Coastal Ecosystem Mapping Validation Framework

Overview

This validation framework is a core component of the Digital Earth Australia (DEA) Coastal Ecosystems Mapping product. The framework implements resampling-based accuracy assessment methods to quantify both map accuracy and uncertainty, addressing the critical need for robust validation in remote sensing classification.

Key Features:

- Monte Carlo or spatially stratified validation with user-defined iteration counts
- Support for the combined, pre-filtered coastal ecosystem parquet input
- Hierarchical model validation (ecosystem → seagrass refinement)
- Probability threshold-based accuracy assessment, including coverage reporting
- Year-specific accuracy metrics (2021 vs 2022 imagery) without re-fitting models
- Optional expert-only runs via CLI while defaulting to all available samples
- Summary statistics reported as medians with 95% intervals (F1 as mean ± std)

Background

Coastal Ecosystem Mapping Context

The Digital Earth Australia (DEA) Coastal Ecosystems Mapping (CEM) product produces high-resolution maps of Australia's coastal ecosystems using Sentinel-2 satellite imagery. The mapping employs a hierarchical classification approach:

1. **Binary Coastal Model**: Separates coastal from non-coastal areas
2. **Ecosystem Model**: Classifies coastal areas into ecosystem types:
    - Tidal flat/Mudflat (class 2)
    - Mangrove (class 3)
    - Saltmarsh (class 4)
    - Salt flat (class 6)
3. **Seagrass Refinement Model**: Identifies seagrass (class 5) within tidal flats

Training Data

The current workflow consumes a single pre-filtered dataset that already mirrors the preprocessing steps in the modelling notebook:

- **Source File**: `/gdata1/projects/coastal/cem/training_data/dea_cem_covariates_v1_2_3_combined.parquet`
- **Status**: Ready for modelling/validation (excluded classes removed, QA thresholds applied, negative connectivity fixed)
- **Override**: Alternate parquet/CSV inputs can be supplied via `--data-files`
- **Total Points**: ~688,000 observations across 7 model regions
- **Data Types**:
    - Expert-labeled points (auto=0): Original field-verified observations
    - Semi-automated points (auto=1): Algorithmically expanded training data
- **Covariates**: 131 variables including:
    - Sentinel-2 multispectral percentiles (10th, 20th, 40th, 60th, 80th)
    - Spectral indices (NDVI, MNDWI, AWEI, etc.)
    - Tasseled Cap transformations
    - DEA Intertidal low-tide composites
    - Coastal connectivity metrics

Class Mapping Schema

Original class_l2	Ecosystem Class	Description	Notes
0, 55, 12, 56	0/10	Terrestrial/Non-coastal	Filtered from ecosystem model
1	1	Water	Filtered from ecosystem model
2, 41	2	Tidal flat/Mudflat	May contain seagrass
3	3	Mangrove	-
5, 52	4	Saltmarsh	Includes algal mats
9	5	Seagrass (intertidal)	Identified within tidal flats
11	6	Salt flat	-

Note: Class 9 (intertidal seagrass) is initially mapped as class 2 (tidal flat) in the ecosystem model, then refined to class 5 by the seagrass model based on a probability threshold (default 70%).

The Validation Problem

Traditional validation approaches using a single train/test split suffer from critical limitations:

1. **High Variance**: A single split can produce accuracy estimates ranging from 40-80% due to random partitioning alone (Lyons et al., 2018)
2. **No Uncertainty Quantification**: Single-split approaches provide no confidence intervals
3. **Spatial Autocorrelation Bias**: Nearby training points can artificially inflate accuracy when randomly split
4. **Unrepresentative Samples**: Any single partition may not represent the true data distribution

The Solution: Resampling-Based Validation

This framework implements Monte Carlo resampling with spatial stratification following best practices from Lyons et al. (2018, *Remote Sensing of Environment*):

Key Principles:

- Multiple train/test splits (500 iterations) create sampling distributions of accuracy metrics
- Spatial stratification using 32km tiles ensures geographic independence between train/test sets
- Median estimates with percentile confidence intervals provide robust accuracy assessment
- Comparison across datasets (all data vs expert-only) tests robustness to training data quality

---

Validation Methodology

Resampling Framework

Spatial Stratification Design

The validation employs **tile-based spatial stratification** to address spatial autocorrelation:

1. **Grid System**: Training points are assigned to 32km × 32km tiles from the GA summary grid
2. **Tile-Level Splitting**: Each iteration randomly assigns 80% of tiles to training, 20% to testing
3. **Geographic Independence**: Training and test points come from entirely different tiles
4. **Spatial Separation**: Reduces optimistic bias from testing on locations near training sites

Tile Assignment Process:

```
Load tiles → Spatial join points to tiles → Validate sufficient tiles (≥2) → Random 80/20 split by tiles
```

Fallback: Regions with insufficient tiles automatically fall back to random sampling.

