

Fast building segmentation from satellite imagery and few local labels

Caleb Robinson

Microsoft AI for Good Research Lab

caleb.robinson@microsoft.com

Anthony Ortiz

Microsoft AI for Good Research Lab

anthony.ortiz@microsoft.com

Hogeun Park
World Bank

hpark2@worldbank.org

Nancy Lozano Gracia
World Bank

nlozano@worldbank.org

Jon Kher Kaw
World Bank

jkaw@worldbank.org

Tina Sederholm

Microsoft AI for Good Research Lab

tinase@microsoft.com

Rahul Dodhia

Microsoft AI for Good Research Lab

radodhia@microsoft.com

Juan M. Lavista Ferres

Microsoft AI for Good Research Lab

jlavista@microsoft.com

Abstract

Innovations in computer vision algorithms for satellite image analysis can enable us to explore global challenges such as urbanization and land use change at the planetary level. However, domain shift problems are a common occurrence when trying to replicate models that drive these analyses to new areas, particularly in the developing world. If a model is trained with imagery and labels from one location, then it usually will not generalize well to new locations where the content of the imagery and data distributions are different. In this work, we consider the setting in which we have a single large satellite imagery scene over which we want to solve an applied problem – building footprint segmentation. Here, we do not necessarily need to worry about creating a model that generalizes past the borders of our scene but can instead train a local model. We show that surprisingly few labels are needed to solve the building segmentation problem with very high-resolution (0.5m/px) satellite imagery with this setting in mind. Our best model trained with just 527 sparse polygon annotations (an equivalent of 1500×1500 densely labeled pixels) has a recall of 0.87 over held out footprints and a R2 of 0.93 on the task of counting the number of buildings in 200×200 meter windows. We apply our models over high-resolution imagery in Amman, Jordan in a case study on urban change detection.

The authors declare that the findings, interpretations, and conclusions expressed in this paper are entirely those of the authors. They do not necessarily represent the views of their affiliated organizations.

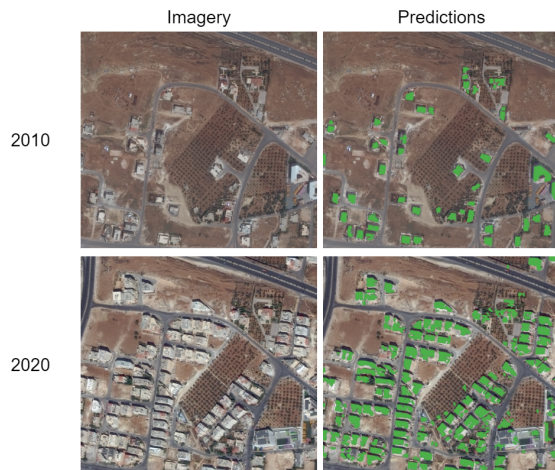


Figure 1. Example predictions from building footprint segmentation models trained from scratch using only high-resolution RGB satellite imagery and < 600 polygon labels. The imagery shows a rapidly developing part of Amman, Jordan in 2010 and 2020. Predictions are made with models trained using 527 and 367 sparse polygon labels collected from a small part of the 2010 and 2020 scenes respectively.

1. Introduction

Building footprints are a crucial piece of data in many applications, for example: disaster response damage assessments, population mapping, and urban planning [10, 24]. High-resolution satellite or aerial imagery can be hand-labeled to create accurate building footprints, however such an approach quickly becomes

prohibitively expensive as the scale of the problem increases. Model-based approaches that predict building footprints from satellite imagery can easily be applied at large scales, however must deal with problems of generalizing across different imaging conditions and geographies to produce meaningful results.

Two approaches are traditionally used to improve the generalization performance of such models: domain adaptation methods, and training with large diverse training sets. First, domain adaptation (DA) methods have been proposed for Earth Observation applications [21] to account for *distribution shifts* in satellite imagery and create generalizable models. In general, DA methods aim to obtain models robust to differences between two different datasets (a source and target) with different image and label distributions. DA methods tackle the domain shift issues by either aligning the distribution of the source set to match that of the target set [1, 5, 15] or by mapping both sets to a common space. Deng et al. proposed a scale aware adversarial learning framework for domain adaptation in building segmentation between datasets collected at different scales [5]. Tasar et al. proposed to use a Generative Adversarial Network (GAN) to generate fake synthetic training images that are semantically similar to the training images, but with spectral distribution similar to the distribution of the target imagery for better generalization of building segmentation models for different cities [20]. These approaches tend to improve overall model performance in the target domain, however the results are often suboptimal compared to supervised learning in the target domain. Second, training with larger diverse datasets can produce models with better generalization performance. Some existing open datasets for building damage assessment or building footprint segmentation, such as the xBD [10] and SpaceNet datasets [23], purposefully include labels from different geographic regions. This allows practitioners to experiment with domain adaptation methods and measure generalization performance. However, these datasets alone are not sufficient for training models that can be applied *as-is* to new satellite imagery from around the Earth [3].

