Supplement of Nat. Hazards Earth Syst. Sci., 24, 4585–4608, 2024
https://doi.org/10.5194/nhess-24-4585-2024-supplement

Supplement of

Automating tephra fall building damage assessment using deep learning

Eleanor Tennant et al.

Correspondence to: Eleanor Tennant (eleanorm001@e.ntu.edu.sg)

S1. Examples of images used for model development

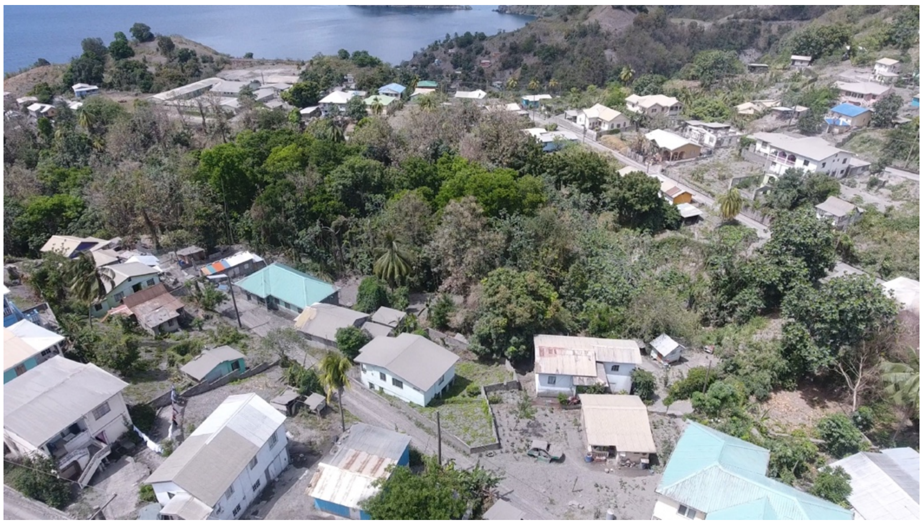

Figure S1. Representative example image from the UWI-TV-Troumaca dataset. Image credit: Javid Collins (2021) for The UWI Seismic Research Centre.

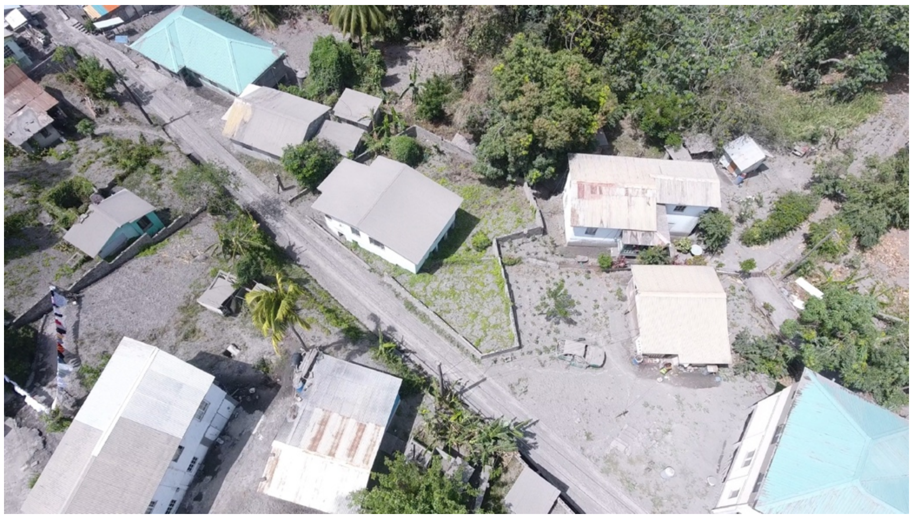

Figure S2. Representative example image from the UWI-TV-Troumaca dataset. Image credit: Javid Collins (2021) for The UWI Seismic Research Centre.