Iteration Process

For each of 500 iterations:

1. **Data Split**: Randomly assign tiles to train (80%) / test (20%) sets
2. **Ecosystem Model**:
   - Filter data to coastal classes (2, 3, 4, 6)
   - Recode class_l2 to ecosystem classes using `class_to_ecosystem` dictionary
   - Train Random Forest on training set
   - Predict on test set
   - Calculate accuracy metrics
3. **Seagrass Refinement**:
   - Filter to tidal flat predictions (class 2)
   - Recode using `class_to_seagrass` dictionary
   - Train Random Forest on tidal flat training samples
   - Predict seagrass probability
   - Apply threshold (default 70%) to identify seagrass
   - Update final classifications (2 → 5 where prob ≥ threshold)
   - Calculate final accuracy metrics
4. **Store Results**: Accumulate accuracy metrics across iterations

Probability-Based Validation

A critical feature is **probability threshold filtering**:

- **Default Mode**: Only evaluates predictions with ≥50% probability
- **Rationale**: Mimics operational mapping where low-confidence predictions are masked
- **Impact**: Provides realistic accuracy estimates for the actual map product
- **Coverage Metric**: Reports % of test samples evaluated (typically >98%)

Example: A mangrove prediction with 45% probability would be excluded from accuracy assessment, reflecting that it would also be masked in the final map.

Year-Specific Accuracy Metrics

Map production is undertaken separately for Sentinel-2 mosaics from 2021 and 2022.
To mirror this, the validation script now:

1. Fits ecosystem and seagrass models across the full dataset (no year-based re-training)

2. Splits accuracy calculations by the `year` column inside every Monte Carlo iteration
3. Aggregates per-year distributions alongside region-wide metrics

The resulting markdown report contains dedicated tables for the whole dataset plus each imagery year, helping downstream users attribute performance differences to particular acquisition periods.

Random Forest Configuration

Model Hyperparameters:

- Trees: 500 (configurable via `--n-estimators`)
- Max depth: 10
- Min samples split: 2
- Min samples leaf: 1
- Parallel processing: All available cores (`n_jobs=-1`)

Accuracy Metrics

Comprehensive per-class and overall metrics are calculated:

- **Overall Accuracy**: Proportion of correct predictions
- **User's Accuracy** (Precision): P(true class | predicted class)
- **Producer's Accuracy** (Recall): P(predicted correctly | true class)
- **F1 Score**: Harmonic mean of precision and recall
- **Confusion Matrix**: Full error matrix for each iteration

Summary statistics are reported as **median accuracy with 95% intervals** derived from the iteration distributions.
F1 remains **mean ± standard deviation**, matching how harmonic means are typically summarised.

Validation Scenarios

Validation subsets are controlled through `--data-subsets` :

1. **all** (default): Uses the complete training dataset (auto=0 and auto=1)
2. **expert**: Restricts to expert-labelled points (auto=0)

You can specify one or both values to compare training data quality within a single run.

Script Functionality & Usage

Main Script: `validation.py`

Located in `/gdata1/projects/coastal/cem/coastalecosystems/validation.py`

Command-Line Interface

```
python validation.py \
    --sampling-method spatial \
    --iterations 500 \
    --n-estimators 500 \
    --seagrass-threshold 70 \
    --regions 1 2 3 4 5 6 7 \
    --output results/validation_results.md \
    --validation-mode probability \
    --probability-threshold 50.0
```

Add `--data-subsets all expert` to run both subsets in one pass; omit the flag to process the full dataset only.

Parameters

Parameter	Type	Default	Description
`--sampling-method`	str	`random`	Sampling strategy: `spatial` or `random`
`--iterations`	int	10	Number of Monte Carlo iterations (increase to 500+ for production)
`--n-estimators`	int	10	Number of trees in Random Forest (raise for production runs)

Parameter	Type	Default	Description
`--seagrass-threshold`	int	70	Probability threshold (%) for seagrass identification
`--regions`	list	all	Model regions to validate (1-7)
`--output`	str	`docs/coastal_ecosystem_validation_results.md`	Output markdown file
`--validation-mode`	str	`probability`	Mode: `probability` or `categorical`
`--probability-threshold`	float	50.0	Minimum prediction probability (%) for inclusion
`--data-subsets`	list	`all`	Specify `all`, `expert`, or both to control which data subsets are processed

