Supplementary Materials 1

Code to run the classification in Google Earth Engine

Example of Google Earth Engine script for sea and land classification for the year 2000

```
//MASK CLOUD LANDSAT
var cloudMaskL457 = function(image) {
  var qa = image.select('pixel_qa');
  // If the cloud bit (5) is set and the cloud confidence (7) is high
  // or the cloud shadow bit is set (3), then it's a bad pixel.
  var cloud = qa.bitwiseAnd(1 << 5)
                .and(qa.bitwiseAnd(1 << 7))
                .or(qa.bitwiseAnd(1 << 3));
  // Remove edge pixels that don't occur in all bands
  var mask2 = image.mask().reduce(ee.Reducer.min());
  return image.updateMask(cloud.not()).updateMask(mask2).clip(jawatimur);
};
//LOAD LANDSAT 7 DATA
var lan7 = ee.ImageCollection('LANDSAT/LE07/C01/T1_SR')
                .filterBounds(jawatimur)
                .map(cloudMaskL457);
print(lan7, 'landsat_7_2000');
var land72000 = lan7.filterDate('2000-01-01', '2001-01-01');
var medianl72000 = land72000.median();
var ndwil72000 = medianl72000.normalizedDifference(['B2', 'B4']).rename('ndwil72000');
var ndvil72000 = medianl72000.normalizedDifference(['B4','B3']).rename('ndvil72000');
var awei72000 = medianl72000.expression('4 * (GREEN - SWIR1) - ( 0.25 * NIR ) + (2.75 * SWIR2)', {
    'GREEN': medianl72000.select('B2'),
    'NIR': medianl72000.select('B4'),
    'SWIR1': medianl72000.select('B5'),
    'SWIR2': medianl72000.select('B7'),
}).rename('awei72000');
var compositel72000 = ee.Image.cat(medianl72000,ndwil72000,ndvil72000,awei72000);
//TRAINING SAMPLE AND INPUT
```

```
var finall72000 = ee.Image.cat(compositel72000);
var bandsl72000 = ['B1', 'B2','B3', 'B4', 'B5', 'B7','ndwil72000' ,'ndvil72000','awei72000'];
var trainingl72000 = finall72000.select(bandsl72000).sampleRegions({
   collection : newfc2000,
    properties : ['landcover'],
 scale : 30,
 tileScale : 8,
 geometries : true
});
//TRAIN CLASSIFICATION
var classifierl72000 = ee.Classifier.gmoMaxEnt().train({
 features:trainingl72000,
  classProperty: 'landcover',
  inputProperties: bandsl72000,
});
var classifierl72000rf = ee.Classifier.smileRandomForest(50).train({
 features:trainingl72000,
  classProperty: 'landcover',
  inputProperties: bandsl72000,
});
var classifierl72000rf2 = ee.Classifier.smileRandomForest(80).train({
 features:trainingl72000,
  classProperty: 'landcover',
  inputProperties: bandsl72000,
});
//RUN CLASSIFICATION
var classifiedl72000 = finall72000.select(bandsl72000).classify(classifierl72000);
var classifiedl72000rf = finall72000.select(bandsl72000).classify(classifierl72000rf);
var classifiedl72000rf2 = finall72000.select(bandsl72000).classify(classifierl72000rf2);
```

GOOGLE EARTH ENGINE SCRIPT FOR SEA AND LAND CLASSIFICIATION FOR 2005 CLASSIFICATION USING LANDSAT 7 ETM

```
//MASK CLOUD LANDSAT
var cloudMaskL457 = function(image) {
  var qa = image.select('pixel_qa');
  // If the cloud bit (5) is set and the cloud confidence (7) is high
  // or the cloud shadow bit is set (3), then it's a bad pixel.
