = "NDVI Distribution, 2014", col = "springgreen", xlab =
    "Indexed Value")
hist(NDVI_2019, main = "NDVI Distribution, 2019", col = "springgreen", xlab =
    "Indexed Value")
hist(NDVI_2024, main = "NDVI Distribution, 2024", col = "springgreen", xlab =
    "Indexed Value")

tmap::tm_shape(RGB_2014) + tmap::tm_raster(title = "RGB, 2014") +
  tmap::tm_shape(RGB_2019) + tmap::tm_raster(title = "RGB, 2019") +
  tmap::tm_shape(RGB_2024) + tmap::tm_raster(title = "RGB, 2024") +
  tmap::tm_shape(country) + tmap::tm_borders(col = "blue", lwd = 2) +
  tmap::tm_shape(aoi) + tmap::tm_borders(col = "red", lwd = 1) +
```

```
  tmap::tm_basemap("Esri.WorldImagery") +
  tmap::tm_compass(position = c("left", "bottom")) +
  tmap::tm_scale_bar() -> tm6

tmap::tm_shape(NDVI_2014) + tmap::tm_raster(title = "NDVI, 2014") +
  tmap::tm_shape(NDVI_2019) + tmap::tm_raster(title = "NDVI, 2019") +
  tmap::tm_shape(NDVI_2024) + tmap::tm_raster(title = "NDVI, 2024") +
  tmap::tm_shape(country) + tmap::tm_borders(col = "blue", lwd = 2) +
  tmap::tm_shape(aoi) + tmap::tm_borders(col = "red", lwd = 1) +
  tmap::tm_basemap("Esri.WorldImagery") +
  tmap::tm_compass(position = c("left", "bottom")) +
  tmap::tm_scale_bar() -> tm7

tmap::tmap_mode("view")
tmap::tmap_options(max.raster = c(plot = 80000000, view = 100000000)) # expand
    memory
tmap::tmap_arrange(tm6, tm7, ncol = 2)
```

Image classification

```
# extract time series signatures fom NDVI bands
samples_2014 = sits::sits_get_data(NDVI_2014, bands = "NDVI")
samples_2019 = sits::sits_get_data(NDVI_2019, bands = "NDVI")
samples_2024 = sits::sits_get_data(NDVI_2024, bands = "NDVI")

# train classifiers
rfor_model_2014 = sits::sits_train(samples_2014, ml_method = sits_rfor(num_
    trees = 50))
rfor_model_2019 = sits::sits_train(samples_2019, ml_method = sits_rfor(num_
    trees = 50))
rfor_model_2024 = sits::sits_train(samples_2024, ml_method = sits_rfor(num_
    trees = 50))

# classify rasters
cube_2014_prob  = sits_classify(cube_2014_reg, rfor_model_2014, output_dir = "
    ./data/cube_2014", memsize = 16, multicores = 4)
cube_2019_prob  = sits_classify(cube_2019_reg, rfor_model_2014, output_dir = "
    ./data/cube_2019", memsize = 16, multicores = 4)
cube_2024_prob  = sits_classify(cube_2024_reg, rfor_model_2014, output_dir = "
    ./data/cube_2024", memsize = 16, multicores = 4)

# bayesian smoothing
cube_2014_bayes = sits_smooth(cube_2014_prob, output_dir = "./data/cube_2014",
     memsize = 16, multicores = 4)
cube_2019_bayes = sits_smooth(cube_2019_prob, output_dir = "./data/cube_2019",
     memsize = 16, multicores = 4)
cube_2024_bayes = sits_smooth(cube_2024_prob, output_dir = "./data/cube_2024",
     memsize = 16, multicores = 4)

# class serialization
cube_2014_class = sits_label_classification(cube_2014_bayes, output_dir = "./
    data/cube_2014", memsize = 16, multicores = 4)
cube_2019_class = sits_label_classification(cube_2019_bayes, output_dir = "./
    data/cube_2019", memsize = 16, multicores = 4)
cube_2024_class = sits_label_classification(cube_2024_bayes, output_dir = "./
```

```
    data/cube_2024", memsize = 16, multicores = 4)

plot(cube_2014_class)
plot(cube_2019_class)
plot(cube_2024_class)
```

Accuracy assessments

Best practices for splitting training and test datasets are recommended in idetnfiying a proportionate coverage across classes and space such as the stratified random sampling of Cochran's method (**cochran1977sampling?**). This method attempts to divide the image into homogeneous strata before randomly sampling within each stratum.

Alternatively, ad-hoc parameterization is suggested as follows, which also offers useful customization when revisions or class omissions are needed, such as with the GLanCE dataset here.