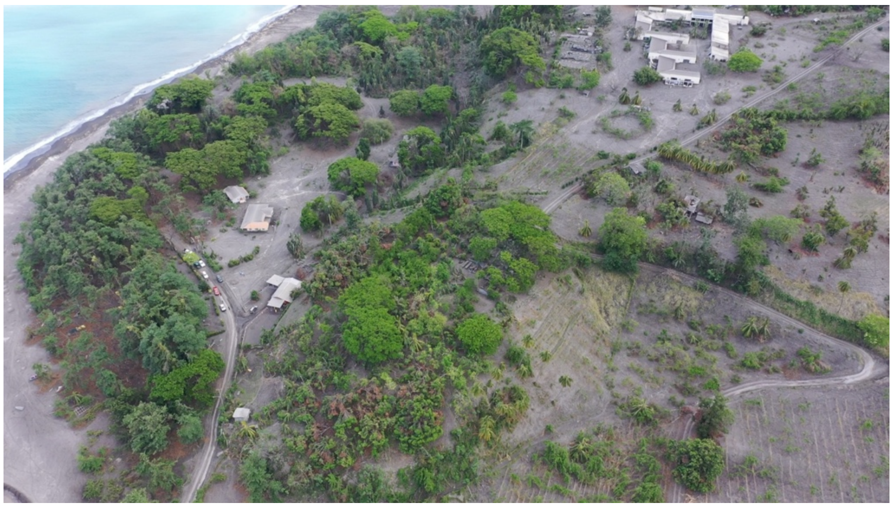

Figure S3. Representative example images from the GOV- Richmond dataset. Image credit: Damage Assessment Team, the Government of St Vincent and the Grenadines Ministry of Transport, Works, Lands and Surveys, and Physical Planning (2021).

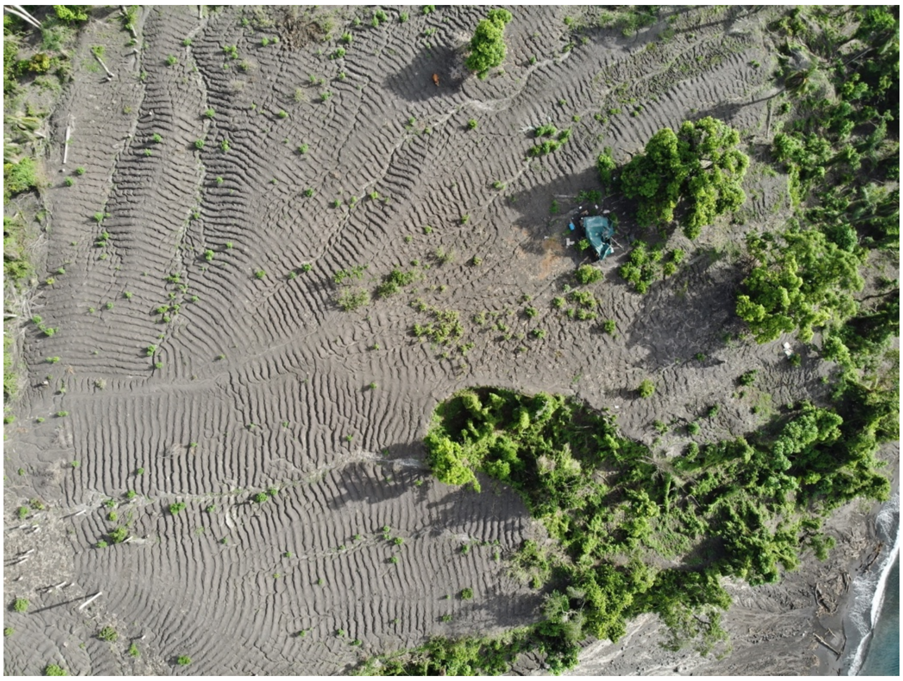

Figure S4. Representative example images from the SRC- Richmond dataset. Image credit: Richard Robertson (2021) for The UWI Seismic Research Centre.