  var cloud = qa.bitwiseAnd(1 << 5)
                .and(qa.bitwiseAnd(1 << 7))
                .or(qa.bitwiseAnd(1 << 3));
  // Remove edge pixels that don't occur in all bands
  var mask2 = image.mask().reduce(ee.Reducer.min());
  return image.updateMask(cloud.not()).updateMask(mask2).clip(jawatimur);
};
//LOAD LANDSAT 7 DATA
var lan7 = ee.ImageCollection('LANDSAT/LE07/C01/T1_SR')
                .filterBounds(jawatimur)
                .map(cloudMaskL457);
var land72005 = lan7.filterDate('2005-01-01', '2006-01-01');
var medianl72005 = land72005.median();
var ndwil72005 = medianl72005.normalizedDifference(['B2','B4']).rename('ndwil72005');
var ndvil72005 = medianl72005.normalizedDifference(['B4','B3']).rename('ndvil72005');
var awel72005 = medianl72005.expression('4 * (GREEN - SWIR1) - ( 0.25 * NIR ) + (2.75 * SWIR2)', {
    'GREEN': medianl72005.select('B2'),
    'NIR': medianl72005.select('B4'),
    'SWIR1': medianl72005.select('B5'),
    'SWIR2': medianl72005.select('B7'),
}).rename('awel72005');
var compositel72005 = ee.Image.cat(medianl72005,ndwil72005,ndvil72005,awel72005);
//TRAINING SAMPLE AND INPUT
var finall72005 = ee.Image.cat(compositel72005);
```

```
var bandsl72005 = ['B1', 'B2','B3', 'B4', 'B5','B7', 'ndwil72005', 'ndvil72005','awel72005'];
var trainingl72005 = finall72005.select(bandsl72005).sampleRegions({
  collection : newfc2005,
   properties : ['landcover'],
 scale : 30,
 tileScale : 8,
 geometries : true,
});
//TRAIN CLASSIFICATION
//var classifierl72005 = ee.Classifier.gmoMaxEnt().train({
//features:trainingl72005,
//  classProperty: 'landcover',
//  inputProperties: bandsl72005,
//});
var classifierl72005rf1 = ee.Classifier.smileRandomForest(50).train({
features:trainingl72005,
  classProperty: 'landcover',
  inputProperties: bandsl72005,
});
var classifierl72005rf2 = ee.Classifier.smileRandomForest(80).train({
  features:trainingl72005,
  classProperty: 'landcover',
  inputProperties: bandsl72005,
});
//RUN CLASSIFICATION
//var classifiedl72005 = finall72005.select(bandsl72005).classify(classifierl72005);
var classifiedl72005rf = finall72005.select(bandsl72005).classify(classifierl72005rf1);
var classifiedl72005rf2 = finall72005.select(bandsl72005).classify(classifierl72005rf2);
```

GOOGLE EARTH ENGINE SCRIPT FOR SEA AND LAND CLASSIFICIATION FOR 2010 USING LANDSAT 7 AND ALOS PALSAR

```
//CLOUD MASKING
var cloudMaskL457 = function(image) {
  var qa = image.select('pixel_qa');
  // If the cloud bit (5) is set and the cloud confidence (7) is high
  // or the cloud shadow bit is set (3), then it's a bad pixel.
  var cloud = qa.bitwiseAnd(1 << 5)
                 .and(qa.bitwiseAnd(1 << 7))
                 .or(qa.bitwiseAnd(1 << 3));
  // Remove edge pixels that don't occur in all bands
  var mask2 = image.mask().reduce(ee.Reducer.min());
  return image.updateMask(cloud.not()).updateMask(mask2).clip(jawatimur);
};
//LOAD ALOS
var dataset = ee.ImageCollection('JAXA/ALOS/PALSAR/YEARLY/SAR').filterBounds(jawatimur)
                 .filter(ee.Filter.date('2010-01-01', '2011-01-01'));
var sarHh = dataset.select('HH');
var HHmed = sarHh.median();
var sarHv = dataset.select('HV');
//var sarHhVis = {
//  min: 0.0,
//  max: 10000.0,
//};
var HVmed = sarHv.median();
//LOAD LANDSAT 7
var check = ee.ImageCollection('LANDSAT/LE07/C01/T1_SR')
    .filterBounds(jawatimur)
    .filterDate('2010-01-01', '2011-01-01')
    .map(cloudMaskL457)
    .filterBounds(jawatimur);
var medianl72010 = check.median();
```

```
var ndvil72010 = medianl72010.normalizedDifference(['B4','B3']).rename('ndvil72010');
var ndwil72010 = medianl72010.normalizedDifference(['B2','B4']).rename('ndwil72010');
var awel72010 = medianl72010.expression('4 * (GREEN - SWIR1) - ( 0.25 * NIR ) + (2.75 * SWIR2)', {
      'GREEN': medianl72010.select('B2'),
      'NIR': medianl72010.select('B4'),
      'SWIR1': medianl72010.select('B5'),
      'SWIR2': medianl72010.select('B7'),
}).rename('awel72010');
//EQUALIZE PROJECTION AND SPATIAL RESOLUTION LANDSAT 7 AS REFERENCE
var hhpro = HHmed
    // Force the next reprojection to aggregate instead of resampling.
    .reproject(l7Projection, null, 30)
    .reduceResolution({
      reducer: ee.Reducer.mean(),
      maxPixels: 1024
    });
var hvpro = HVmed
    // Force the next reprojection to aggregate instead of resampling.