Previous projects have shown positive results from building footprint segmentation models at continent level scales using a mixture of domain adaptation methods and large training datasets. For example, Google released a dataset of modeled building footprint estimates for Africa [19], Microsoft has released building footprint datasets for the continental US, Canada, Australia, and South America [13], while Facebook has released population density datasets driven by building detection models for over 160 countries [8]. The Google Open Buildings model required “manually labelling 1.75 million buildings in 100k images” [19], the Microsoft US Building footprint model was generated with access to “millions of labels” and used unsupervised learning techniques to improve generalization performance [14], while Facebook’s building mapping efforts started with “a seed dataset of around 1M labeled patches of imagery” and used a variety of techniques to improve generalization performance [4]. While these projects have all

resulted in useful *datasets* that help advance humanitarian goals, the models and imagery behind these efforts have not been open-sourced and therefore cannot be used on new imagery. This is a limitation for applications such as *building damage assessment after disasters* that require on-demand processing of new imagery and *urban change detection* that require processing multiple layers of historical imagery on-demand (versus generating a static map of current buildings).

In this work, we come at the applied problem of building segmentation from a different direction and do not attempt to train models that can generalize to new inputs at all. Instead, we ask the question, “how many labels do we need to collect from a single scene in order to train a model to an acceptable performance within that scene?”. We build a workflow in which we can quickly solicit polygon-based labels with a web application that are tied to a given satellite imagery scene, train a model using those labels and imagery, then run the model over the whole scene producing the desired predictions. Given new imagery, this workflow can be executed in less than a day of work to produce consistent results without concerns about out-of-domain generalization performance. We use this workflow in Amman, Jordan to create a detailed change map of buildings from 2010 to 2020 that can directly inform urban planning decisions.

To summarize, our contributions include:

- An analysis of the necessary components in an on-demand building segmentation modeling method with few sparse labels.
- An open-source web-based tool for collecting polygon labels over a given remotely sensed imagery scene: <https://github.com/microsoft/satellite-imagery-labeling-tool>.
- A case study applying our methods to urban change detection in Amman, Jordan.

2. Problem statement

Given a large satellite image, \mathbf{X} , and an accompanying sparse label mask, \mathbf{Y} , we want to learn the parameters of a semantic segmentation model, $f(\mathbf{X};\theta) = \hat{\mathbf{Y}}$ that can make dense predictions, $\hat{\mathbf{Y}}$, over the entire image. Here, entries in the label mask Y_{ij} indicate that the corresponding pixel in the imagery X_{ij} is either part of a building footprint, is not part of a building footprint, or is unknown. We note that with modern commercial high-resolution satellite imagery, the spatial dimensions of \mathbf{X} can exceed $100,000 \times 100,000$ pixels and cover hundreds of square kilometers. We assume that \mathbf{X} was captured at a single point in time, i.e. that the only source of variance in the imagery that f needs to capture is related to how buildings look in that particular scene (and not due other differences such as lighting, off nadir angles, seasonality, etc.). While in this work we specifically focus on the building segmentation problem, the same ideas can be applied to other earth observation tasks.

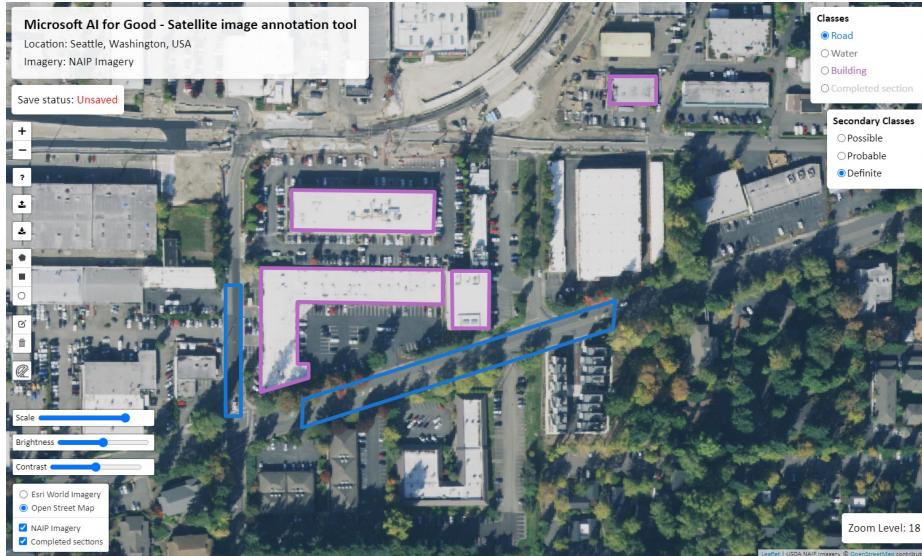


Figure 2. A web-based satellite imagery labeling tool allows us to quickly solicit polygon labels that are aligned with a particular satellite imagery scene. Users receive a link for a particular scene, then can annotate the imagery with different labels and confidence levels, and finally download the annotations in a GeoJSON format for further processing or distribution.