Sampling Methods

Spatial Stratification (Recommended for production)

```
python validation.py --sampling-method spatial --iterations 500
```

- Ensures geographic independence between train/test sets
- Reduces optimistic bias from spatial autocorrelation
- 80/20 train/test split by tiles
- More conservative, realistic accuracy estimates

Random Sampling (For comparison/testing)

```
python validation.py --sampling-method random --iterations 500
```

- Traditional random split without spatial consideration
- 67/33 train/test split
- Faster computation
- May overestimate accuracy for spatially clustered data

Validation Modes

Probability Mode (Default, recommended)

```
python validation.py --validation-mode probability --probability-threshold 50.0
```

- Only includes predictions meeting probability threshold
- Reflects operational map accuracy (low-confidence areas are masked)
- Reports coverage metric (% of samples evaluated)

Categorical Mode (For comparison)

```
python validation.py --validation-mode categorical
```

- Uses sklearn default categorical predictions
- No probability filtering
- Traditional accuracy assessment

Example Workflows

Quick Test Run (Single Region)

```
python validation.py \
    --sampling-method spatial \
    --iterations 50 \
    --regions 7 \
    --output test_results.md
```

Production Run (All Regions)

```
python validation.py \
    --sampling-method spatial \
    --iterations 500 \
    --n-estimators 500 \
    --regions 1 2 3 4 5 6 7 \
    --output coastal_ecosystem_validation_results.md
```

Expert-Only Comparison

```
# Run both subsets in a single invocation
python validation.py \
    --data-subsets all expert \
    --output results/all_vs_expert.md

# Expert-only (auto=0) only
python validation.py \
    --data-subsets expert \
    --output results/expert_only.md
```

Key Functions

Data Loading & Preparation

`load_and_prepare_data()`

- Loads one or more parquet/CSV files (combined parquet by default)
- Aligns column names where needed (`clear_count` → `qa_count_clear`)
- Assumes upstream filtering has already prepared the dataset
- Verifies mandatory columns (`class_l2`, `model_region`, `auto`, `year`, `lat`, `lon`, covariates)

`load_tiles()`

- Loads 32km grid geometries from `/gdata1/projects/coastal/cem/grids/ga_summary_grid_c3_32km_cem_tps_only.geojson`
- Returns GeoDataFrame of tile polygons

`assign_points_to_tiles()`

- Performs spatial join of training points to tiles
- Uses lat/lon coordinates to create point geometries
- Returns DataFrame with tile assignments

Class Recoding

`recode_ecosystem_classes(data)`

- Applies `class_to_ecosystem` dictionary
- Filters to coastal classes (2, 3, 4, 6)
- Returns ecosystem-recoded DataFrame

Seagrass recoding is handled within the same function, which attaches an `is_seagrass` binary column used when training the refinement model.

Sampling

`split_tiles_train_test(tiles_with_points, train_ratio=0.8)`

- Randomly assigns tiles to train/test sets
- Returns DataFrames split by tile assignment
- Ensures specified train/test ratio

Model Training

`run_single_iteration(data, iteration, sampling_method, ...)`

- Core iteration function
- Handles train/test split (spatial or random)
- Trains both ecosystem and seagrass models
- Calculates accuracy metrics
- Returns dictionary of results

Validation Execution

`process_region(region, sampling_method, iterations, ...)`

- Manages validation for a single region
- Parallel execution of iterations using multiprocessing

- Aggregates results across iterations
- Returns summary statistics

Output Generation

`format_results_markdown(all_results, ...)`

- Builds the markdown report summarising each region/subset
- Presents medians with 95% intervals plus F1 mean ± std
- Adds dedicated sections for each imagery year when available
- Documents sampling/validation settings used for the run

Computational Considerations

Multiprocessing Strategy:

- Random Forest models use all CPU cores (`n_jobs=-1`)
- Iterations run in parallel via `multiprocessing.Pool`
- Balance set to optimize resource usage without over-subscription

Performance:

- ~500 iterations across 7 regions: 2-4 hours on sandbox environment
- Memory usage scales with training data size per region
- Spatial stratification adds ~10% overhead vs random sampling

Data I/O:

- Training data loaded once per region
- Tiles loaded once globally
- Results accumulated in memory, written at completion

Key Results Summary

Full validation results are written to coastal_ecosystem_validation_results.md.
Each run documents:

- Sampling configuration, iteration count, and probability threshold
- Region-by-region summaries for every requested subset (`all` , `expert` , etc.)
- Median accuracies with 95% intervals for ecosystem and final models
- Evaluation coverage statistics (median evaluated samples and coverage %)
- Per-class metrics (user/producer medians with intervals, F1 mean ± std)
- Year-specific tables (e.g. 2021, 2022) that mirror the same layout

Reading the Report

Within each region/subset block you will see tables such as:

Stage	Accuracy
Ecosystem Level	0.95 [0.92, 0.98]
Final (with Seagrass)	0.94 [0.91, 0.97]

Interpretation:

- 0.94 represents the median of all Monte Carlo iterations
- [0.91, 0.97] is the 95% interval (2.5th–97.5th percentiles)

F1 scores continue to be displayed as `mean ± std` because they are derived from the harmonic mean of user's/producer's accuracies.