    .reproject(l7Projection, null, 30)
    .reduceResolution({
      reducer: ee.Reducer.mean(),
      maxPixels: 1024
    });
var compositeland72010 = ee.Image.cat(medianl72010,ndvil72010,ndwil72010,awel72010,hhpro,hvpro);
var finall72010 = ee.Image.cat(compositeland72010);
var bandsl72010 = ['B1', 'B2','B3', 'B4', 'B5', 'B7', 'ndvil72010', 'ndwil72010','awel72010', 'HH', 'HV'];
var trainingl72010 = finall72010.select(bandsl72010).sampleRegions({
   collection : newfc2010,
    properties : ['landcover'],
  scale : 30, });
var classifierl72010 = ee.Classifier.gmoMaxEnt().train({
features:trainingl72010,
  classProperty: 'landcover',
```

```
  inputProperties: bandsl72010,
});
var classifierl72010rf = ee.Classifier.smileRandomForest(50).train({
features:trainingl72010,
  classProperty: 'landcover',
  inputProperties: bandsl72010,
});
var classifierl72010rf2 = ee.Classifier.smileRandomForest(80).train({
features:trainingl72010,
  classProperty: 'landcover',
  inputProperties: bandsl72010,
});
var classifiedl72010 = finall72010.select(bandsl72010).classify(classifierl72010);
var classifiedl72010rf = finall72010.select(bandsl72010).classify(classifierl72010rf);
var classifiedl72010rf2 = finall72010.select(bandsl72010).classify(classifierl72010rf2);
```

GOOGLE EARTH ENGINE SCRIPT FOR SEA AND LAND CLASSIFICIATION FOR 2015 USING LANDSAT 8 AND SENTINEL 1

```
//LOAD SENTINEL 1
var collectionVV1 = ee.ImageCollection('COPERNICUS/S1_GRD')
                .filter(ee.Filter.eq('instrumentMode', 'IW'))
                .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VV'))
                .filter(ee.Filter.eq('orbitProperties_pass', 'DESCENDING'))
                .filterMetadata('resolution_meters', 'equals' , 10)
                .filterBounds(jawatimur)
                .select('VV')
                .map(function(image){return image.clip(jawatimur)});
var collectionVH = ee.ImageCollection('COPERNICUS/S1_GRD')
                .filter(ee.Filter.eq('instrumentMode', 'IW'))
                .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VH'))
                .filter(ee.Filter.eq('orbitProperties_pass', 'DESCENDING'))
                .filterMetadata('resolution_meters', 'equals' , 10)
                .filterBounds(jawatimur)
                .select('VH')
                .map(function(image){return image.clip(jawatimur)});
//APPLY SPECKLE FILTER VV (focal median)
var filterSpeckles = function(img) {
  var vv = img.select('VV') //select the VV polarization band
  var vv_smoothed = vv.focal_median(50,'circle','meters').rename('VV_Filtered') //Apply a focal median filter
  return img.addBands(vv_smoothed) // Add filtered VV band to original image
};
var filterSpeckles2 = function(img) {
  var vh = img.select('VH') //select polarization band
  var vh_smoothed = vh.focal_median(50,'circle','meters').rename('VV_Filtered') //Apply a focal median filter
  return img.addBands(vh_smoothed) // Add filtered Vh band to original image
};
// filter date (2015) to speckle-noise-filtered data and add median value represents entire year image
var S1vv = collectionVV1.map(filterSpeckles);
```

```
var vv2015 = S1vv.filterDate('2015-01-01', '2016-01-01' );
var compvv2015 = vv2015.median();
var S1vh = collectionVH.map(filterSpeckles2);
var vh2015 = S1vh.filterDate('2015-01-01', '2016-01-01' );
var compvh2015 = vh2015.median();
//CLOUD MASK LANDSAT 8
var cloudShadowBitMask = 1 << 3;
var cloudsBitMask = 1 << 5;
// Get the pixel QA band.
var qa = image.select('pixel_qa');
// Both flags should be set to zero, indicating clear conditions.
var mask = qa.bitwiseAnd(cloudShadowBitMask).eq(0)
.and(qa.bitwiseAnd(cloudsBitMask).eq(0));
// Return the masked image, scaled to reflectance, without the QA bands.