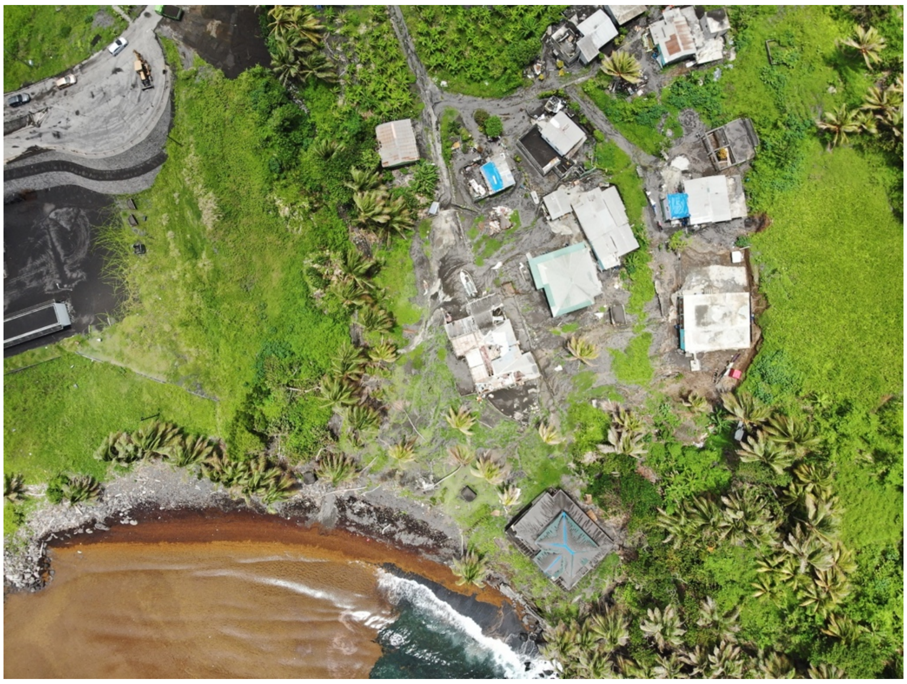

Figure S5. Representative example images from the SRC- Owia dataset. Image credit: Richard Robertson (2021) for The UWI Seismic Research Centre.

S2. Image labelling

To train models for building localisation and two levels of damage classification we labelled 2,811 image frames. In total 49,173 building bounding boxes were drawn around ~2,000 individual buildings (with some buildings being present in multiple images). Bounding boxes were drawn by a team of five including the lead author, and all boxes were checked by the lead author. Boxes were drawn to fit the buildings closely and minimise background information. In areas where buildings are close together, off-nadir images may include parts of other buildings. Nevertheless, this was not considered an issue since deep learning models for object localisation will quickly learn to ignore background pixels (Bouchard et al., 2022).

In some images tarpaulins can be seen partially or fully covering roofs (~30 buildings). These were potentially placed to cover damage that occurred during the eruption, including corrosion due to prolonged presence of tephra on metal roofs or, holes generated by nails lifted out through sub-optimal cleaning approaches (VM personal communication). Alternatively, tarpaulins may have been placed as a preventative measure to help shed tephra (e.g., Ambae

Vanuatu, Jenkins et al., 2024). Erring on the conservative side, we considered buildings with a tarpaulin to be damaged; we assessed the severity of the damage for each building based on the level of visible deformation. We assigned buildings with a tarpaulin and no visible deformation to the moderately damaged class and those with a tarpaulin and visible deformation to the major damage class.

Table S1. The approximate number of buildings covered by the UAV data for each location and dataset. Number was approximated by overlaying the UAV track with building footprints obtained from Open Street Map and does not consider the off-nadir angle of the drone. For Richmond there are no OSM buildings overlapping with the SRC UAV track (missing from OSM).

Location	**SRC**	**GOV**
Chateaubelair	933	284
Fancy	240	
Fitzhughes		
SandyBay	183	310
Tourama	178	
Belmont	18	
London	71	66
Point	70	
Rabacca dry river	15	
Orange Hill	12	
Owia	273	
Richmond		29
Troumaca		

S3. Model selection

For each of the three tasks in our tephra fall building damage assessment approach (building localisation, classification 1 and classification 2) we ran a series of experiments with the goal of iterating towards the best model. This involved training and evaluating models with different image preprocessing approaches, CNN architectures, and combinations of hyperparameters. One experiment consisted of three replicates of a given combination of these aspects. Three replicates were conducted since the stochastic nature of the training process can cause models to converge at slightly different points (Aggarwal, 2018). For each experiment the replicate with the highest evaluation metric was the one compared against the other experiments. In the following we provide details on the experiments conducted for each task and present the results.

