1 Supplementary Information

1.1 Model based on Co-Occurring Environmental Parameters (MCE) Grid Search

Here we detail the method used in finding the optimal thresholds employed in the MCE model in our study. First, a grid search is employed over different threshold values for each environmental predictor. The grid search ensures the unfavorable thresholds for the environmental predictors always remain lower than the favorable threshold values. A grid search method was chosen to ensure an exhaustive approach to the numerous combinations of available thresholds for all 5 predictors. The grid search is conducted in 2 parts over the different multiples of each predictor's standard deviation. The first grid search looks through multiplier values of 0 through 2.5 in increments of 0.5 for each predictor. A second narrower grid search is conducted in increments of 0.1 of multiplier values centered around the value that was produced from the first grid search. The multipliers for each environmental predictor for the top 10% best performing models are averaged to avoid possibly any extreme multiplier values.

The grid search is conducted using the predictor training dataset. Each model from the grid search is evaluated against the predictand training dataset and uses the critical success index (CSI) metric to determine the best performing models. CSI (Roebber, 2009), also known as Threat Score, was chosen to evaluate the models due to its wide use in operational forecasting of severe weather events (Yang et al, 2020; Tam et al, 2021). Higher CSI scores indicate better performing models. Further, in Doswell et al (1990) and Roebber (2009)'s analyses of forecasting metrics, they underline the suitability of

CSI for forecasting rare events since the metric disregards skill from true negative (TN) predictions. This is when the model correctly predicts a non-event, which in our case is a non-RI event. In rare event forecasting, since non-events are more abundant, skill can be overinflated when metrics account for TNs, so CSI is chosen as for the grid search metric to avoid any such skill inflation.

1.2 Logistic Regression (LR) Grid Search

For a comparison with the MCE, we aim to find the best tuned LR model. The GridSearchCV 'fit' and 'score' method from the Python sci-kit learn package is used to perform a hyperparameter optimization search. From the scikit-learn package, the specific LR hyperparameters searched over are 'solver', 'c_value', 'penalty' and 'class_weight.' Class weight is chosen to aid in balancing the imbalanced training dataset where the minority class are the RI cases and majority class are the non-RI cases. The full extent of values searched for each hyperparameter are included in the table below. Optimization is performed by maximizing the CSI score for each model. The search includes different probability thresholds in 0.05 increments to produce a fully optimized model. A second narrow grid search is then implemented using probability threshold increments of 0.01 centered around the best probability threshold produced by the first coarser grid search. The specific cross-validation scheme used in the grid search is the LOYO (Leave-One-Year-Out) method where the samples that are reserved for cross validation all belong to one year. Similar to its use in Xu et al (2021), LOYO cross validation was chosen due to the temporal nature of the dataset. The best performing model returned by the grid search method, detailed in Table 1, is then tested on the unseen testing dataset.

Hyperparameter	Values Searched	Optimal Value
Probability Threshold	0-1 (increment: 0.05)	0.49
Penalty	‘l1’, ‘l2’, ‘None’	‘l2’
Solver	‘saga’, ‘liblinear’, ‘newton-cg’, ‘lbfgs’, ‘sag’	‘liblinear’
C	100, 10, 1, 0.1, 0.01	10
Class_Weight	{0:1,1:1}-{0:1,1:12} (increment: 1), ‘balanced’	{0:1,1:6}

Table 1 Logistic Regression hyperparameters searched in GridSearchCV and the returned optimal value.

1.3 Decision Tree (DT) Grid Search

To determine the best performing model, the GridSearchCV method from the Python sci-kit learn package is used to perform a hyperparameter optimization search. GridSearchCV is chosen as it is an exhaustive hyperparameter search that uses a ‘fit’ and ‘score’ method. The hyperparameters are optimized using a specified cross-validation technique. In the scikit-learn package, the DT-specific hyperparameters included in the grid search are ‘max_depth’, ‘min_samples_split’, ‘min_samples_leaf’ and ‘class_weight.’ Class_weight is included in the hyperparameter search to bolster performance of the model when dealing with the imbalanced training dataset. In increments of 1, the range starting at 1 (where the majority and minority class are equally weighted) to the value of class_weight when it is equal to ‘balanced’ for the minority class is searched. A ‘balanced’ class_weight indicates that the weights are automatically adjusted inversely proportional to the frequency of each class in the training dataset. The full extent of values searched for each hyperparameter are included in the table below. Hyperparameter tuning is optimized using a CSI metric scoring method, similar to the grid search method used when developing the MCE. Different probability thresholds in increments of 0.05 are searched to produce a fully optimized decision tree model. A second narrower grid search is then implemented using probability threshold increments of 0.01 centered around the best probability threshold produced by the first coarser grid search. The probability thresholds are implemented within the CSI scoring method. We use the same cross-validation scheme used for developing the LR model, detailed in the section above. The best performing model returned by the grid search method, detailed in table 2 is then tested on the unseen testing dataset.

Hyperparameter	Values Searched	Optimal Value
Probability Threshold	0-1 (increment: 0.05)	0.55
Max_Depth	2-10 (increment: 1), 'None'	7
Min_Samples_Split	2, 5-40 (increment: 5)	2
Min_Samples_Leaf	1, 5-20 (increment: 5)	1
Class_Weight	{0:1,1:1}-{0:1,1:12} (increment: 1), 'balanced'	{0:1,1:6}

Table 2 Decision Tree hyperparameters searched in GridSearchCV and the returned optimal value.

References

Doswell C, Davies-Jones R, Keller DL (1990) On summary measures of skill in rare event forecasting based on contingency tables. Weather and forecasting 5(4):576–585

Roebber PJ (2009) Visualizing multiple measures of forecast quality. Weather and Forecasting 24(2):601–608

Tam Hf, Choy Cw, Wong Wk (2021) Development of objective forecast guidance on tropical cyclone rapid intensity change. Meteorological Applications 28(2):e1981

Xu W, Balaguru K, August A, et al (2021) Deep learning experiments for tropical cyclone intensity forecasts. Weather and Forecasting 36(4):1453–1470

Yang Q, Lee CY, Tippett MK (2020) A long short-term memory model for global rapid intensification prediction. Weather and Forecasting 35(4):1203–1220