return image.updateMask(mask).clip(jawatimur).divide(10000)
.select("B[0-9]*")
.copyProperties(image, ["system:time_start"]);
}
//LOAD LANDSAT 8
var landsat = ee.ImageCollection('LANDSAT/LC08/C01/T1_SR')
.filterBounds(jawatimur)
.map(maskL8sr);
var landsat2015 = landsat.filterDate('2015-01-01', '2016-01-01');
        print(landsat2015);
var median2015 = landsat2015.median();
var ndvi2015 = median2015.normalizedDifference(['B5', 'B4']).rename('NDVI');
var ndwi2015 = median2015.normalizedDifference(['B3', 'B5']).rename('NDWI');
var awel82015 = median2015.expression('4 * (GREEN - SWIR1) - ( 0.25 * NIR ) + (2.75 * SWIR2)', {
    'GREEN': median2015.select('B3'),
    'NIR': median2015.select('B5'),
    'SWIR1': median2015.select('B6'),
    'SWIR2': median2015.select('B7'),
}).rename('awel82015');
```

```
var compositeL82015 = ee.Image.cat(median2015,ndvi2015,ndwi2015,awel82015);
//EQUALIZE PROJECTION AND SPATIAL RESOLUTION USING LANDSAT 8 AS REFERENCE
var l8Projection =  median2015.projection();
var VHpro = compvh2015
    .reproject(l8Projection, null, 30)
   .reduceResolution({
      reducer: ee.Reducer.mean(),
      maxPixels: 1024
    });
var VVpro = compvv2015
    .reproject(l8Projection, null, 30)
    .reduceResolution({
      reducer: ee.Reducer.mean(),
      maxPixels: 1024
    });
////TRAINING SAMPLE & INPUT
var newfc2015 = darat.merge(air).merge(suspen);
var opt_sar2015 = ee.Image.cat(compositeL82015, VVpro, VHpro);
var band_opt_sar2015 = ['B2', 'B3', 'B4', 'B5','B6','B7', 'NDVI','NDWI','awel82015', 'VH', 'VV'];
var training2015 = opt_sar2015.select(band_opt_sar2015).sampleRegions({
    collection : newfc2015,
   properties : ['landcover'],
    scale : 30,
});
//TRAIN CLASSIFICATION
var classifierl8sar_2015 = ee.Classifier.gmoMaxEnt().train({
    features:training2015,
  classProperty: 'landcover',
  inputProperties: band_opt_sar2015
});
var classifierl8sar_2015rf = ee.Classifier.smileRandomForest(50).train({
    features:training2015,
  classProperty: 'landcover',
```

```
  inputProperties: band_opt_sar2015
});
var classifierl8sar_2015rf2 = ee.Classifier.smileRandomForest(80).train({
    features:training2015,
  classProperty: 'landcover',
  inputProperties: band_opt_sar2015
});
//RUN CLASSIFICATION 2015
var classified_2015 = opt_sar2015.select(band_opt_sar2015).classify(classifierl8sar_2015);
var classified_2015rf = opt_sar2015.select(band_opt_sar2015).classify(classifierl8sar_2015rf);
var classified_2015rf2 = opt_sar2015.select(band_opt_sar2015).classify(classifierl8sar_2015rf2);var newfc2015 =
darat.merge(air).merge(suspen);
```

GOOGLE EARTH ENGINE SCRIPT FOR SEA AND LAND CLASSIFICIATION FOR 2019 USING LANDSAT 8 AND SENTINEL 1

```
//LOAD SENTINEL 1
var collectionVV1 = ee.ImageCollection('COPERNICUS/S1_GRD')
                .filter(ee.Filter.eq('instrumentMode', 'IW'))
                .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VV'))
                .filter(ee.Filter.eq('orbitProperties_pass', 'DESCENDING'))
                .filterMetadata('resolution_meters', 'equals' , 10)
                .filterBounds(jawatimur)
                .select('VV')
                .map(function(image){return image.clip(jawatimur)});
var collectionVH = ee.ImageCollection('COPERNICUS/S1_GRD')
                .filter(ee.Filter.eq('instrumentMode', 'IW'))
                .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VH'))
                .filter(ee.Filter.eq('orbitProperties_pass', 'DESCENDING'))
                .filterMetadata('resolution_meters', 'equals' , 10)
                .filterBounds(jawatimur)
                .select('VH')
                .map(function(image){return image.clip(jawatimur)});
//APPLY SPECKLE FILTER VV (focal median)
var filterSpeckles = function(img) {
  var vv = img.select('VV') //select the VV polarization band
  var vv_smoothed = vv.focal_median(50,'circle','meters').rename('VV_Filtered') //Apply a focal median filter
  return img.addBands(vv_smoothed) // Add filtered VV band to original image
};
var filterSpeckles2 = function(img) {
  var vh = img.select('VH') //select polarization band
  var vh_smoothed = vh.focal_median(50,'circle','meters').rename('VV_Filtered') //Apply a focal median filter
  return img.addBands(vh_smoothed) // Add filtered Vh band to original image
};
// filter date (2015) to speckle-noise-filtered data and add median value represents entire year image
var S1vv = collectionVV1.map(filterSpeckles);
```

```
var vv2019 = S1vv.filterDate('2019-01-01', '2020-01-01' );
var compvv2019 = vv2019.median();
var S1vh = collectionVH.map(filterSpeckles2);
var vh2019 = S1vh.filterDate('2019-01-01', '2020-01-01' );
var compvh2019 = vh2019.median();
//CLOUD MASKING LANDSAT 8
// Function to cloud mask from the pixel QA band of Landsat 8 SR data.