Typical Production Configuration

- **Sampling**: Spatial stratification (80/20 tile split)
- **Iterations**: 500+
- **Random Forest**: ≥500 trees
- **Validation Mode**: Probability with ≥50% inclusion threshold
- **Seagrass Threshold**: 70%

Although defaults are lightweight for development, the markdown report records the actual options supplied so results can be traced to specific parameter sets.

Implications for Mapping Products

1. Map Accuracy Confidence

Positive Findings:

- Overall accuracy >90% for most regions supports operational use
- Spatial stratification provides conservative, realistic estimates
- High coverage (>98%) means probability filtering doesn't exclude substantial data

Recommendation: Report accuracy with confidence intervals in usage metadata, e.g.:

```
Region 6 Overall Accuracy: 94.5% (95% CI: 91.6-97.4%)
```

2. Class-Specific Reliability

High-Confidence Classes (suitable for decision-making):

- Mangrove (F1: 88-99%)
- Water and Land boundaries (F1: 89-99%)
- Salt flat in Regions 1, 2, 3, 4 (F1: 90-98%)

Moderate-Confidence Classes (use with caution):

- Saltmarsh (F1: 66-89%, particularly low producer's accuracy)
- Tidal flat (F1: 70-93%)
- Seagrass in Regions 2-7 (F1: 74-93%)

Low-Confidence Classes (require future investigation):

- Seagrass in Region 1 (F1: 0.5%)
- Salt flat in Region 5 (F1: 15%, very low producer's accuracy)

Recommendation: Include class-specific accuracy metadata in derived map product usage. Consider flagging low-confidence classes or providing confidence layers.

3. Training Data Quality

Key Finding: Expert-only data shows minimal accuracy degradation (≤0.4%) compared to all data (this is an option flag in `validation.py`)

Implication: Semi-automated training point expansion (auto=1 points) does not significantly affect model performance.

Recommendation: Continue using combined training data approach. The expanded dataset provides valuable spatial coverage without compromising accuracy.

4. Probability Threshold Strategy

Current Approach: 50% minimum prediction probability for inclusion in accuracy assessment.

Findings:

- High coverage (>98%) indicates most predictions are confident
- Mimics operational mapping where low-confidence areas are masked
- Provides realistic accuracy for the actual map product

Recommendations:

- **Maintain 50% threshold** for operational products
- **Consider class-specific thresholds** for problematic classes:
    - Saltmarsh: Increase to 60-70% to reduce commission errors
    - Seagrass: Regional thresholds (higher in challenging regions)
- **Probability layers** are provided for all classes, as well as Salt Flat, which is not included in categorical product

5. Regional Model Strategy

Observation: Accuracy varies across regions (87.1-97.1%).

Implications:

- Regional models are appropriate given environmental variability

Recommendations:

- Continue region-specific usage and reporting
- Develop region-specific seagrass probability thresholds for specific use cases

6. Spatial Uncertainty

Finding: Tile-based stratification reduces but doesn't eliminate spatial variance.

Implication: Accuracy varies spatially within regions. Areas far from training tiles may have lower accuracy.

7. Seagrass Mapping

Critical Issue: Seagrass detection highly variable (0.5-93% F1).

Analysis:

- Region 1: Near-zero accuracy suggests fundamental data deficiency

- Region 7: Excellent accuracy (91-93% F1) demonstrates feasibility
- Regions 2-6: Moderate performance (66-93% F1)

Useage:

1. Region 1 has insufficient seagrass training samples
2. Spectral separability varies by water clarity and seagrass density
3. 70% probability threshold may be inappropriate for some regions

Recommendations:

- **Region-specific seagrass thresholds**: Lower for Region 1 if any signal exists

8. Saltmarsh Producer's Accuracy

Finding: Consistently lower producer's accuracy (55-90%) vs user's accuracy (79-91%).

Interpretation: Saltmarsh is frequently misclassified as other classes (omission errors), but predictions are relatively precise.

Implication: Saltmarsh extent may be **underestimated** in maps.