3. Methods

Our proposed methods are simple, to generate building footprints in a large high-resolution satellite imagery scene we 1.) sparsely label instances of buildings and non-building (or background) classes in that scene; 2.) train a semantic segmentation model from scratch using these labels; and 3.) run the trained model over the entire scene.

3.1. Satellite imagery labeling tool

Creating labels over large satellite imagery scenes is a non-trivial task. First, multispectral high-resolution satellite imagery scenes can be many gigabytes in size and therefore difficult to download locally without a dedicated internet connection. Second, these scenes usually require domain expertise and specialized desktop GIS software to visualize and annotate (for example, a user may need to select the RGB bands in the imagery and perform normalization steps to convert the spectral values into a reasonable range for visualization). Finally, once a user has created annotations they must be exported and distributed with their geographic metadata to avoid alignment issues between the annotations and imagery.

Considering these difficulties, we have created an open-source web-based tool that allows users to create annotations over basemap imagery layers, then export the annotations in a GeoJSON format for use in modeling pipelines (Figure 2 shows a screenshot of the tool). This allows a technical user with GIS expertise to create a hosted basemap from a satellite imagery scene, then simply distribute a URL in order to solicit labels from non-technical annotators. We note that this same type of setup – web-based annotation of satellite imagery – is used

in other hosted applications such as Open Street Map’s online map editor. Our implementation, however, does not require any server components other than a standard web server (optionally, a tile server for dynamically rendering of basemaps). In other words, the entire application runs in a client’s web browser. This simplifies the deployment of new instances of the application.

3.2. Modeling

We treat the problem of creating building footprints as a standard semantic segmentation problem with an additional building polygonization step (i.e. converting pixel-wise predictions of buildings into polygons) and use a U-Net model architecture [18] as implemented in the Segmentation models PyTorch library [25].

We train models using an unweighted pixel-wise cross entropy loss that ignores pixels that haven’t been labeled. Related work has proposed using weighting schemes that heavily weight the loss of pixels at the edges of building masks [12, 18, 19], using different loss functions such Focal Loss, Tversky loss, Focal-Tversky loss [19], Jaccard loss [11], or F-Beta loss [12], as well as deriving and predicting a “building-edge” class in a multi-task setting [11]. These innovations focus on reducing false positive predictions, reducing the impact of imbalanced training samples, and, importantly, encouraging the network to predict background class labels between adjacent buildings (versus “merging” adjacent buildings). In contrast to these approaches, we found it sufficient to buffer our sparse building footprint labels with a ring of “background” labels. Specifically, we assume that every pixel within 2 meters of a “building” label (and not labeled otherwise) is a “background” label. We find that models

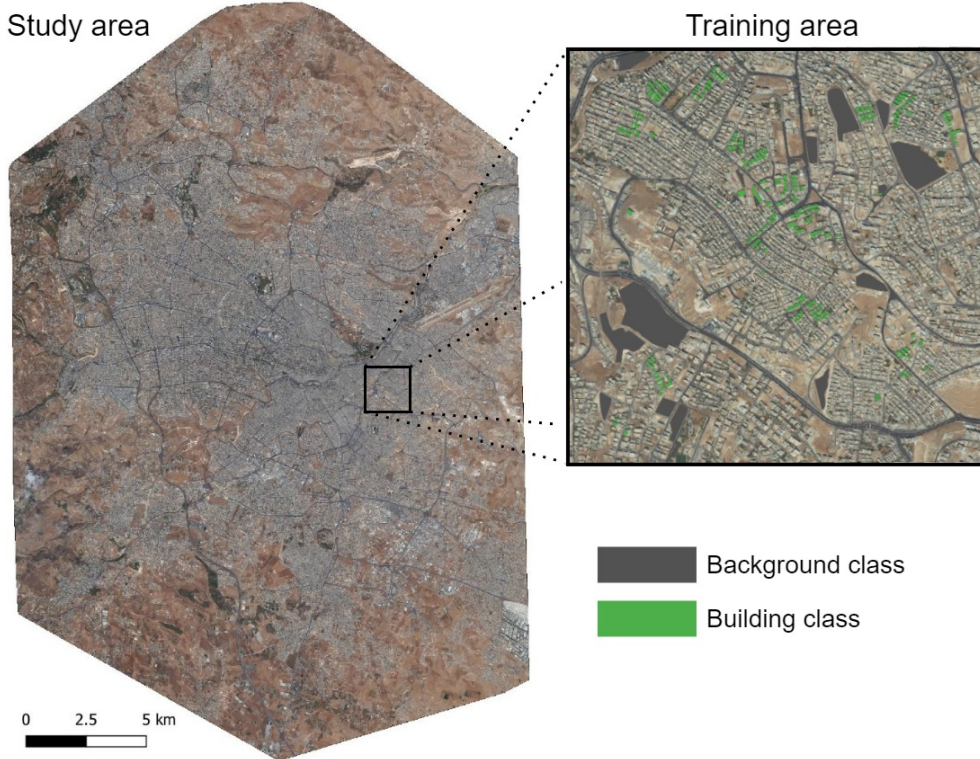


Figure 3. Overview of the Amman, Jordan study area showing the 2020 imagery layer. The map inset shows the area over which the **train** split labels were created along with the 367 training polygons for 2020.

trained with these buffered labels are able to separate buildings while models trained on the unbuffered sparse labels do not (see Section 4). Finally, we train using an AdamW optimizer, we decay learning rate by a factor of 0.1 on training loss plateaus and we use random rotation, flipping, sharpness and color jitter augmentations as implemented by the Kornia library [17]. We tune learning rate, weight decay, and weight initialization hyper-parameters based on validation set performance.