S3.1. Building localisation

S3.1.1. Method

For building localisation, we used the cutting edge two-stage object detector Faster R-CNN (Ren et al. 2017). Faster R-CNN is an improvement on the Fast R-CNN algorithm proposed by Girshick, (2015). The improvement comprises an initial region proposal network (RPN) which speeds up performance. In Faster R-CNN, image feature maps are extracted by passing the input image through a pretrained backbone CNN. The RPN then utilises these features to generate proposals for potential object-containing areas, this is achieved by tiling a set of anchor boxes of assorted sizes across the extracted feature maps. The resulting region proposals are subsequently processed by the Fast R-CNN module, which includes a classifier that is used to determine the probability of the proposal containing an object, and a regressor that is used for adjusting the proposal box positions.

Image preprocessing

To reduce Faster R-CNN detector training and implementation time we split full-sized images (1920 x 1080 or 4056 x 3040) into overlapping blocks and resized them. We conducted experiments using two different block overlap proportions (20% and 50%) and four different block sizes: 450^2, 550^2 and 650^2 pixels, and a mixed block size. Block sizes were chosen as a tradeoff between meeting the required input dimensions of the backbone CNN (224 x 224 pixels) without significant loss of resolution through shrinking and, limiting the number of buildings that were dissected. The mixed block size was developed based on the observation that in

London SRC, Orange Hill SRC, all the GOV datasets and Chateaubelair UWI datasets buildings appear much smaller than in the other sets (Section S4). For the image sets with 'smaller' buildings we split the full-sized images into blocks of 224 x 224 pixels directly, (as opposed to splitting into boxes of 450^2/550^2 /650^2 and resizing) leading to what we term 'mixed block size' experiments. We also looked at the effect of removing very small boxes from the data since small objects (<32 x 32 pixels) are notoriously hard to detect (Lin et al., 2014). To increase variety in the training data we applied data augmentation consisting of random flipping along the X axis, scaling between 0.5-1.5 times the original image, and random colour jitters (hue=0.05, saturation=0.2, brightness= 0.3); these were chosen to represent realistic variations that can be expected in future data.

Model training

In all experiments the backbone CNN used in the Faster R-CNN detector was ResNet50 trained on the ImageNet dataset. To understand if better performance could be achieved using a model that had already 'seen' the building images before, we also conducted experiments using the best model out of the classification experiments as the backbone. The full UAV imagery dataset consists of images acquired at different viewing angles, the SRC dataset consists of buildings viewed from nadir, while the GOV and UWI sets include very off nadir imagery. To understand if a better model could be obtained by separating the data based on viewing angle, we experimented with the development of models for only the SRC portion or only the GOV and UWI (combined) portions of the dataset. In all models we used stochastic gradient descent as the optimizer, and an initial learning rate of 0.001. Stochastic gradient descent is a widely used optimizer, while the learning rate was determined through preliminary analysis. To avoid overfitting models to the training data, we used early stopping with a patience of five iterations, meaning that when the loss calculated during training starts to increase, training is stopped. We used five anchor boxes based on tests that showed the performance was not improved by using any more, while the time taken to train was greatly increased. The optimum anchor box dimensions were calculated by performing K-means clustering on the bounding boxes in the training data.

S3.1.2. Results

Results from localisation experiments evaluated on the validation data are shown in Table S2 sorted by average precision from high to low. APs ranged from 0.295 to 0.701. The best experiment (AP of 0.701) used a block size of 550 x 550, with blocks resized to meet the size of the backbone CNN, with an overlap of 50%; no pretraining was conducted and very small boxes were removed from the data.