```
validation_design_2024 = sits_sampling_design(
  cube = cube_2024_class,
  expected_ua = c(
    "Farmbush"  = 0.75,
    "Regrowth"  = 0.70,
    "TreeCover" = 0.75,
    "Urban"     = 0.70,
    "Water"     = 0.70
  ),
  alloc_options    = c(120, 100),
  std_err          = 0.01,
  rare_class_prop = 0.1
  )
validation_design_2024

# split test samples
samples_test_2024 = sits_stratified_sampling(
  cube              = cube_2024_class,
  sampling_design   = validation_design_2024,
  alloc             = "alloc_120",
  multicores        = 4,
  shp_file          = "./data/training_samples/glance_test_2024.shp"
  )

sf::st_write(samples_test_2024,
  "./data/training_samples/glance_test_2024.csv",
  layer_options = "GEOMETRY=AS_XY",
  append = FALSE # TRUE if editing existing sample
  )

# confusion matrix: Olofsson's method (add citation)
acc_2024 = sits_accuracy(cube_2024_class, samples_test_2024, multicores = 4)
acc_2024$error_matrix
#acc_2024
```

Map uncertainty areas[3]

[3]Question to self: Is it possible to map uncertainty of cube_2024_prob & cube_2024_bayes, which would allow comparison between pre- and post-segmentation estimates.

To improve model performance, we estimate classifier's uncertainty and plot pixels' error metrics. We then add new samples to areas of high uncertainty and rerun accuracy assessment to monitor improvements or investigate model weaknesses.

```
cube_2014_uncert = sits_uncertainty(cube_2014_class, output_dir = "./data/cube
    _uncertainty")
cube_2019_uncert = sits_uncertainty(cube_2019_class, output_dir = "./data/cube
    _uncertainty")
cube_2024_uncert = sits_uncertainty(cube_2024_class, output_dir = "./data/cube
    _uncertainty")

hist(cube_2014_uncert)
hist(cube_2019_uncert)
hist(cube_2024_uncert)
plot(cube_2014_uncert)
plot(cube_2019_uncert)
plot(cube_2024_uncert)
```

Add samples to areas of high uncertainty

```
# Find samples with high uncertainty
new_samples    = sits_uncertainty_sampling(
   uncert_cube = cube_2024_uncert,
   n           = 20,
   min_uncert  = 0.5,
   sampling_window = 10
   )
sits_view(new_samples)

new_samples$label <- "Wetland"

# Obtain the time series from the regularized cube
new_samples_ts <- sits_get_data(
   cube = s2_reg_cube_ro,
   samples = new_samples
   )

# Add new class to original samples
samples_round_2 <- dplyr::bind_rows(
   samples_4classes_3bands,
   new_samples_ts
   )

# Train a RF model with the new sample set
rfor_model_v2 <- sits_train(
   samples = samples_round_2,
   ml_method = sits_rfor()
   )

# Classify the small area cube
s2_cube_probs_v2 <- sits_classify(
   data = s2_reg_cube_ro,
   ml_model = rfor_model_v2,
   output_dir = "./cubes/02_class/",
   version = "v2",
```

```
  memsize = 16,
  multicores = 4
  )

# Post-process the probability cube
s2_cube_bayes_v2 <- sits_smooth(
  cube = s2_cube_probs_v2,
  output_dir = "./cubes/04_smooth/",
  version = "v2",
  memsize = 16,
  multicores = 4
  )

# Label the post-processed  probability cube
s2_cube_label_v2 <- sits_label_classification(
  cube = s2_cube_bayes_v2,
  output_dir = "./cubes/05_tuned/",
  version = "v2",
  memsize = 16,
  multicores = 4
  )

# Plot the second version of the classified cube
plot(s2_cube_label_v2)
```

Remap uncertainty & confusion matrix

```
# Calculate the uncertainty cube
s2_cube_uncert_v2 <- sits_uncertainty(
  cube = s2_cube_bayes_v2,
  type = "margin",
  output_dir = "./cubes/03_error/",
  version = "v2",
  memsize = 16,
  multicores = 4
)

plot(s2_cube_uncert_v2)
```


Liberia, Government of. 2019. *Liberia's Forest Reference Emission Level Submission to the UNFCCC*. 1. Forestry Development Authority.

Verra. 2021. *VT0007: Unplanned Deforestation Allocation Tool*. 0.1. Verra. https://verra.org/wp-content/uploads/2024/02/VT0007-Unplanned-Deforestation-Allocation-v1.0.pdf.

———. 2023. *VM0048: Reducing Emissions from Deforestation and Forest Degradation*. 1.0. Verra. https://verra.org/wp-content/uploads/2023/11/VM0048-Reducing-Emissions-from-Deforestation-and-Forest-Degradation-v1.0-1-1.pdf.

———. 2024. *VMD0055: Estimation of Emission Reductions from Avoiding Unplanned Deforestation*. VCS Module 1.1. Verra. https://verra.org/wp-content/uploads/2024/10/VMD0055-Estimation-of-Emission-Reductions-from-Avoiding-Unplanned-Deforestation-v1.1-CLEAN-2024.10.21.24.pdf.

Woodcock, Curtis, Pontus Olofsson, Thomas Loveland, Chris Barber, and Zhe Zhu. n.d. "Global Land Cover Estimation (GLanCE) Product User Guide Version 1.0 August 2022."