In a post-processing “polygonization” step, we convert the per-pixel predictions made by our model into building polygons for use in downstream applications. Here, we assume we have used a trained model to run inference over a whole area classifying each pixel as “building” or not. First, we run a median filter with a 7×7 kernel to remove small erroneous predictions and fill gaps in predicted buildings. Next, we convert each connected component of building pixels into a polygon (where pixel borders are replaced by edges), and simplify the polygon with a tolerance of 0.5m using the Douglas-Peucker algorithm [6] as implemented by the shapely library [9]. Finally, we remove a polygon (replacing the predictions with “background”) if it has an area that is less than 30 square meters.

3.3. Datasets

We use 4 separate high-resolution (0.5m/px) RGB satellite imagery scenes from Maxar’s WorldView-2 satellite in this study,

Date	Off Nadir Angle	Sun Elevation	Max Target Azimuth	Area
6/4/2010	18.6°	74.5°	172.7°	
6/4/2010	11.5°	74.6°	49.8°	581.7 km ²
6/15/2010	13.4°	74.7°	61.1°	
9/24/2020	27.8°	52.2°	298.9°	558.6 km ²

Table 1. World View 2 imagery metadata from the scenes used over Amman.

detailed in Table 1. We create a mosaic covering Amman with the three scenes from June 2010, and use this as the **2010** layer, and use the single scene from September 2020 as the **2020** layer.

We use the labeling tool described in Section 3.1 to collect four sets of labels, or *splits*, over each layer of imagery:

Train consists of sparse polygon labels created over the same 4km² area from both layers. These labels include “background”, “road”, and “building” classes.

Val consists of dense “building” class labels created over a subset of the 4km² training area (that does not contain labels in the **train** split). Here, we label each building in a small area so that we can assume that all *unlabeled* pixels in that area are not buildings (i.e. background or

Split	Layer	Background	Road	Building	Totals
Train	2010	59	31	437	527
	2020	27	27	313	367
Val	2010	-	-	40	-
	2020	-	-	41	-
Test	2010	-	-	344	-
	2020	-	-	467	-
Test counts	2010	-	-	-	29 counts
	2020	-	-	-	29 counts

Table 2. Size of the different splits (in number of polygons) for the 2010 and 2020 layers over Amman.

road). This allows us to measure the pixel-wise precision and recall for the “building” class during training.

Test consists of sparse polygon “building” class labels created by randomly labeling buildings over the entirety of each layer. We hold these labels out during training and use them to approximately measure the recall of our models over the entire layers. Note that we cannot measure precision using this set of labels as knowing whether a prediction is a false positive requires knowing where *all* true positives are.

Test counts consists of counts of the number of buildings over 29 randomly sampled 200×200 meter polygons from each layer. Similar to the **test** set, we hold these labels out during training. This allows us to measure how well our models capture the density of actual buildings in each layer. For example, a trivial model that labels every pixel as the “building” class would have 100% recall, however the *count* of the number of buildings predicted by this model would be completely uncorrelated with the actual number of buildings in an area.

Figure 3 shows an overview of the study area and the small area used for training in Amman, Jordan for 2020. Notice how the training area is less than 1% of the total study area with only 367 training polygons available.

Table 2 shows the number of labels of each class we collected over each layer. We differentiate between “background”, “road”, and “building” classes in order to test whether a finer grained classification can improve the performance of models on the building class. Finally, we estimate that an experienced user can create ~ 6 labels per minute (our average rate). At this rate it would take a total of ~ 5 hours of labeling effort to reproduce the **train**, **val** and **test** splits we use here. Reproducing the **test counts** dataset is similarly easy as it only involves counting and recording the number of buildings over a few fixed areas.

3.4. Baseline methods and performance metrics

We compare our modeling approach against off-the-shelf building segmentation models from [12] and to a random forest

model from the scikit-learn library [16].

Jiwani et al. provide pre-trained weights for three models trained on the SpaceNet, CrowdAI, and Urban3D datasets [12]. Identical to our problem setting, these models are trained to segment buildings from RGB satellite imagery, and notably, were the *only* publicly available models we could find for this task. We use these models *as-is* to test how the best off-the-shelf approaches will work when applied on new imagery. That is, to simulate how a practitioner may try to use the models in an applied setting. We expect that these models would perform better if fine-tuned with labeled data from our two layers of imagery, and we expect that the methods proposed in [12] are relevant to our task however have left this investigation to future work.