We found that block size had an impact on model performance: experiments with the smallest block size (450 x 450) had the poorest performance, in general the 550-block size produced better results. The removal of very small boxes (< 32 x 32 pixels) from the data had a large effect on the results, all experiments with these boxes removed were at the top of the table, while experiments that did not remove the small boxes were at the bottom (Table S2). For both the 550 and 650 block sizes, experiments trained and evaluated on only the SRC data had a higher AP than the equivalent experiment trained and evaluated on all three datasets, while the experiments trained and evaluated on the UWI and GOV data had a lower AP than both. Larger block overlap (50 % as opposed to 20%) produced higher AP for the 550 and 650-block size.

Table S2. Experiments conducted for building localisation using the Faster R-CNN object detector sorted from high to low by the average precision. For each experiment three models were trained and evaluated, the model that produced the maximum average precision is presented in the table.

Row id	Block size	Mixed block size	Block overlap	Block resized	Pretrained on best classifier	Remove boxes < 32 x 32	All training/ UWI&GOV/ SRC	Max Average Precision	F1 score
1	550	N	50%	Y	N	Y	all	0.701	0.669
2	550	N	20%	Y	N	Y	all	0.700	0.668
3	550	N	20%	Y	Y	Y	all	0.700	0.642
4	650	N	50%	Y	N	Y	all	0.691	0.654
5	650	N	20%	Y	N	Y	all	0.678	0.670
6	650	N	20%	Y	Y	Y	all	0.667	0.528
7	650	Y	20%	Y	Y	Y	all	0.660	0.620
8	550	N	50%	Y	Y	Y	all	0.654	0.668
9	550	N	20%	Y	Y	Y	all	0.651	0.644
10	550	Y	20%	Y	N	Y	all	0.643	0.639

11	650	N	50%	Y	Y	Y	all	0.643	0.676
12	650	Y	20%	Y	N	Y	all	0.637	0.556
13	650	Y	20%	Y	N	N	SRC	0.637	0.604
14	650	Y	20%	Y	Y	N	GOV&UWI	0.600	0.660
15	550	Y	20%	Y	Y	N	SRC	0.566	0.585
16	550	Y	20%	Y	N	N	SRC	0.560	0.552
17	550	Y	20%	Y	N	N	all	0.559	0.591
18	650	Y	20%	Y	N	N	all	0.550	0.578
19	550	Y	20%	Y	Y	N	all	0.541	0.554
20	650	Y	20%	Y	Y	N	all	0.520	0.614
21	650	Y	20%	Y	Y	N	SRC	0.517	0.535
22	650	Y	20%	Y	N	N	GOV&UWI	0.515	0.608
23	650	N	20%	N	Y	N	all	0.502	0.506
24	550	Y	20%	Y	Y	N	GOV&UWI	0.501	0.578
25	550	N	20%	N	N	N	all	0.487	0.435
26	550	N	20%	N	Y	N	all	0.472	0.532
27	650	N	20%	N	N	N	all	0.459	0.532
28	650	N	50%	Y	Y	N	all	0.449	0.579
29	550	Y	20%	Y	N	N	GOV&UWI	0.444	0.581
30	650	N	50%	Y	N	N	all	0.433	0.53
31	450	N	20%	N	Y	N	all	0.374	0.477
32	450	N	20%	Y	N	N	all	0.325	0.481
33	450	N	20%	N	N	N	all	0.320	0.458
34	450	N	20%	Y	Y	N	all	0.295	0.464

S3.2. The sieve network

S3.2.1 Method

To improve the performance of the building localisation model we developed a sieve network that runs as an add on to the Faster R-CNN building detector. Bounding boxes produced by the detector are passed to the sieve network to filter out detections that are false positives. A false positive occurs when the detector predicts a bounding box that does not have an overlapping labelled building (i.e., detects a building when there is not one).

The dataset used for training and evaluating the sieve network consists of randomly cropped background samples from full sized images in the training and validation sets. Samples were cropped from each of the datasets, and samples containing buildings were removed until 100 no-building samples were achieved for each dataset. These samples were supplemented with an additional 10% targeted image samples on the observation that trained detectors were mistakenly detecting cars and boats. For the building dataset we stochastically sampled the equivalent number (n=990 train, 660 validation) from the building images. Experiments for the sieve network were conducted using two different CNN architectures (ResNet50 and GoogleNet), and by undertaking a grid search to find the best hyperparameter combination (learning rate, batch size, and L2 regularisation). A total of five experiments were conducted, each consisting of three replicates.