function maskL8sr(image) {
// Bits 3 and 5 are cloud shadows and clouds, respectively.
var cloudShadowBitMask = 1 << 3;
var cloudsBitMask = 1 << 5;
// Get the pixel QA band.
var qa = image.select('pixel_qa');
// Both flags should be set to zero, indicating clear conditions.
var mask = qa.bitwiseAnd(cloudShadowBitMask).eq(0)
.and(qa.bitwiseAnd(cloudsBitMask).eq(0));
// Return the masked image, scaled to reflectance, without the QA bands.
return image.updateMask(mask).clip(jawatimur).divide(10000)
.select("B[0-9]*")
.copyProperties(image, ["system:time_start"]);
}
// LOAD LANDSAT 8
var landsat = ee.ImageCollection('LANDSAT/LC08/C01/T1_SR')
.filterBounds(jawatimur)
.map(maskL8sr);
var landsat2019 = landsat.filterDate('2019-01-01', '2020-01-01');
        print(landsat2019);
var median2019 = landsat2019.median();
var ndvi2019 = median2019.normalizedDifference(['B5', 'B4']).rename('NDVI');
var ndwi2019 = median2019.normalizedDifference(['B3', 'B5']).rename('NDWI');
var awel82019 = median2019.expression('4 * (GREEN - SWIR1) - ( 0.25 * NIR ) + (2.75 * SWIR2)', {
    'GREEN': median2019.select('B3'),
    'NIR': median2019.select('B5'),
```

```
      'SWIR1': median2019.select('B6'),
      'SWIR2': median2019.select('B7'),
}).rename('awel82019');
var compositeL82019 = ee.Image.cat(median2019,ndvi2019,ndwi2019,awel82019);
//EQUALIZE PROJECTION AND SPATIAL RESOLUTION USING LANDSAT 8 AS REFERENCE
var l8Projection =  median2019.projection();
var VHpro2 = compvh2019
      .reproject(l8Projection, null, 30)
   .reduceResolution({
     reducer: ee.Reducer.mean(),
      maxPixels: 1024
   });
var VVpro2 = compvv2019
    .reproject(l8Projection, null, 30)
    .reduceResolution({
     reducer: ee.Reducer.mean(),
    maxPixels: 1024
    });
////TRAINING SAMPLE & INPUT
var opt_sar2019 = ee.Image.cat(compositeL82019, VHpro2, VVpro2);
var band_opt_sar2019 = [ 'B2', 'B3', 'B4', 'B5','B6','B7', 'NDVI','NDWI','awel82019', 'VH', 'VV'];
var training_opt_sar2019 = opt_sar2019.select(band_opt_sar2019).sampleRegions({
    collection : newfc2019,
    properties : ['landcover'],
    scale : 30,
    tileScale : 16,
});
//TRAIN CLASSIFICATION
var classifier2019 = ee.Classifier.gmoMaxEnt().train({
  features:training_opt_sar2019,
  classProperty: 'landcover',
  inputProperties: band_opt_sar2019
});
```

```
var classifier2019rf = ee.Classifier.smileRandomForest(50).train({
  features:training_opt_sar2019,
  classProperty: 'landcover',
  inputProperties: band_opt_sar2019
});
var classifier2019rf2 = ee.Classifier.smileRandomForest(80).train({
  features:training_opt_sar2019,
  classProperty: 'landcover',
  inputProperties: band_opt_sar2019
});
//RUN CLASSIFICATION
var classified2019 = opt_sar2019.select(band_opt_sar2019).classify(classifier2019);
var classified2019rf = opt_sar2019.select(band_opt_sar2019).classify(classifier2019rf);
var classified2019rf2 = opt_sar2019.select(band_opt_sar2019).classify(classifier2019rf2);
```