The local random forest model uses the RGB values at a pixel as a feature representation and is trained over the labeled pixels in each layer’s **train** split. We merge the road and background classes into a single class to reduce the problem to a binary classification. This serves as very simple color-based segmentation of the imagery and a lower bound on what should be possible for other approaches.

We test the predictions made by each method in three ways:

- Pixel-wise F1 score of the building class using the dense labels from the **val** splits.
- Recall@ k using the building level labels from the **test** splits. We define Recall@ k as the percentage of buildings in which $\geq k\%$ of the pixels in a building are correctly predicted as the building class. In other words, a labeled building in the **test** split is counted as a true positive if $k\%$ of the pixels in that building are predicted as the building class, else it is counted as a false negative.
- Coefficient of determination (R2) calculated between the counts of buildings over the 29 areas in the **test count** splits compared to the predicted number of buildings over the same areas. Specifically, for each of the 29 200×200 meter areas in the **test count** split, we compute the number of model predicted buildings by totaling the number of polygons that intersect with that area after the polygonization post-processing step described in Section 3.2.

4. Experiments and results

Table 3 shows the performance of our best performing U-Net model trained with local labels along with the baseline method performance. We observe that the CrowdAI pretrained model and U-Net models are the only models that make reasonable predictions. Here, the CrowdAI model makes a large number of false positive predictions and frequently ignores boundaries between adjacent buildings in both layers of imagery. This results in high recall scores, but predictions that don’t correlate as well with the actual number of buildings over an area – achieving 0.76 and 0.67 R2 scores for 2010 and 2020 respectively. The

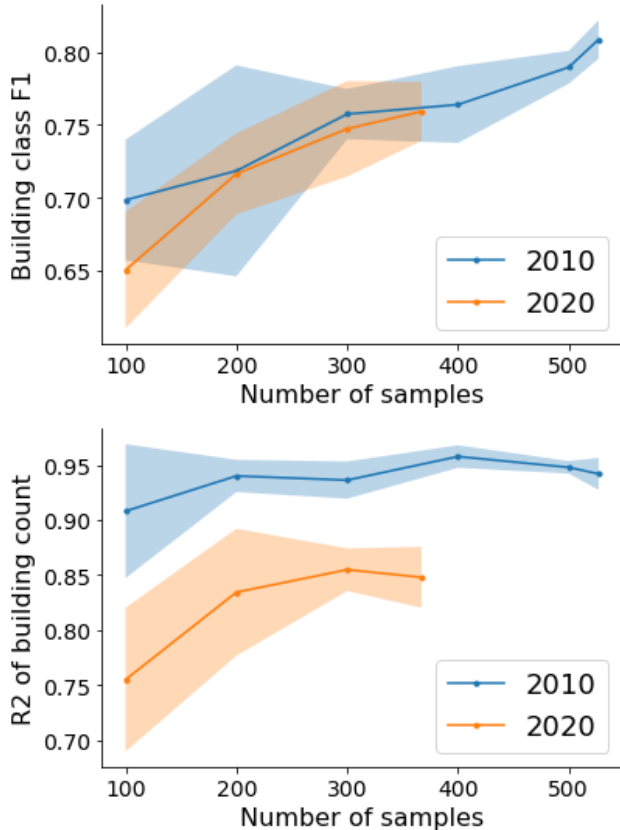


Figure 4. The effect of the number of training samples on U-Net model performance in the 2010 and 2020 layers. (Top panel) shows the impact on the building class F1 score as measured on the **val** splits. (Bottom panel) shows the impact on the R2 of the predicted building footprint counts as measured on the **test counts** splits.

U-Net, in comparison, has a slightly lower recall scores (86.9 vs. 90.7 and 83.7 vs. 89.1) but is able to consistently segment individual buildings. This results in predictions that *are highly correlated with the actual number of buildings* – achieving 0.93 and 0.84 R2 scores. The SpaceNet model achieves near perfect recall, but with a 0 R2 score, as it predicts the building class for almost any input. As expected, the random forest model does not perform well as per-pixel color features alone are not sufficient to identify buildings in high-resolution imagery.

We also investigate the impact of the number of labels used on the performance of the best U-Net model configurations. Here, we subsample different numbers of polygons without replacement and train the U-Net with the reduced set of labels. Figure 4 shows the averaged performance over 5 different subsamples / random weight initializations (± 1 standard deviation is shown in the shaded area). We observe a consistent increase in building class F1 scores across both layers of imagery up to the maximum number of labels we gathered which suggests that more labels would continue to increase performance of the model locally and produce better footprints. However, we also

observe an immediate plateau in the R2 counting performance of the models, and large standard deviations in performance. This suggests that even 100 to 200 labels is sufficient to train a building footprint model especially if the predictions made by that model are used primarily for building density estimation tasks. The fact that some sets of 100 labels achieve very high R2 values in the 2010 layer further suggest that *which* labels are used in training are more important than how many are used. Overall, we find that the performance of the models on the 2020 layer is uniformly worse than on the 2010 layer. One explanation for this is that the 2020 imagery was taken with a greater off-nadir angle than the 2010 imagery (Table 1) and is of generally lower quality.