Experiments for the sieve network consisted of fine-tuning two different pretrained CNNs to determine which was better and should be used in the final model: ResNet50 (He et al., 2015) trained on the ImageNet dataset (Deng et al. 2009), and GoogleNet (Szegedy et al., 2015) trained on the places365 dataset (López-Cifuentes et al., 2019). For each experiment we conducted a grid search for the best hyperparameters: learning rate (0.0001, 0.001, 0.01, 0.1), which controls the size of the steps taken during optimisation, batch size (32, 64, 128), which is the number of images passed to the network at a time, and L2 regularisation (0.00001, 0.0001, 0.001, 0.01), which is a regularization technique used to prevent overfitting. This meant that each experiment involved training models for all 48 possible hyperparameter combinations (4x3x4). Dropout is another regularization technique that can be employed to prevent models from overfitting to the training data. For ResNet50 experiment IDs 1:4 we tested dropout probabilities of 0, 0.2, 0.4, 0.6.

S3.2.2 Results

All trained sieve networks achieved macro and class F1 scores that were > 0.973 (Table S3). The best performing experiment had a dropout of 0.6. The experiment that used the GoogleNet architecture had the poorest performance out of the five.

Table S3. Experiments conducted for the sieve network, a small network designed as an add on to the Faster R-CNN object detector. Experiments are sorted from high to low by the F1 macro score. For each experiment three models were trained and evaluated, the model that produced the maximum average precision is presented in the table.

Row ID	Architecture	Dropout	F1 building	F1 background	F1 macro
1	ResNet50	0.6	0.978	0.977	0.977
2	ResNet50	0.4	0.977	0.976	0.977
3	ResNet50	0	0.975	0.975	0.975
4	ResNet50	0.2	0.976	0.974	0.975
5	GoogleNet	0	0.973	0.973	0.973

S3.3. Building damage classification

S3.3.1. Method

Building damage classification was split into two stages that were trained and evaluated separately, Classification 1: No damage-minor damage vs combined moderate and major damage and Classification 2: Moderate damage vs Major Damage. As with experiments conducted for the sieve network, damage classification experiments consisted of fine-tuning two different pretrained CNNs to determine which was better and should be used in the final model for each classifier: ResNet50 (He et al., 2015) trained on the ImageNet dataset (Deng et al. 2009), and GoogleNet (Szegedy et al., 2015) trained on the places365 dataset (López-Cifuentes et al., 2019).

Image preprocessing

Building bounding boxes were cropped from the full-sized images and sorted by damage state into folders. Zero padding was added to each cropped building image before resizing to meet the input dimensions of the CNNs (224 x 224 pixels). To remove redundant duplicates, we cleaned each damage state folder in the training and validation sets by passing each sample through the ImageNet trained ResNet50 network to extract the features. We then calculated the cosine similarity coefficient between each image and all other images in the folder. Images with a coefficient > 0.9, a threshold that was chosen based on visual inspection, were considered near identical and were removed. Using this approach altered the split of boxes between the datasets to 74% train, 12% validation, and 13% test. The same data augmentations were applied to the images used for the classification experiments as for the detection experiments.

Model training

Class imbalance exists in our dataset, which contains many more samples of the Not damaged-minor damage class (n = ~41k) than it does the moderate and major damage classes (n = ~8k). This has been shown to influence model training as models will preferentially learn the majority class (Johnson and Khoshgoftaar 2019). To address the class imbalance, we performed experiments where we either oversampled the minority class or undersampled the majority class. As with the experiments conducted for the sieve network, for each experiment we conducted a grid search for the best hyperparameters: learning rate (0.0001, 0.001, 0.01, 0.1), batch size (32, 64, 128), and L2 regularisation (0.00001, 0.0001, 0.001, 0.01). This meant that each experiment involved training models for all 48 possible hyperparameter combinations

(4x3x4) to find the combination that produced the highest macro F1 score on the validation data. We tested dropout probabilities of 0.2, 0.4, 0.6 for a set of ResNet50 experiments.