Recommendations:

- **Investigate confusion patterns**: Examine which classes saltmarsh is confused with
- **Consider threshold adjustment**: Lower probability threshold to increase recall

9. Rare Class Handling (Salt Flat)

Finding: Salt flat shows extreme variance in some regions (Region 5: 15% F1, large std dev).

Cause: Class rarity leads to small test samples and unstable estimates.

Recommendations:

- **Stratified sampling**: Ensure sufficient salt flat samples in each train/test split
- **Balanced accuracy metrics**: Consider using balanced accuracy or weighted metrics
- **Reporting caveats**: Include sample size in accuracy tables
- **Targeted useage**: Categorical Salt flat product is not included, users can access the model probability layers instead

---

Technical Details

Dependencies

Core:

- Python ≥3.8
- pandas ≥1.3.0
- numpy ≥1.21.0
- scikit-learn ≥1.0.0
- pyarrow (for parquet reading)

Spatial:

- geopandas ≥0.10.0
- shapely ≥1.8.0

Parallel Processing:

- multiprocessing (standard library)
- joblib (via scikit-learn)

Data Requirements

Input Data Locations:

- Training data: `/gdata1/projects/coastal/cem/training_data/dea_cem_covariates_v1_2_3_combined.parquet`
- Tiles: `/gdata1/projects/coastal/cem/grids/ga_summary_grid_c3_32km_cem_tps_only.geojson`

Required Columns:

- Geometry: `lat`, `lon`
- Labels: `class_l2`, `model_region`, `auto`, `year`
- Quality: `qa_count_clear`, `qa_coastal_connectivity`
- Covariates: `s2_*`, `low_*` (131 bands total)

Output Format

Markdown Report Structure:

```
# Header (metadata)
## Sampling Method
## Ecosystem Class Mapping
## Region: X
  ### <Subset Name>
    #### Overall Accuracy (Median [95% interval])
    #### Evaluation Coverage
    #### Per-Class Accuracy (F1 as mean ± std)
    #### Year-specific summaries (2021, 2022, etc.)*
## Methodology Notes

*Year sections appear whenever the `year` column contains multiple imagery epochs.
```

Performance Tuning

For Faster Execution:

- Reduce iterations: `--iterations 100`
- Reduce RF trees: `--n-estimators 250`
- Select fewer regions: `--regions 6 7`

For Higher Precision:

- Increase iterations: `--iterations 1000`
- Increase RF trees: `--n-estimators 1000`
- Use spatial stratification (always)

Memory Optimization:

- Process regions sequentially (modify script to loop)
- Reduce multiprocessing pool size
- Clear iteration results periodically

References

Primary Methodology Reference

Lyons, M.B., Keith, D.A., Phinn, S.R., Mason, T.J., & Elith, J. (2018). A comparison of resampling methods for remote sensing classification and accuracy assessment. *Remote Sensing of Environment*, 208, 145-153. https://doi.org/10.1016/j.rse.2018.02.026

Key Findings from Lyons et al. (2018):

- Single train/test splits produce accuracy estimates ranging 40-80% due to random variation
- All resampling procedures (bootstrap, Monte Carlo CV, k-fold CV) provide accurate estimates
- Resampling enables confidence intervals that are informative about classifier uncertainty
- Spatial stratification reduces bias from spatial autocorrelation
- ~200 iterations sufficient for stable estimates, minimum 50 for variance estimation

Additional References

Spatial Cross-Validation:

- Roberts, D.R. et al. (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40, 913-925.

Accuracy Assessment Best Practices:

- Olofsson, P. et al. (2014). Good practices for estimating area and assessing accuracy of land change. *Remote Sensing of Environment*, 148, 42-57.

Resampling Applications:

- Brenning, A. (2009). Benchmarking classifiers to optimally integrate terrain analysis and multispectral remote sensing. *Remote Sensing of Environment*, 113, 239-247.
- Champagne, C. et al. (2014). A bootstrap method for assessing classification accuracy and confidence for agricultural land use mapping. *International Journal of Applied Earth Observation and Geoinformation*, 29, 44-52.

Contact & Contributions

Project Lead: Digital Earth Australia - Coastal Ecosystems Mapping Team **Technical Lead**: Mitchell Lyons (UNSW)

For Questions or Issues:

- Review validation results: coastal_ecosystem_validation_results.md
- Check methodology: Lyons et al. (2018) reference above

- Consult product documentation: Link to DEA CEM documentation

---

Document Version

- **Version**: 1.0
- **Last Updated**: 2025-10-24
- **Validation Run Date** : 2025-10-15
- **Status**: Production

Document End