Finally, ablation studies to investigate the effect of different modeling choices. Specifically, we test the impact of merging the “road” class with the “background” class or not, the impact of using ImageNet pre-trained weights versus random weight initialization, and the effect of *buffering* the “building” class labels with a small ring of “background” class labels. We run each combination of these choices for several learning rate and weight decay parameters. Table 4 show the building class F1 scores for each choice averaged over all valid combinations (e.g. the score of 0.710 for “Road class separate” in 2010 is averaged over all other runs in the grid that were trained using “road” as a unique class). First, we observe that merging the road class with the background class results in slightly better performance in both layers of imagery. Second, we find that buffering the “building” class does not impact the results in 2010, however dramatically improves the performance in the more off-nadir 2020 imagery. Finally, we find that starting from a random weight initialization is *better* than starting from ImageNet pre-trained weights, with a large improvement in 2010. We observe that the models that were trained starting from ImageNet weights are often able to overfit completely, achieving close to 0 training loss but producing nonsensical predictions elsewhere in the imagery.

Method	Layer	Recall @ 0.7	R2
Local random forest	2010	21.22%	0.76
SpaceNet model	2010	99.71%	0.00
CrowdAI model	2010	90.70%	0.76
Urban3D model	2010	21.22%	0.05
Local U-Net	2010	86.92%	0.93
Local random forest	2020	31.26%	0.22
SpaceNet model	2020	99.57%	0.05
CrowdAI model	2020	89.08%	0.67
Urban3D model	2020	25.48%	0.01
Local U-Net	2020	83.73%	0.84

Table 3. Model performance on each layer of imagery over Amman evaluated using the **test** and **test count** splits. The SpaceNet, CrowdAI and Urban3D models are pre-trained models from [12] and are *not* fine-tuned. Best values in each column are shown in bold.

Road merging	Road class separate	Road class merged
2010	0.710	0.728
2020	0.674	0.687
Building buffering	Buildings not buffered	Buildings buffered
2010	0.720	0.718
2020	0.649	0.710
Weight initialization	ImageNet	Random
2010	0.670	0.768
2020	0.664	0.697

Table 4. Ablation results showing the average impact of different modeling choices. Values are building class F1 score measured.

5. Case study in Amman

Jordan has received over 670,000 refugees since the beginning of the Syrian conflict in 2011 [22]. Although overall violence has declined in the region, poor economic performance and limited basic services in hosting cities exacerbate residents’ living conditions and livelihoods. Given that the vast majority of the refugees in Jordan, 83 percent, live in cities instead of camps [22], the influx of people has caused severe challenges in maintaining and providing basic infrastructure and public services. However, due to limited planning resources and the excessive number of incoming refugees, our understanding of how Amman has changed within this period is limited. Thus, we use the methods described above to create building footprint layers for the 2010 and 2020 imagery, create density maps, and use these maps to identify growth at an urban scale.

From Table 3 we know that the *count* of buildings predicted by our model are highly correlated with the true count of buildings over 200×200 meter windows. However, our model could systematically over or under predict buildings and still have highly correlated counts. To account for this and derive counts that accurately reflect the number of buildings on the ground, we can fit a linear regression model using the predicted and actual counts, then adjust our model’s predicted counts over the same sized windows using the slope and intercept of the model. After this procedure our models *adjusted* predicted counts have a root-mean-squared error of 2.81 in 2010 and 5.13 in 2020. This allows us to directly compare the counts between the two years of imagery to see *where* in Amman new buildings were created between 2010 and 2020.

Over the entire study area we find $\sim 138,500$ buildings in 2010 and $\sim 150,500$ buildings in 2020, for an estimated growth of 12,000 buildings. Figure 5 shows the spatial distribution of growth, i.e. the difference in number of buildings between 2020 and 2010 over 0.25 square kilometer grid cells. We observe that the north-west portion of the city has experienced the largest amount of growth.

The change in number of buildings over time sheds light on the spatial configuration of the city. In a post-conflict setting,

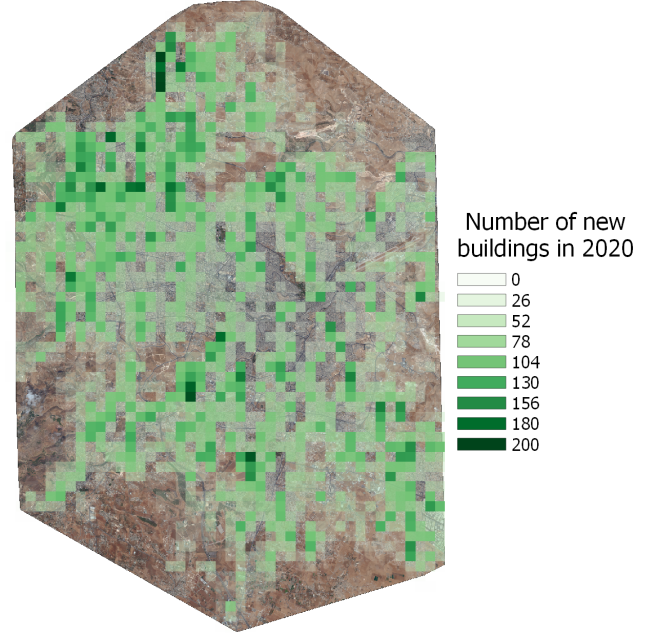


Figure 5. Map of the *increase* in number of buildings between 2010 and 2020 over a 0.25 square kilometer grid in Amman, Jordan.