S2.3.2. Results

Macro F1 scores for experiments conducted for classifier 1 ranged from 0.836-0.753 (Table S3), the best performing model was the ResNet architecture, trained on an unbalanced dataset with a dropout probability of 40%. This produced an F1 score of 0.962 for the Not Damaged class and 0.710 for the Damaged class. In all experiments the F1 scores for the Not Damaged class are higher than for the damaged class. We found that for Classifier 1 the ResNet architecture produced the top seven scores. The method of data balancing influenced the model performance with the unbalanced dataset producing the best performance when compared to equivalent experiments with either over or under sampling. The inclusion of dropout for a given experiment does not produce a result that is consistently better or worse than in the absence of dropout (Table S3).

For Classifier 2 the macro F1 scores ranged from 0.810-0.776 (Table S4), the maximum score was achieved using the ResNet50 architecture with an unbalanced dataset and no dropout. This produced an F1 score of 0.770 for the Moderate damaged class and 0.851 for the Major damaged class. In all experiments the F1 scores for the Major damage class are higher than for the Moderate damage class, however the difference between the classes is consistently lower than for Classifier 1. Unlike for Classifier 1 there is no consistency in the effect of data balancing method on the results.

Table S4. Experiments conducted for building damage classification 1 which classifies buildings as Not/minor damaged or Damaged. Each experiment consists of a grid search comprising 48 simulations with various combinations of learning rate, mini batch size and L2 regularization hyperparameters. For each of the 48 combinations three models were trained and evaluated, the model that produced the maximum Macro F1 score was saved to be compared with the other combinations. The results shown are the highest Macro F1 scores for each experiment. Results are ordered from high to low by the Macro F1 score.

Classifier 1						
Row ID	**Architecture**	**Class balancing: Not Balanced/ under-sampled/ over-sampled**	**Dropout**	**F1 Not Damaged**	**F1 Damaged**	**F1 Macro**
1	Resnet50	not	0.4	**0.962**	**0.710**	**0.836**
2	Resnet50	not	0	0.960	0.696	0.828
3	Resnet50	not	0.6	0.957	0.699	0.828
4	Resnet50	not	0.2	0.962	0.692	0.827
5	Resnet50	under	0	0.951	0.646	0.799
6	Resnet50	over	0.2	0.950	0.643	0.795
7	Resnet50	over	0.4	0.943	0.646	0.795
8	GoogleNet	not	0	0.952	0.633	0.792
9	Resnet50	under	0.6	0.945	0.634	0.789
10	Resnet50	under	0.2	0.940	0.634	0.787
11	Resnet50	under	0.4	0.940	0.634	0.787
12	GoogleNet	over	0	0.941	0.63	0.786
13	Resnet50	over	0	0.946	0.623	0.784
14	Resnet50	over	0.6	0.937	0.623	0.782
15	GoogleNet	under	0	0.925	0.581	0.753

Table S5. Experiments conducted for building damage classification 2 which classifies damaged buildings into Moderate damage and Major damage. Each experiment consists of a grid search comprising 48 simulations with various combinations of learning rate, mini batch size and L2 regularization hyperparameters. For each of the 48 combinations three models were trained and evaluated, the model that produced the maximum Macro F1 score was saved to be compared with the other combinations. The results shown are the highest Macro F1 scores for each experiment. Results are ordered from high to low by the Macro F1 score.

Classifier 2						
Row ID	**Architecture**	**Class balancing: Not Balanced/ under-sampled/ over-sampled**	**Dropout**	**F1 Mod damage**	**F1 Maj damage**	**F1 Macro**
1	Resnet50	not	0	**0.770**	**0.851**	**0.810**
2	GoogleNet	over	0	0.737	0.848	0.793
3	Resnet50	over	0	0.749	0.835	0.792
4	Resnet50	not	0.4	0.749	0.835	0.792
5	Resnet50	under	0.6	0.735	0.845	0.790
6	Resnet50	over	0.2	0.739	0.8371	0.788
7	GoogleNet	under	0	0.742	0.829	0.7855
8	Resnet50	under	0.2	0.735	0.832	0.784
9	Resnet50	over	0.4	0.726	0.842	0.784
10	Resnet50	under	0.4	0.730	0.836	0.783
11	Resnet50	not	0.6	0.743	0.822	0.782
12	Resnet50	over	0.6	0.731	0.829	0.78
13	Resnet50	under	0	0.718	0.839	0.778
14	Resnet50	not	0.2	0.715	0.841	0.778
15	GoogleNet	not	0	0.729	0.824	0.776