official statistical surveys (i.e., census) often cannot be carried out and/or do not fully address emerging population growth due to limited institutional resources and long-time span of survey instruments. However municipal and central governments still need to allocate fiscal/institutional resources to align with the broader goals of city development (e.g., solid waste management and water and sewer services). Building change maps enable us to understand when/where new buildings were constructed and thus estimate the associated needs of infrastructure and basic service provision. The findings in this case study will be used to carry out strategic policy dialogue with the local and central government.

Low-quality building classification We can further investigate the predicted building footprints to understand quality of buildings in the city of Amman. In the context of post-conflict and disaster setting, incoming refugees and migrants are often found living in poor quality housing that needs immediate assistance. Here, we define the quality of buildings based on their morphological features (e.g., size, density, shape, slope, etc.). From our partner’s local experience in Amman we know low quality buildings are unevenly distributed within a plot, small, irregularly shaped, often built on steep terrain, and are closely packed with surrounding buildings (i.e., narrow gap between buildings).

First, we extract building level features from the predicted footprint layers, following methodology similar to [2, 7]. Specifically, for each building we determine: the area of the footprint, the ratio of the area of the minimum bounding



Figure 6. Low-quality housing neighborhood in Amman (photo taken on Feb 6, 2022)

rectangle to the area of the footprint, the number of other buildings in a 200-meter radius, the distance to the nearest building, the maximum slope of the ground the building is on, and the number of corners in the building.

Second, we generate labels indicating whether a building is either “Regular” or “Low-quality” in two ways: 1) through visual inspections of the high-resolution imagery with a series of consultations with local experts, and 2) through a field visit to take geo-coded “ground truth” photos. During the field visit, which was held in Feb 3-8, 2022, we took 308 photos near Al Nathif, Al Akhdar, and Al Zohour. As shown in Figure 6, these unplanned areas suffer from overcrowding and limited basic service and infrastructure provisions. Overall, we collected a dataset that positively identifies 399 “Regular” buildings and 105 “Low-quality” buildings using the 2020 layer of imagery.

Using the above features and labels, we train a gradient boosting classifier 50 times on random 25%/75% data splits and report the average and standard deviation F1 score per class. Here, we observe F1 scores of 0.97 ± 0.01 and 0.89 ± 0.03 for the regular and low-quality classes respectively. We find that the number of neighbors in 200m of a building is the most important feature in the model, with an importance of 87.23%, which aligns with how the local experts evaluate the imagery. Importantly, the trained gradient boosting classifier is now able to be run on any layer of predicted footprints (despite only having labels from recent years) after the same feature extraction step. Figure 7, for example, shows the predictions of the model over part of predicted building footprint layer for 2010. This allows us to further break down the changes in Amman by predicted building quality.

6. Conclusion

Building footprint segmentation from high-resolution imagery is an important task in many urban planning and disaster response applications. However, creating models that generalize

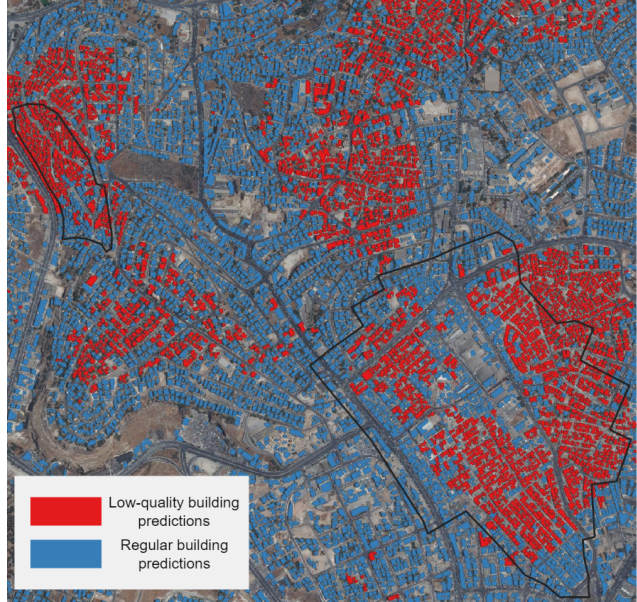


Figure 7. Map of predicted building footprints along with their predicted regular or low-quality classification in a section of Amman in 2010. Black outlines show known refugee camp locations. The model predictions align with on-the-ground knowledge.

well for on-demand building footprint segmentation is difficult due to differences in input imagery. We propose a workflow to bypass problems in generalization that simply involves creating labels for a new scene, training a building segmentation model from scratch, then running the model over the scene. We argue that this workflow is easily reproducible in *any* location with several hours of labeling efforts. As an example, we successfully apply this method in a case study mapping buildings over time in Amman, Jordan to quantify urban change.