S4. Cross validation

Once we identified the best performing experimental setup for each task through model selection, we conducted K-fold cross validation to understand how the choice of training and validation data affects model performance (see Section 3.1.3, Section 3.2.2). The full image set consists of images collected by three different parties across 13 different locations on the island. To test the robustness of our models to location, we trained on nine out of the ten locations present in the combined training and validation sets and evaluated each model's performance on the remaining location. To test the robustness to the dataset, we trained models and evaluated the performance for each of the three locations that contain images from more than one dataset (e.g., Chateaubelair-GOV, Chateaubelair-UWI-TV, Chateaubelair-SRC) separately.

S5. Bounding box sizes

Bounding box sizes were notably smaller in several datasets due to the Uninhabited Aerial Vehicle altitude. When cropping full sized images into blocks we experimented with a variable size crop based on the original box sizes.

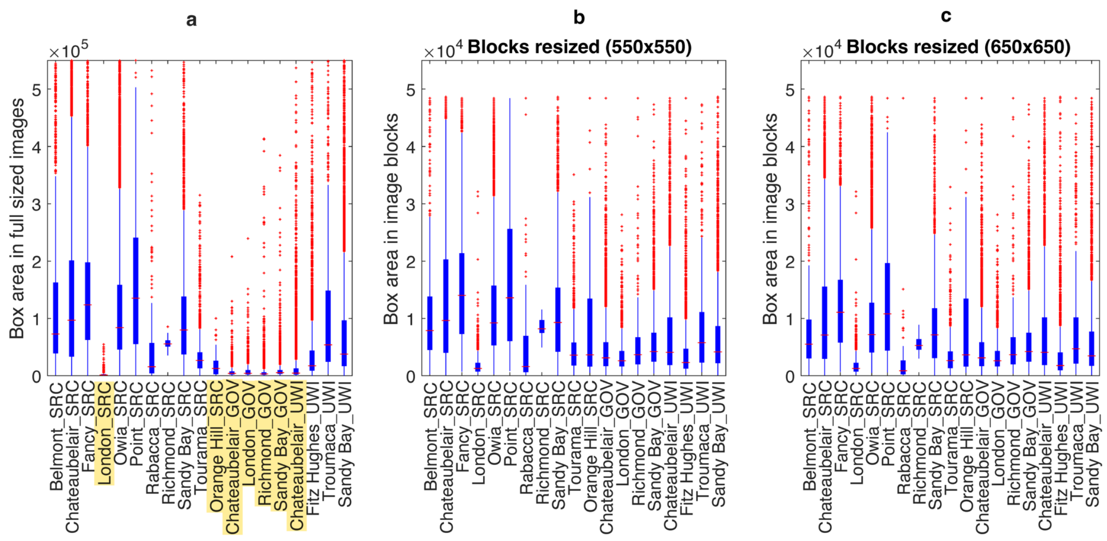

Figure S6. Box plots showing the distribution of bounding box sizes (area) in a) the full-sized images prior to cropping into image blocks, highlighted in yellow are the datasets where bounding boxes are notably smaller, b) in the extracted image blocks used for training and evaluating the object detector where block sizes are 550 x 550 pixels (resized to 224 x 224) for all data subsets besides those highlighted in a. These sets highlighted in a were cropped to 224 x 224 pixels directly. C) Full sized images were cropped to blocks of 650 x 650 and resized to 224, for data subsets highlighted images were cropped to 224 x 224 directly. For each box the red horizontal line marks the median, and the bottom and the top of the blue box marks the 25th and 75th percentiles respectively. Outliers are marked with red crosses.