Acknowledgements

This work was made possible by a grant from Microsoft’s AI for Humanitarian Action program.

References

- [1] Nassim Ammour, Laila Bashmal, Yakoub Bazi, Mohamad Mahmoud Al Rahhal, and Mansour Zuair. Asymmetric adaptation of deep features for cross-domain classification in remote sensing imagery. *IEEE Geoscience and Remote Sensing Letters*, 15(4):597–601, 2018. 2
- [2] Qonita P Ashilah, Revi Hemina, et al. Urban slum identification in bogor tengah sub-district, bogor city using unmanned aerial vehicle (uav) images and object-based image analysis. In *IOP Conference Series: Earth and Environmental Science*, volume 716, page 012133. IOP Publishing, 2021. 7
- [3] Vitus Benson and Alexander Ecker. Assessing out-of-domain generalization for robust building damage detection. *arXiv preprint arXiv:2011.10328*, 2020. 2

- [4] Derrick Bonafilia, James Gill, Saikat Basu, and David Yang. Building high resolution maps for humanitarian aid and development with weakly-and semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2019. 2
- [5] Xueqing Deng, Yi Zhu, Yuxin Tian, and Shawn Newsam. Scale aware adaptation for land-cover classification in remote sensing imagery. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2160–2169, 2021. 2
- [6] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2):112–122, 1973. 4
- [7] Noah J Durst, Esther Sullivan, Huiqing Huang, and Hogeun Park. Building footprint-derived landscape metrics for the identification of informal subdivisions and manufactured home communities: A pilot application in hidalgo county, texas. *Land Use Policy*, 101:105158, 2021. 7
- [8] Facebook Connectivity Lab and Center for International Earth Science Information Network. High Resolution Settlement Layer (HRSL). <https://dataforgood.facebook.com/dfg/tools/high-resolution-population-density-maps>, 2016. 2
- [9] Sean Gillies et al. Shapely: manipulation and analysis of geometric objects, 2007–. 4
- [10] Ritwik Gupta, Richard Hosfelt, Sandra Sajeev, Nirav Patel, Bryce Goodman, Jigar Doshi, Eric Heim, Howie Choset, and Matthew Gaston. xbd: A dataset for assessing building damage from satellite imagery. *arXiv preprint arXiv:1911.09296*, 2019. 1, 2
- [11] Vladimir Iglovikov, Selim Seferbekov, Alexander Buslaev, and Alexey Shvets. Terausnetv2: Fully convolutional network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 233–237, 2018. 3
- [12] Aatif Jiwani, Shubhrakanti Ganguly, Chao Ding, Nan Zhou, and David M Chan. A semantic segmentation network for urban-scale building footprint extraction using rgb satellite imagery. *arXiv preprint arXiv:2104.01263*, 2021. 3, 5, 6
- [13] Microsoft. Building Footprints. <https://www.microsoft.com/en-us/maps/building-footprints>, 2021. 2
- [14] Microsoft. US Building Footprints. <https://github.com/microsoft/USBuildingFootprints>, 2021. 2
- [15] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyunghnam Kim. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4500–4509, 2018. 2
- [16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 5
- [17] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3674–3683, 2020. 4
- [18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 3
- [19] Wojciech Sirko, Sergii Kashubin, Marvin Ritter, Abigail Annkah, Yasser Salah Eddine Bouchareb, Yann Dauphin, Daniel Keysers, Maxim Neumann, Moustapha Cisse, and John Quinn. Continental-scale building detection from high resolution satellite imagery. *arXiv preprint arXiv:2107.12283*, 2021. 2, 3
- [20] Onur Tasar, SL Happy, Yuliya Tarabalka, and Pierre Alliez. Colormapgan: Unsupervised domain adaptation for semantic segmentation using color mapping generative adversarial networks. *IEEE Transactions on Geoscience and Remote Sensing*, 58(10):7178–7193, 2020. 2
- [21] Devis Tuia, Claudio Persello, and Lorenzo Bruzzone. Recent advances in domain adaptation for the classification of remote sensing data. *arXiv preprint arXiv:2104.07778*, 2021. 2
- [22] UNHCR. Jordan fact sheet, September 2021. 7
- [23] Adam Van Etten, Dave Lindenbaum, and Todd M Bacastow. Spacenet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*, 2018. 2
- [24] Yongyang Xu, Liang Wu, Zhong Xie, and Zhanlong Chen. Building extraction in very high resolution remote sensing imagery using deep learning and guided filters. *Remote Sensing*, 10(1):144, 2018. 1
- [25] Pavel Yakubovskiy. Segmentation Models Pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2020. 3